CAD-Deform: Deformable Fitting of CAD Models to 3D Scans

Vladislav Ishimtsev*[1], Alexey Bokhovkin*[1], Alexey Artemov[1], Savva Ignatyev[1], Matthias Niessner[2], Denis Zorin[3,1], and Evgeny Burnaev[1]

[1] Skolkovo Institute of Science and Technology, Russia
[2] Technical University of Munich, Germany
[3] New York University, USA

Abstract. Shape retrieval and alignment are a promising avenue towards turning 3D scans into lightweight CAD representations that can be used for content creation such as mobile or AR/VR gaming scenarios. Unfortunately, CAD model retrieval is limited by the availability of models in standard 3D shape collections (*e.g.*, ShapeNet). In this work, we address this shortcoming by introducing CAD-Deform[1], a method which obtains more accurate CAD-to-scan fits by non-rigidly deforming retrieved CAD models. Our key contribution is a new non-rigid deformation model incorporating smooth transformations and preservation of sharp features, that simultaneously achieves very tight fits from CAD models to the 3D scan and maintains the clean, high-quality surface properties of hand-modeled CAD objects. A series of thorough experiments demonstrate that our method achieves significantly tighter scan-to-CAD fits, allowing a more accurate digital replica of the scanned real-world environment while preserving important geometric features present in synthetic CAD environments.

Keywords: scene reconstruction, mesh deformation

1 Introduction

A wide range of sensors such as the Intel RealSense, Google Tango, or Microsoft Kinect can acquire point cloud data for indoor environments. These data can be subsequently used for reconstructing 3D scenes for augmented and virtual reality, indoor navigation and other applications [25,32,33,42,40,12]. However, available 3D reconstruction algorithms are not sufficient for many applied scenarios as the quality of the result may be significantly affected by noise, missing data, and other artifacts such as motion blur found in real scans, disabling reconstruction of fine-scale and sharp geometric features of objects. In most instances, reconstructions are still very distant from the clean, 3D models created manually.

An approach to overcome these problems has been proposed in [36,30] and more recently developed using modern ML methods in [5,6]. Building on the

* equal contribution
[1] The code for the project: `https://github.com/alexeybokhovkin/CAD-Deform`

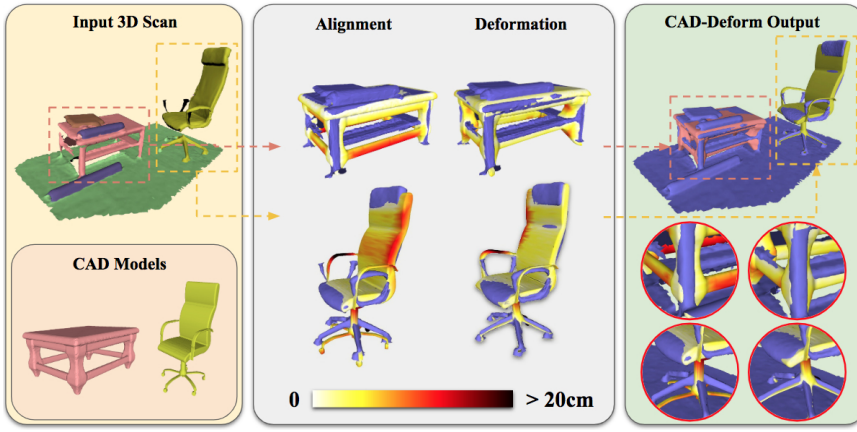

Fig. 1: CAD-Deform takes as input a set of 3D CAD models aligned on a RGB-D scan (left). In order to achieve tight fits (middle), we propose a novel part-based deformation formulation that maintains the desired CAD properties such as sharp features.

availability of parametric (CAD) models [9,27] they perform reconstruction by finding and aligning similar CAD models from a database to each object in a noisy scan. To realize this approach, the authors of [5] introduced the Scan2CAD dataset comprising of pairwise keypoint correspondences and 9 DoF (degrees of freedom) alignments between instances of unique synthetic models from ShapeNet [9] and reconstructed scans from ScanNet [11]; in order to find and align CAD models to an input scan, they developed a deep neural model to predict correspondences, with a further optimization over potential matching correspondences for each candidate CAD model. The difference of [6] compared to the approach from [5] is an end-to-end procedure, combining initially decoupled steps to take into account additional feedback through the pipeline by learning the correspondences specifically tailored for the final alignment task.

However, geometric fidelity achieved between scans and CAD objects remains limited. CAD model geometry (clean and complete) differs significantly from scan geometry in low-level geometric features. As these methods focus on finding alignments by optimizing a small number of parameters (9 DoF), resulting alignments only roughly approximate scans, not capturing geometric details such as variation in 3D shapes of parts of individual objects.

In contrast, to improve geometric fidelity while keeping the benefit of mesh-based representations, we propose to increase the number of degrees of freedom by allowing the CAD objects to *deform* rather than stay rigid. In this work, we introduce a deformation framework CAD-Deform, which significantly increases the geometric quality of object alignment regardless of the alignment scheme. For an input scan, given object location and 9 DoF alignment of a potentially matching CAD model, we apply a specially designed mesh deformation procedure resulting in a more accurate shape representation of the underlying object.

The deformation matches each semantic part of a 3D shape extracted from the PartNet [31] to the corresponding data in the scans and keeps sufficient rigidity and smoothness to produce perceptually acceptable results while minimizing the distance to scanned points. Thus, even if the initial location and alignment are not entirely accurate, the deformation can compensate to a significant extent for the discrepancy.

Our approach builds highly detailed scene descriptions with a high level of semantic accuracy for applications in 3D graphics. The approach outperforms state-of-the-art methods for CAD model alignment and mesh deformation by 2.1–6.3% for real-world 3D scans. To the best of our knowledge, the approach we propose is the first to use mesh deformation for scan-to-CAD alignment and real-world scene reconstruction. In summary, our work has several contributions:

- We developed a mesh deformation approach that 1) is computationally efficient, 2) does not require exact correspondences between each candidate CAD model and input scan, and 3) provides perceptually plausible deformation thanks to a specially introduced smoothness term and inclusion of geometric features of a CAD model in the optimization functional.
- We developed a methodology to assess the fitting accuracy and the perceptual quality of the scan-to-CAD reconstruction. The methodology includes standard data fitting criteria similar to Chamfer distance to evaluate alignment accuracy, complimentary local and global criteria for visual quality assessment of resulting deformations, and a user study.
- We performed an ablation study to assess the influence of inaccuracies in the initial object location and alignment on the final reconstruction results. For that we used both ground-truth alignments from Scan2CAD dataset [5] along with predictions of their method, and alignments trained in the end-to-end fashion [6]. We compared results with the state-of-the-art methods for mesh deformation to highlight the advantages of our approach.

2 Related work

RGB-D scanning and reconstruction. RGB-D scanning and reconstruction are increasingly widely used, due to the availability of commodity range sensors and can be done both in real-time and offline modes. There are many methods for RGB-D-based real-time reconstruction such as KinectFusion [25], Voxel Hashing [33] or Bundle Fusion [12] that use the well-known volumetric fusion approach from [10]. ElasticFusion [42] is a representative offline approach to the reconstruction. These methods can produce remarkable results for large 3D environments. However, due to occlusions and imperfections of existing sensors, the reconstructed scenes contain many artifacts such as noise, missing surface parts, or over-smooth surfaces. Some methods aim to predict unobserved or corrupted parts of 3D scans from depth data. In [18], the world is modeled as a grid of voxels representing the signed distance to the nearest surface. Then structured Random Forest is used to predict the value of the signed distance function for each of the voxels computed to form a final occupancy estimate for unobserved

voxels. [37] is to encode a depth map as a 3D volume and then aggregate both local geometric and contextual information via a 3D CNN to produce the probability distribution of voxel occupancy and object categories for all voxels inside the camera view frustum. Another approach is [13], where a 3D CNN architecture predicts a distance field from a partially-scanned input and finds the closest 3D CAD model from a shape database. By copying voxels from the nearest shape neighbors, they construct a high-resolution output from the low-resolution predictions, hierarchically synthesize a higher-resolution voxel output, and extracts the mesh from the implicit distance field. However, although all these methods can complete partial scans and improve 3D geometry reconstruction, the quality of the results still is far from artist-created 3D content.

CAD model alignment. Instead of reconstructing 3D geometry in a bottom-up manner, one can perform reconstruction by retrieving CAD models from a dataset and aligning them to the noisy scans. Matching CAD models to scans requires extracting 3D feature descriptors; thus, approaches have been proposed for 3D feature extraction. Hand-crafted features are often based on various histograms of local characteristics (*e.g.*, [16]). Such approaches do not generalize well to inherent variability and artifacts in real-world scans. Deep learning approaches lead to further improvements: *e.g.*, [43] propose a view consistency loss for a 3D keypoint prediction network based on RGB-D data; [14] develop 3D local learnable features for pairwise registration. After the descriptors are extracted, one can use a standard pipeline for CAD-to-scan alignment: first, matches between points based on 3D descriptors are identified, and then variational alignment is used to compute 6- or 9-DoF CAD model alignments. Typical examples, realizing this two-step approach, are described in [36,30,28,5,6]. The most recent ones [5,6] use learnable descriptors and differ in that the latter considers an end-to-end scan-to-CAD alignment, reconstructing a scene in a single forward pass. An obvious limitation of these two-step pipelines is that the resulting alignments approximate scans only coarsely due to a pre-defined set of available CAD models and a highly constrained range of transformations. Other approaches in this category [4,21,22], although relying on the same two-step strategy, use only a single RGB or RGB-D input.

Mesh deformation. To improve surface reconstruction accuracy and obtain a more faithful description of geometric details (*e.g.*, preserve distinctive geometric features), it is desirable to consider a more general space of deformations than a simple non-uniform scaling. To this end, a mesh deformation approach based on Laplacian surface editing is presented in [39]. Another iterative mesh deformation scheme [38] imposes rigidity on the local transformations. Despite their conceptual simplicity, both methods require specifying correspondences between mesh vertices and scan data points. The same is true for [29,1,3] and unsuitable in our setting, due to extremely low (2–8) number of correspondences available per part, that additionally may not even be well defined for noisy and incomplete real-world scans. Many methods [1,8,34,15] focus on automatic posing of human models, dense surface tracking and similar applications. However, while producing compelling results, the methods implicitly use the assumption that a tem-

plate mesh and its target shape are very similar; as a result, the semantic parts of individual 3D objects are not changed either in shape or relative scale. In our work, we are comparing to ARAP (as-rigid-as-possible) [38] and Harmonic [7,26] mesh deformation methods, however, with an added Laplacian smoothing term to leverage second-order information about the surface. This modification makes ARAP/Harmonic similar to [1,29] as far as non-data-dependent energy terms are concerned. Other methods exist that propose non-linear constraints [23,20,19]. The energy terms of our framework were designed as a natural match for our problem: we define local 3D transformations on the CAD model, mapping each subpart to the 3D scene volume, and require smooth changes of these transformations over the model. In contrast to most deformation methods, we do not aim to keep the surface close to isometric: *e.g.*, a table reference model can be stretched highly anisotropically to match a different table in the data. Our energy is defined using local 3D affine transform primitives, penalizing sharp changes, without penalizing anisotropic deformations. These primitives also allow us to express 1D feature preservation simply, and the non-data terms are quadratic which is critical for our efficient preconditioned optimization. Methods in [23,20,19] propose non-linear energies, focusing on large rotations, but implicitly assuming quasi-isometry; adapting these methods is nontrivial.

3 Overview of CAD-Deform framework

Our approach is built on top of the framework from [5,6] for CAD model retrieval and 9 DoF alignment in 3D scans. By running any of the approaches from [5,6] for an input scan, we obtain initial object locations and 9 DoF alignments of CAD models potentially matching specific parts of the scan. Next, we apply our proposed mesh deformation procedure (see Section 4), resulting in a more accurate shape representation of the aligned objects:

1. We segment the CAD models into semantic 3D parts following the labelling from the PartNet dataset [31].
2. For each aligned object, we select points in the scan that are the nearest (within some fixed radius) to each vertex of the CAD model. We assign a label of the nearest part of the aligned CAD model to each such point.
3. As an input to the proposed mesh deformation procedure, we use the mesh model with semantic part labels and labelled segment of the 3D scene.
4. We deform the mesh by optimizing the energy depending on the relative positions of mesh vertices and labelled points of the scene, see Section 4.

4 Data-driven shape deformation

In this section, we describe our method for fitting a closest-match mesh from the shape dataset to the scanned data. Our algorithm assumes two inputs:

– an initial mesh $\mathbf{M}$ from the dataset, with part labels assigned to vertices and 9 degrees-of-freedom (9 DoF) transformation assigned to mesh;

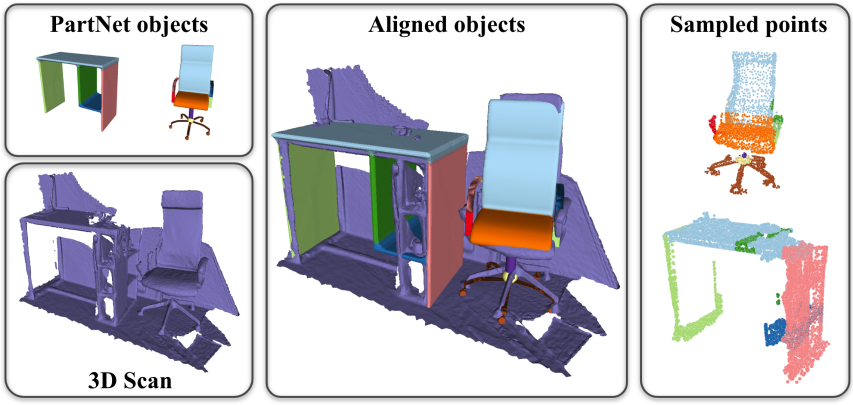

Fig. 2: Data acquisition for CAD-Deform: we project PartNet labels onto aligned ShapeNet CAD models (left), register these models to a 3D scene scan (middle), and extract points on the scene within ε-neighborhood of aligned mesh surface (we set $\varepsilon = 10\,\text{cm}$), copying labels corresponding to nearest part of CAD model (right).

– a subset of the scanned data, segmented with the same part labels as $\mathbf{M}$.

Fig. 2 shows an example of the input data, see also Section 3.

Notations. A mesh $\mathbf{M} = (\mathbf{V}^0, \mathbf{E}, \mathbf{F})$ consists of a set of initial 3D vertex coordinates $\mathbf{V}^0$ of size n_v, the edge set $\mathbf{E}$ of size n_e, and the triangle face set $\mathbf{F}$ of size n_f. We compute the deformed vertex positions $\mathbf{V}$ by minimizing a deformation energy. All vertices are assigned with part labels, which is a map $\mathbf{Q} : \mathbf{V}^0 \to \mathbf{C}$, where $\mathbf{C}$ is the set of labels $c_i, i = 1, \ldots, n_c$, and n_c is the number of parts in all objects in our dataset. For a mesh $\mathbf{M}_m$, $\mathbf{C}_m \subset \mathbf{C}$ is the set of labels of its parts. The set of $n_\mathbf{P}$ points $\mathbf{P}$ has the same labels as the mesh $\mathbf{M}_m$ we fit, *i.e.*, $\mathbf{C}_\mathbf{P} = \mathbf{C}_m$. In addition, we assume that for every mesh $\mathbf{M}_m$ on the scan we have a scaling transformation, represented by a 4×4 matrix T_m^0, that aligns it with voxelized points $\mathbf{P}$. In our optimization, we use *per-edge* transformations T_e, discussed below, which we use to measure the deviation from a scaled version of the original shape and deformation smoothness.

4.1 Deformation energy

Our goal is to define a deformation energy to match the scanned data as closely as possible, while maintaining continuity and deformation smoothness, and penalizing deviation from the original shape. In addition, we include a term that preserves perceptually important linear geometric features of the mesh.

Conceptually, we follow common mesh deformation methods such as ARAP [38] which estimate a local transformation from vertex positions and penalize the deviation of this transformation from the target (*e.g.*, closest rotation). An important distinction is that the transformation we need to estimate locally is a

3D, rather than 2D transformation, and cannot be estimated from the deformed positions of vertices of a single triangle.

Edge transformations. We associate local 3D affine transformations with mesh edges. Each transformation is defined in a standard way by a 4×4 matrix in projective coordinates, with the last row $(0, 0, 0, c)$, $c \neq 0$. The vertices of two faces (i_1, i_2, i_3) and (i_2, i_1, i_4), incident at an edge $e = (i_1, i_2)$, form a (possibly degenerate) tetrahedron. If it is not degenerate, then there is a unique linear transformation T_e mapping the undeformed positions $(v^0_{i_1}, v^0_{i_2}, v^0_{i_3}, v^0_{i_4})$ to $(v_{i_1}, v_{i_2}, v_{i_3}, v_{i_4})$. Matrix T^0 of the affine transformation has 12 coefficients that are uniquely determined by the equations $T_e v^0_i = v_i$, $i = 1 \dots 4$; moreover, these are linear functions of the deformed positions v_i, as these are only present on the right-hand side of the equations. Handling of degenerate tetrahedra is discussed in Section 4.2.

Energy. We define the following non-linear objective for the unknown deformed vertex positions $\mathbf{V}$ and (a fixed) point set $\mathbf{P}$:

$$\mathcal{E}(\mathbf{V}, \mathbf{P}) = \underbrace{\boxed{E_{\text{shape}} + \alpha_{\text{smooth}} E_{\text{smooth}} + \alpha_{\text{sharp}} E_{\text{sharp}}}}_{\text{quadratic problem}} + \alpha_{\text{data}} E_{\text{data}} + \alpha_{\text{Lap}} E_{\text{Lap}},$$

$$E_{\text{shape}} = \underbrace{\sum_{e \in \mathbf{E}} \|T_e(\mathbf{V}) - T^0_e\|_2^2}_{\text{deviation}}; \qquad E_{\text{smooth}} = \sum_{f \in \mathbf{F}} \sum_{e_i, e_j \in f} \|T_{e_i}(\mathbf{V}) - T_{e_j}(\mathbf{V})\|_2^2;$$

$$E_{\text{sharp}} = \sum_{k=1}^{n_p} \sum_{e_s \in \mathbf{E}^k_{\text{sharp}}} \|T_{e_s}(\mathbf{V}) - T_{e_{s+1}}(\mathbf{V})\|_2^2; \qquad E_{\text{data}} = f_{\text{data}}(\mathbf{V}, \mathbf{P}). \tag{1}$$

The first term penalizes deviations of the 3D affine transformations defined by the deformed vertex positions from the transformation (a non-uniform scale) that aligns mesh with the data. This term directly attempts to preserve the shape of the object, modulo rescaling.

As explained above, the transformations $T_e(\mathbf{V})$ are defined for non-degenerate input mesh configurations. Suppose the four initial vertex positions $(v^0_{i_1}, v^0_{i_2}, v^0_{i_3}, v^0_{i_4})$ form a degenerate tetrahedron $\mathbf{W}$, *i.e.*, two faces incident at the edge are close to co-planar. In this case, we use an energy term consisting of two terms defined per triangle. Instead of 4×4 matrix for each non-degenerate tetrahedron, there is a 3×2 matrix for each triangular face of degenerate tetrahedron that represents a transformation restricted to the plane of this face. Note that in this case, the deformation in the direction perpendicular to the common plane of the triangles does not affect the energy, as it does not have an impact on the local shape of the deformed surface. We explain the remaining energy terms in the next sections.

To bring second-order information about mesh surface in our energy formulation, we add the Laplacian smoothness term $E_{\text{Lap}}(\mathbf{V}, \mathbf{V}') = \sum_{i=1}^{|\mathbf{V}|} \|L(v_i) - L(v'_i)\|^2$, where $L(v_i) = \frac{1}{\mathcal{N}_i} \sum_{j=1}^{|\mathcal{N}(i)|} v_j$ and $\mathcal{N}(i)$ is a set of one-ring neighbors of vertex $v_i \in \mathbf{V}$ ($v'_i \in \mathbf{V}'$).

4.2 Quadratic terms

Smoothness term. This term can be thought of as a discrete edge-based Laplacian, applied to the transforms associated with edges: the difference of any pair of transforms for edges belonging to the same triangle is penalized.

Sharp features term. We have observed that a simple way to preserve some of the perceptually critical geometric aspects of the input meshes is to penalize the change of deformations T_e along sharp geometric features, effectively preserving their overall shape (although still allowing, possibly non-uniform, scaling, rotations and translations). We detect sharp edges based on a simple dihedral angle thresholding, as this is adequate for the classes of CAD models we use. Detected sharp edges are concatenated in sequences, each consisting of vertices with exactly two incident sharp edges each, except the first and the last. Those sequences are defined for each part of the mesh. Sequences of different parts has no common sharp edges or vertices belonging to them. Effectively, the sharpness term increases the weights for some edges in the smoothness term.

4.3 Data term

We use two approaches to defining the data term, one based on screened attraction between close points in the mesh $\mathbf{M}_m$ and $\mathbf{P}$, and the other one based on attraction between *a priori* chosen corresponding points of the mesh $\mathbf{M}_m$ and data points in the set $\mathbf{P}$. We found that the former method works better globally, when the deformed mesh is still far from the target point cloud, while the latter is able to achieve a better match once a closer approximation is obtained.

Part-to-part mapping. We define a data-fitting term based on point proximity: we set an energy proportional to the distance between sufficiently close points. To avoid clustering of mesh points, we add a screening term that disables attraction for mesh vertices close to a given point.

We denote by $H(x)$ the Heaviside function, *i.e.*, a function with value 1 if $x \geqslant 0$ and zero otherwise. We set

$$f^{\mathrm{p2p}}_{\mathrm{data}}(\mathbf{V}, \mathbf{P}) = \sum_{c \in \mathbf{C}} \sum_{v \in \mathbf{V}_c} \sum_{p \in \mathbf{P}_c} \xi^{\sigma}(p, \mathbf{V}_c) \big(d^{\varepsilon}(v - p)\big)^2, \tag{2}$$

where $d^{\varepsilon}(v-p) = (v-p) \cdot H(\varepsilon - \|v-p\|_2^2)$, and $\xi^{\sigma}(p, \mathbf{V}) = H(\min_{\{v \in \mathbf{V}\}} \|v-p\| - \sigma)$ is a "screening" function. The value for σ is chosen to be the mean edge length, the value of ε is about 10 edge lengths. To make $f^{\mathrm{p2p}}_{\mathrm{data}}$ differentiable, instead of the Heaviside function and min, we can use their smoothed approximations.

Nearest-neighbor mapping. Recall that the vertices $\{v_i\}$ of the mesh m and the points $\{p_i\}$ of the set $\mathbf{P}$ have the same part label sets $\mathbf{C}_m = \mathbf{C}_{\mathbf{P}}$. For each label c, we consider $\mathbf{V}^0_c$ and $\mathbf{P}_c$, the set of mesh vertices in initial positions and the set of points with the same label c in $\mathbf{P}$. Let $B_{\mathbf{V}_c}, B_{\mathbf{P}_c}$ be the bounding boxes of these sets, and consider the affine transform T^c_B that maps $B_{\mathbf{V}_c}$ to $B_{\mathbf{P}_c}$. Among all possible correspondences between corners of the boxes, we choose the one that produces the affine transform closest to identity. Then the index

$i = \iota^T(p)$ of the vertex v_i, corresponding to a point $p \in \mathbf{P}_c$, is determined as $\iota^c(p) = \arg\min_{\{i, v_i \in \mathbf{V}_c\}} \|T_B^c v_i - p\|_2^2$. Then the data term is defined as

$$f_{\text{data}}^{\text{nn}}(\mathbf{V}, \mathbf{P}) = \sum_{c \in \mathbf{C}} \sum_{p \in \mathbf{P}_c} \|p - v_{\iota^c(p)}\|_2^2. \tag{3}$$

4.4 Optimization

The optimization of (1) is highly nonlinear due to the data term. However, all other terms are quadratic functions of vertex coordinates, so minimizing

$$\begin{aligned} E_{\text{quad}} &= E_{\text{shape}} + \alpha_{\text{smooth}} E_{\text{smooth}} + \alpha_{\text{sharp}} E_{\text{sharp}} \\ &= \overline{\mathbf{V}}^\top A_{\text{shape}} \overline{\mathbf{V}} + \alpha_{\text{smooth}} \overline{\mathbf{V}}^\top A_{\text{smooth}} \overline{\mathbf{V}} + \alpha_{\text{sharp}} \overline{\mathbf{V}}^\top A_{\text{sharp}} \overline{\mathbf{V}} + b^\top \overline{\mathbf{V}} \end{aligned}$$

is equivalent to solving a system of linear equations. Denoting the sum of the matrices in the equation by A_{quad}, we obtain the optimum by solving $A_{\text{quad}} \overline{\mathbf{V}} = 0$ where the vector $\overline{\mathbf{V}}$ is a flattening of the vector $\mathbf{V}$ of 3D vertex positions to a vector of length $3n_{\mathbf{V}}$.

The data term is highly nonlinear, but solving the complete optimization problem can be done efficiently using A_{quad}^{-1} as the preconditioner. For our problem, we use the preconditioned L-BFGS optimizer summarized in Algorithm 1.

Algorithm 1: Preconditioned L-BFGS mesh optimization (PL-BFGS)

$M_{\text{precond}} = A_{\text{quad}}^{-1}$ `// stored as LU decomposition`
$\overline{\mathbf{V}} = T_m^0(\overline{\mathbf{V}}^0)$
for $i \leftarrow 0$ **to** N_{iter} **do**
  $g_{\text{tot}} = \alpha_{\text{data}} \mathrm{d}E_{\text{data}}/\mathrm{d}p + A_{\text{quad}} \overline{\mathbf{V}} + b$
  $\overline{\mathbf{V}} = \text{L-BFGS-step}(\overline{\mathbf{V}}, g_{\text{tot}}, M_{\text{precond}})$
end

5 Datasets

Our method relies on a number of publicly available datasets to compute mesh annotations for our deformations and assess fitting performance. To assess fitting performance, we use Scan2CAD dataset [5] that consists of 14225 ground-truth 9 DoF transformations between objects in 1506 reconstructions of indoor scenes from Scannet [11] and 3049 unique CAD models from ShapeNet [9].

Our deformation framework requires high-quality watertight meshes to support numerically stable optimization, which does not hold for ShapeNet CAD models. Thus, we remesh these to around 10k–15k vertices using [24], obtaining more uniform discretizations. To annotate these remeshed CAD models with semantic part labels required by our deformation procedure, we register them with the corresponding part-labeled meshes from the PartNet dataset [31] by first choosing an appropriate 90° rotation around each axis and then optimizing

for a finer alignment using a point-to-point ICP algorithm [35] between vertices of the two meshes. We annotate the semantic parts for vertices in each mesh by projecting it from the respective closest vertices of the registered PartNet mesh.

The original ShapeNet meshes, however, are used to extract sharp geometric features, as these have easily detectable sharp angles between adjacent faces. We label as sharp all edges adjacent to faces with a dihedral angle smaller than the threshold $\alpha_{\text{sharp}} = 120°$. We further project vertex-wise sharpness labels from the original to the remeshed CAD models and select a sequence of edges forming the shortest paths between each pair of vertices as sharp.

6 Results

6.1 Evaluation setup

Our performance evaluation of obtained deformations is multifaceted and addresses the following quality-related questions:

- Scan fitting performance: *How well do CAD deformations fit?*
- Perceptual performance for deformations: *How CAD-like are deformations?*
- Contributions of individual energy terms: *Which energy terms are essential?*
- Deformation flexibility: *Can better shapes be achieved by approximating clean meshes rather than noisy scans?*

Fitting and perceptual metrics. We quantify the deformation performance in terms of fitting quality between the scene scans and 3D CAD models using a family of related measures computed on a per-instance basis. For vertices $\mathbf{V} = (v_i)$ of the deformed mesh $\mathbf{M}$, we compute distances to their respective nearest neighbors ($\text{NN}(v_i, \mathbf{S})$) in the scan $\mathbf{S}$. We compute per-instance Accuracy $= |\mathbf{V}_{\text{close}}|/|\mathbf{V}|$, reflecting the fraction of closely located vertices, and trimmed minimum matching distance $\text{tMMD} = \sum_{v_i \in \mathbf{V}} \min(\tau, \|v_i - \text{NN}(v_i, \mathbf{S})\|_1)/|\mathbf{V}|$, where $\mathbf{V}_{\text{close}} = \{v_i \in \mathbf{V} : \|v_i - \text{NN}(v_i, \mathbf{S})\|_1 < \tau\}$ is the set of vertices falling within L_1-distance τ to their nearest neighbor in the scan, and τ controls robustness w.r.t. incomplete scans. We set $\tau = 0.2$ (see supplementary) in our experiments and report Accuracy and tMMD values averaged over classes and instances.

There is no universally agreed perceptual quality measure for meshes; thus, we opted for a tripartite evaluation for our resulting deformations. First, we measure dihedral angle mesh error (DAME) [41], revealing differences in *local surface quality* between the original and the distorted meshes:

$$\text{DAME}(\mathbf{M}, \mathbf{M}_{\text{def}}) = \frac{1}{|\mathbf{E}|} \sum_{\text{adjacent} f_1, f_2} \left| D_{f_1,f_2} - \overline{D_{f_1,f_2}} \right| \cdot \exp\left\{ (Z_{\text{DAME}} D_{f_1,f_2})^2 \right\},$$

where D_{f_1,f_2} and $\overline{D_{f_1,f_2}}$ represent oriented dihedral angles between faces f_1 and f_2 in the original and deformed meshes, respectively, and $Z_{\text{DAME}} = \sqrt{\log(100/\pi)}/\pi$ is a parameter scaling DAME values to $[0, 100]$.

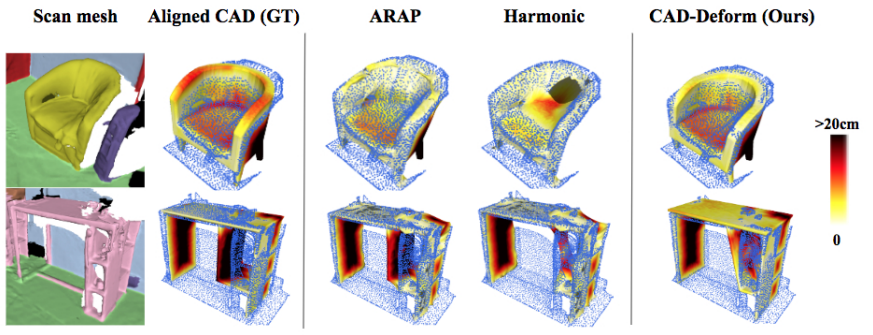

Fig. 3: Deformations obtained using our method and the baselines, with mesh colored according to the Euclidean distance to its nearest point in the scan. We note that high accuracy scores for Harmonic and ARAP deformations are achieved at the cost of destroying the initial structure of the mesh, particularly in regions where scan is missing (note that back side and armrests are gone for chairs in the first and second rows). In contrast, our method is better able to preserve smooth surfaces, sharp features, and overall mesh integrity, while keeping accurate local alignment.

Second, we assess *abnormality of deformed shapes* with respect to the distribution of the undeformed shapes, building on the idea of employing deep autoencoders for anomaly detection in structured high-dimensional data. By replicating the training instances, autoencoders learn features that minimize reconstruction error; for novel instances *similar* to those in the training set, reconstruction error is low compared to that of strong *outliers*. We train six autoencoders [2,17] for point clouds using vertices of undeformed meshes separately for the top six classes present in Scan2CAD annotation: *table, chair, display, trashbin, cabinet,* and *bookshelf*. Passing vertices $\mathbf{V}_{\text{def}}$ of a deformed shape to the respective autoencoder, one can assess how accurately deformed meshes can be approximated using features of undeformed meshes. This property can be evaluated with Earth Mover's Distance (EMD) $d_{\text{EMD}}(\mathbf{V}_{\text{def}}, \mathbf{V}'_{\text{def}}) = \min_{\phi:\mathbf{V}_{\text{def}}\to\mathbf{V}'_{\text{def}}} \sum_{v\in\mathbf{V}_{\text{def}}} \|v - \phi(v)\|_2$, where ϕ is a bijection, obtained as a solution to the optimal transportation problem involving $\mathbf{V}_{\text{def}}$ and $\mathbf{V}'_{\text{def}}$, that can intuitively be viewed as the least amount of work needed to transport $\mathbf{V}_{\text{def}}$ vertices to positions of $\mathbf{V}'_{\text{def}}$.

Lastly, we assess *real human perception* of deformations in a user study, detailed in Section 6.3.

Optimization details. To perform quantitative comparisons, we use 299 scenes in ScanNet constituting Scan2CAD validation set [5], but with 697 shapes present in PartNet dataset, amounting to 1410 object instances. Our full experimental pipeline is a sequence of deformation stages with different optimization parameters, and Hessian being recomputed before each stage. Specifically, we perform one *part-to-part* optimization with parameters $\alpha_{\text{shape}} = 1, \alpha_{\text{smooth}} = 0, \alpha_{\text{sharp}} = 0, \alpha_{\text{data}} = 5 \times 10^4$ for 100 iterations, then we perform 5 runs of *nearest-neighbor*

Method	Class avg. GT	S2C [5]	E2E [6]	Instance avg. GT	S2C [5]	E2E [6]
# TPs	1410	499	882	1410	499	882
TP undeformed	89.2	83.7	88.5	90.6	79.4	93.9
Ours: NN only	89.7	84.3	89.0	91.4	84.7	94.4
Ours: p2p only	90.3	88.3	89.4	91.6	90.3	94.9
Ours: w/o smooth	90.6	**90.0**	89.6	92.3	90.3	95.0
Ours: w/o sharp	90.3	86.9	**90.6**	92.3	89.4	**95.2**
CAD-Deform	**91.7**	89.8	90.3	**93.1**	**92.8**	94.6

Table 1: Comparative evaluation of our deformations to true positive (TP) alignments by non-deformable approaches in terms of Accuracy (%). Note that deformations improve performance for all considered alignment approaches.

Method	bookshelf	cabinet	chair	display	table	trashbin	other	class avg.	avg.
# instances	142	162	322	86	332	169	197	201.4	1410
Ground-truth	88.0	75.2	94.8	98.9	89.6	96.6	81.4	89.2	90.6
Ours	**90.5**	**82.2**	**95.4**	**99.1**	**91.0**	**98.6**	**84.8**	**91.7**	**93.1**

Table 2: Comparative evaluation of our approach to non-deformable ground-truth and baselines in terms of scan approximation Accuracy (%). We conclude that our deformations improve fitting accuracy across all object classes by 2.5 % on average.

deformation for 50 iterations with parameters $\alpha_{\text{shape}} = 1, \alpha_{\text{smooth}} = 10, \alpha_{\text{sharp}} = 10, \alpha_{\text{data}} = 10^3$. More details about optimization and timings are provided in the supplementary.

6.2 Fitting Accuracy: How well do CAD Deformations fit?

We first demonstrate how deformation affects scan fitting performance for meshes aligned using different methods, specifically, we use true-positive shape alignments computed using Scan2CAD (S2C) [5], End-to-End (E2E) [5], as well as ground-truth alignments. We start with an aligned mesh, copy the 9 DoF transformation to each of the mesh vertices, and optimize using our deformation method with parameters described in Section 6.1. We report Accuracy scores in terms of fraction of well-approximated points in the scan for aligned shapes pre- and post-optimization in Table 2, achieving improved performance across all considered alignment procedures.

Surprisingly, we improve even over ground-truth alignments by as much as 2.5 %. Thus, we compute per-class scores in Table 1 (comparison across alignments), reporting improvements of up to 7 % in average scan approximation accuracy that are consistent *across all object classes*. We visualize example deformations obtained using our approach and baselines in Figure 3.

We have discovered our deformation framework to be robust w.r.t. level of detail in the data term in (2) and provide more detail in the supplementary.

6.3 CAD Quality: How CAD-like are deformed models?

Having obtained a collection of deformed meshes, we aim to assess their visual quality in comparison to two baseline deformation methods: as-rigid-as-possible (ARAP) [38] and Harmonic deformation [7,26], using a set of perceptual quality measures. The details of our user study design and visual assessment are provided in the supplementary.

6.4 Ablation study

To evaluate the impact of individual energy terms in (1) on both scan fitting performance and perceptual quality, we exclude each term from the energy and compute deformations by optimizing the remaining ones. First, we exclude sharpness or smoothness terms, optimizing for deformations using the original two-stage method; second, to better understand the influence of each stage, we perform experiments with only the first or the second stage (a single run only). We aggregate results into Table 1 and display them visually in Figure 4, concluding that our CAD-Deform maintains the right balance between fit to the scan and perceptual quality of resulting deformations.

6.5 Shape morphing results

To demonstrate the ability of our mesh deformation framework to perform shape interpolation, we choose two different meshes in the same ShapeNet category and optimize our energy (1) to approximate one with the other, see Fig. 5.

7 Conclusion

In this work, we have presented CAD-Deform, a shape deformation-based scene reconstruction method leveraging CAD collections that is capable of improving over existing alignment methods. More specifically, we introduce a composite deformation energy formulation that achieves regularization from semantic part

	DAME		**EMD** $\times 10^{-3}$		**User**	**Accuracy**	
Method	cls.	inst.	cls.	inst.	**study**	cls.	inst.
No deformation	0	0	77	77	8.6	89.2	90.6
ARAP [38]	47.1	45.7	88	87	4.0	90.8	91.8
Harmonic [7,26]	65.1	65.2	104	102	2.6	**96.2**	**96.6**
CAD-Deform	**20.5**	**17.2**	**84**	**84**	**7.7**	91.7	93.1

Table 3: Quantitative evaluation of visual quality of deformations obtained using ARAP [38], Harmonic deformation [7,26], and our CAD-Deform, using a variety of local surface-based (DAME [41]), neural (EMD [2,17]), and human measures.

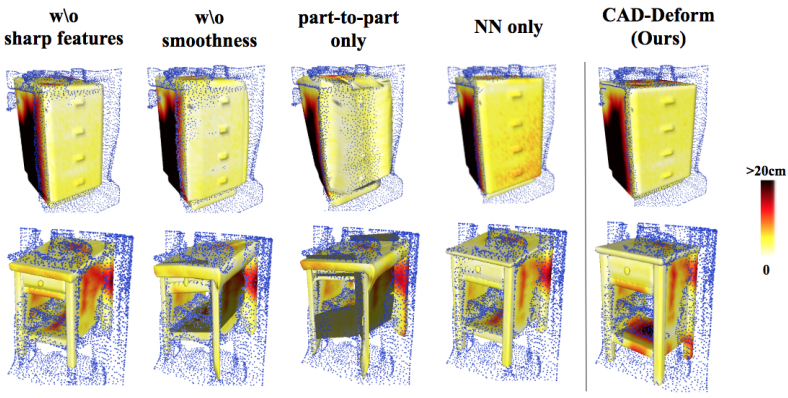

Fig. 4: Qualitative results of ablation study usind our deformation framework, with mesh coloured according to the value of the tMMD measure. Note that including smoothness term is crucial to prevent surface self-intersections, while keeping sharpness allows to ensure consistency in parallel planes and edges.

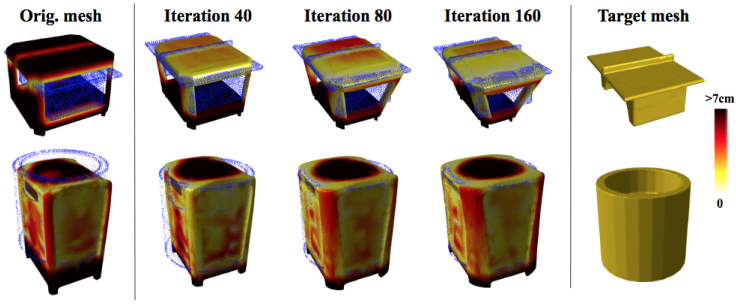

Fig. 5: Qualitative shape translation results, interpolating between the original mesh (left) and the target mesh (right).

structures, enforces smooth transformations, and preserves sharp geometric features. As a result we obtain significantly improved perceptual quality of final 3D CAD models compared to state-of-the-art deformation formulations, such as ARAP and polyharmonic deformation frameworks. Overall, we believe that our method is an important step towards obtaining lightweight digital replica from the real world that are both of high-quality and accurate fits at the same time.

Acknowledgements

The authors acknowledge the usage of the Skoltech CDISE HPC cluster Zhores for obtaining the results presented in this paper. The work was partially supported by the Russian Science Foundation under Grant 19-41-04109.

References

1. Achenbach, J., Zell, E., Botsch, M.: Accurate face reconstruction through anisotropic fitting and eye correction. In: VMV. pp. 1–8 (2015)
2. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International Conference on Machine Learning. pp. 40–49 (2018)
3. Amberg, B., Romdhani, S., Vetter, T.: Optimal step nonrigid icp algorithms for surface registration. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8 (2007)
4. Aubry, M., Maturana, D., Efros, A., Russell, B., Sivic, J.: Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In: CVPR (2014)
5. Avetisyan, A., Dahnert, M., Dai, A., Savva, M., Chang, A.X., Nießner, M.: Scan2cad: Learning cad model alignment in rgb-d scans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2614–2623 (2019)
6. Avetisyan, A., Dai, A., Nießner, M.: End-to-end cad model retrieval and 9dof alignment in 3d scans. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2551–2560 (2019)
7. Botsch, M., Kobbelt, L.: An intuitive framework for real-time freeform modeling. ACM Transactions on Graphics (TOG) **23**(3), 630–634 (2004)
8. Cagniart, C., Boyer, E., Ilic, S.: Iterative mesh deformation for dense surface tracking. In: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops. pp. 1465–1472. IEEE (2009)
9. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
10. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. p. 303–312. SIGGRAPH '96, Association for Computing Machinery, New York, NY, USA (1996). https://doi.org/10.1145/237170.237269, `https://doi.org/10.1145/237170.237269`
11. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5828–5839 (2017)
12. Dai, A., Nießner, M., Zollöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. ACM Transactions on Graphics 2017 (TOG) (2017)
13. Dai, A., Ruizhongtai Qi, C., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5868–5877 (2017)
14. Deng, H., Birdal, T., Ilic, S.: 3d local features for direct pairwise registration. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
15. Dey, T.K., Fu, B., Wang, H., Wang, L.: Automatic posing of a meshed human model using point clouds. Computers & Graphics **46**, 14–24 (2015)
16. Drost, B., Ilic, S.: 3d object detection and localization using multimodal point pair features. In: 3DIMPVT. pp. 9–16. IEEE Computer Society (2012)

17. Egiazarian, V., Ignatyev, S., Artemov, A., Voynov, O., Kravchenko, A., Zheng, Y., Velho, L., Burnaev, E.: Latent-space laplacian pyramids for adversarial representation learning with 3d point clouds (12 2019)
18. Firman, M., Mac Aodha, O., Julier, S., Brostow, G.J.: Structured prediction of unobserved voxels from a single depth image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5431–5440 (2016)
19. Fröhlich, S., Botsch, M.: Example-driven deformations based on discrete shells. Comput. Graph. Forum **30**, 2246–2257 (12 2011). https://doi.org/10.1111/j.1467-8659.2011.01974.x
20. Grinspun, E., Hirani, A.N., Desbrun, M., Schröder, P.: Discrete shells. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. p. 62–67. SCA '03, Eurographics Association, Goslar, DEU (2003)
21. Guo, R., Zou, C., Hoiem, D.: Predicting complete 3d models of indoor scenes. arXiv preprint arXiv:1504.02437 (2015)
22. Gupta, S., Arbeláez, P., Girshick, R., Malik, J.: Aligning 3d models to rgb-d images of cluttered scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4731–4740 (2015)
23. He, L., Schaefer, S.: Mesh denoising via l0 minimization. Proc. ACM SIGGRAPH pp. 64:1–64:8 (01 2013)
24. Huang, J., Su, H., Guibas, L.: Robust watertight manifold surface generation method for shapenet models. arXiv preprint arXiv:1802.01698 (2018)
25. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In: UIST '11 Proceedings of the 24th annual ACM symposium on User interface software and technology. pp. 559–568. ACM (2011)
26. Jacobson, A., Tosun, E., Sorkine, O., Zorin, D.: Mixed finite elements for variational surface modeling. In: Computer graphics forum. vol. 29, pp. 1565–1574. Wiley Online Library (2010)
27. Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., Panozzo, D.: Abc: A big cad model dataset for geometric deep learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
28. Li, Y., Dai, A., Guibas, L., Nießner, M.: Database-assisted object retrieval for real-time 3d reconstruction. Computer Graphics Forum **34**(2), 435–446 (2015)
29. Liao, M., Zhang, Q., Wang, H., Yang, R., Gong, M.: Modeling deformable objects from a single depth camera. pp. 167 – 174 (11 2009). https://doi.org/10.1109/ICCV.2009.5459161
30. Mattausch, O., Panozzo, D., Mura, C., Sorkine-Hornung, O., Pajarola, R.: Object detection and classification from large-scale cluttered indoor scans. In: Computer Graphics Forum. vol. 33, pp. 11–21. Wiley Online Library (2014)
31. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 909–918 (2019)
32. Newcombe, R.A., Izadi, S., Hilliges, O., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: IEEE ISMAR. IEEE (October 2011)
33. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3d reconstruction at scale using voxel hashing. ACM Transactions on Graphics (TOG) (2013)

34. Park, S.I., Lim, S.J.: Template-based reconstruction of surface mesh animation from point cloud animation. ETRI journal **36**(6), 1008–1015 (2014)
35. Rusinkiewicz, S., Levoy, M.: Efficient variants of the icp algorithm. In: Proceedings Third International Conference on 3-D Digital Imaging and Modeling. pp. 145–152. IEEE (2001)
36. Salas-Moreno, R.F., Newcombe, R.A., Strasdat, H., Kelly, P.H.J., Davison, A.J.: Slam++: Simultaneous localisation and mapping at the level of objects. In: CVPR. p. 1352–1359. IEEE Computer Society (2013)
37. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1746–1754 (2017)
38. Sorkine, O., Alexa, M.: As-rigid-as-possible surface modeling. In: Symposium on Geometry processing. vol. 4, pp. 109–116 (2007)
39. Stoll, C., Karni, Z., Rössl, C., Yamauchi, H., Seidel, H.P.: Template deformation for point cloud fitting. In: SPBG. pp. 27–35 (2006)
40. Sungjoon Choi, Zhou, Q., Koltun, V.: Robust reconstruction of indoor scenes. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5556–5565 (June 2015). https://doi.org/10.1109/CVPR.2015.7299195
41. Váša, L., Rus, J.: Dihedral angle mesh error: a fast perception correlated distortion measure for fixed connectivity triangle meshes. Computer Graphics Forum **31**(5), 1715–1724 (2012). https://doi.org/10.1111/j.1467-8659.2012.03176.x, `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2012.03176.x`
42. Whelan, T., Leutenegger, S., Salas-Moreno, R.F., Glocker, B., Davison, A.J.: ElasticFusion: Dense SLAM without a pose graph. In: Robotics: Science and Systems (RSS). Rome, Italy (July 2015)
43. Zhou, X., Karpur, A., Gan, C., Luo, L., Huang, Q.: Unsupervised domain adaptation for 3d keypoint estimation via view consistency. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. pp. 141–157. Springer International Publishing, Cham (2018)