Black-Box Test-Time Shape REFINEment for Single View 3D Reconstruction

Brandon Leung   Chih-Hui Ho   Nuno Vasconcelos
University of California, San Diego
{b7leung, chh279, nvasconcelos}@ucsd.edu

Abstract

Much recent progress has been made in reconstructing 3D object shape from an image, i.e. single view 3D reconstruction. However, due to the difficulty of collecting large datasets in the wild with 3D ground truth, it remains a significant challenge for methods to generalize across domain, viewpoint, and class. Current methods also tend to produce averaged "nearest-neighbor" memorized shapes instead of genuinely understanding the image, thus eliminating important details. To address this we propose REFINE, a post-processing mesh refinement step easily integratable into the pipeline of any black-box method in the literature. At test time, REFINE optimizes a network per mesh instance, to encourage consistency between the mesh and the given object view. This, with a novel combination of losses addressing degenerate solutions, reduces domain gap and restores details to achieve state of the art performance. A new hierarchical multiview, multidomain image dataset with 3D meshes called 3D-ODDS is also proposed as a uniquely challenging benchmark. We believe that the novel REFINE paradigm and 3D-ODDS are important steps towards truly robust, accurate 3D reconstructions.

1. Introduction

Single view reconstruction (SVR) aims to generate the 3D shape of an object from an image of it. SVR networks are usually learned from datasets with many views of different objects. While ideally these datasets should be large, composed of real images, cover many object classes with many views per object, and come with corresponding 3D ground truth, this is extremely difficult to achieve in practice. As a result most methods are trained on renders of synthetic 3D CAD models [34, 51, 57], or only applicable to a specific object class per network, such as birds [21, 28], beyond which they cannot generalize. The goal of learning universal SVR models, applicable to any object, remains a significant challenge. This is compounded by the difficulty of generalizing across domains. As illustrated in Figures 1 and 4, the application of an SVR network trained

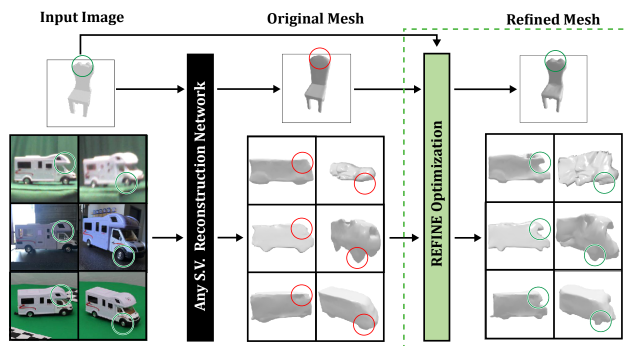

Figure 1. **bbTTSR:** Reconstructions by a SVR network trained on ShapeNet (center of the figure) are fed to the proposed *external* shape REFINEment network at test-time. **Evaluation:** Images from the training domain (top, from ShapeNet) are combined with a *new* dataset, composed of real images of objects in 3 domains & 8 viewpoints (bottom, from 3D-ODDS), to evaluate how bbTTSR enhances the accuracy and robustness of SVR. Gains for both accuracy and robustness are shown for REFINE (right).

on ShapeNet [3] to real images leads to severe reconstruction failures. Even with 3D synthetic data, current methods tend to recognize the object, perform a "nearest-neighbor" search for a "mean class shape" memorized during training [46], and make slight adjustments that are usually not enough to recover intricate shape details. As shown in Figure 2, while reconstructions (bottom row) reflect the category of the object in the image (top row), details that determine fine-grained identity are usually lost.

Test-Time Shape Refinement (TTSR) [39] is a promising solution to these problems. It poses the question of whether the SVR network reconstruction can be improved upon at test-time by providing some additional information about the object, e.g. a silhouette. TTSR has at least two interesting properties. First, because it is a test-time operation, it only requires relatively small datasets to design and evaluate. This enables the collection of datasets in the lab, to explicitly test how TTSR can enhance the robustness of reconstruction across many object classes and different image domains, while providing a dense coverage of object views. In this work, we leverage this observation and propose a new hierarchical multiview, multiclass, multido-

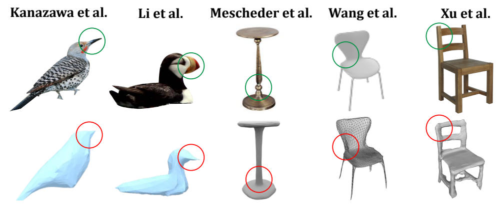

Figure 2. Important image details (circled in green) are frequently lost by state of the art 3D reconstruction methods (circled in red).

main dataset called the *3D Object Domain Dataset Suite (3D-ODDS)*, containing 71,496 real images of objects collected under many different controlled poses and domains, along with their scanned 3D meshes (Figures 1, 8). A second interesting property of TTSR is that it provides the opportunity to exploit optimization at test time, instead of just a forward pass, to improve reconstruction results. This was shown in [39] but posed as a fine-tuning problem, where parameters of their network are adjusted to achieve this goal.

In this work we ask the broader question of whether TTSR can be performed by an *external* network which refines the mesh shape produced by the SVR network, a posteriori as illustrated in Figure 1, and is applicable to any SVR method. We denote this as *black-box* TTSR (bbTTSR). There are several advantages in bbTTSR over TTSR. First, it is agnostic to the SVR architecture. As demonstrated in this work, it can be equally easily applied to approaches like DeepSDF [37] or OccNet [34] which use implicit functions, Mesh R-CNN [12] or Pix2Vox [57] which have voxel-based components, and AtlasNet [13] which represents meshes using atlas surface elements. Second, because it does not even require knowledge of the inner workings of the SVR network, it supports applications where the latter is provided by a third party and not publicly available. Finally, unlike network finetuning, bbTTSR encourages the joint development of networks and losses that explicitly address the degenerate solution tendencies and extreme data efficiency challenges seen in test-time refinement.

Given these potential advantages, we propose a *REFine INstances at Evaluation* (REFINE) architecture for bbTTSR. REFINE utilizes a mesh feature encoder with a graph refiner network, trained using a novel combination of loss functions encouraging both silhouette consistency and confidence-based mesh symmetry. We then combine existing datasets [3, 5, 43] with 3D-ODDS to produce an experimental framework to test how bbTTSR methods improve the effectiveness and robustness of SVR. These extensive experiments rigorously show that REFINE improves the reconstruction accuracy of many SVR networks as measured by several metrics, both in the presence and absence of domain gap between training and inference data, for both synthetic and real images, across diverse object classes/views.

Overall, this work makes four main contributions. The first is bbTTSR, i.e. using external post-processing networks at test time, to improve the quality of meshes produced by SVR methods. The second is the 3D-ODDS dataset. This is the first SVR dataset to deliberately target questions such as robustness of SVR to domain shift, using real world images of objects from many classes, and precise control of object pose. Third, we propose the first solution to the challenging bbTTSR problem with REFINE, which successfully suppresses degenerate solutions to provide performance gains. Finally, extensive experiments show that REFINE outperforms the state of the art in TTSR, is an effective solution for bbTTSR, enhancing performance of many SVR networks under many experimental conditions.

2. Related Work

Single View 3D Reconstruction. While many SVR methods have been proposed, they all suffer from the inconsistencies of Figure 2, and can benefit from REFINE. The main 3D output modalities are voxels, pointclouds, and meshes. Voxel methods typically encode an image into a latent vector, then decoded into a 3D voxel grid with upsampling 3D convolutions [5, 57]. Octrees can enable higher voxel resolution [45, 52]. Pointclouds have been explored as an alternative to voxels [8, 29] but usually require voxel or mesh conversion for use by downstream tasks. Among mesh methods, some learn to displace vertices on a sphere [22, 51] or a mean shape [21] to reconstruct. Current state of the art methods rely on an intermediate implicit function representation to describe shape [11, 34, 36, 37, 58], mapped into a mesh by marching cubes [33].

Methods also vary by their level of supervision. Most are fully supervised, requiring a large dataset of 3D shapes such as ShapeNet [3]. Recently, weakly-supervised methods have also been introduced, using semantic keypoints [21] or 2.5D sketches [53] as supervision. Alternatively, [28] has proposed a fully unsupervised method, combining part segmentation and differentiable rendering. Few-shot is considered in [35, 49] where classes have limited training data. Domain adaptation was explored in [38], which assumes access to data from a known target domain.

Despite progress in single view 3D reconstruction, questions arise on what is actually being learned. In particular, [46] shows that simple nearest-neighbor model retrieval can beat state of the art reconstruction methods. This raises concerns that current methods bypass genuine reconstruction, simply combining image recognition and shape memorization. Such memorization is consistent with Figures 1 and 2, leading to suboptimal reconstructions and inability to generalize across domains. It is likely a consequence of learning the reconstruction network over a training set of many instances from the same class. In contrast, REFINE uses test-time optimization to refine a single shape, encouraging consistency with a single silhouette. This prevents memorization, directly addressing the concerns of [46]. It also makes REFINE complementary to reconstruction methods and applicable as a postprocessing stage to any of them.

Test-Time Optimization. Test-time training [44] or optimization usually exploits inherent structure of the data in a self-supervised manner, as no ground truth labels are available. In [44], an auxiliary self-supervised rotation angle prediction task is leveraged to reduce domain shift in object classification. The same goal is achieved in [50] by test-time entropy minimization. Meanwhile, [47] uses self-supervision at test time to improve human motion capture. Additionally, interactive user feedback serves to dynamically optimize segmentation predictions [19, 40, 42].

Test-Time Shape Refinement. Test-time shape refinement (TTSR) requires a postprocessing procedure to improve the accuracy of meshes produced by a reconstruction network. Most previous approaches are white-box methods, i.e. they are specific to a particular model (or class of models) and require access to the internal workings of the model. Examples include methods that exploit temporal consistency in videos, akin to multi-view 3D reconstruction [27,30]. [27] requires the unsupervised part-based video reconstruction architecture proposed by the authors and [60] optimizes over a shape space specific to their architecture for zebra images. Among white-box methods, the approach closest to REFINE is that of [39], which finetunes the weights of the reconstruction network at test time, to better match the object silhouette. But even this method is specific to sign distance function (DeepSDF [37]) networks. By instead adopting the black-box bbTTSR paradigm, where the mesh refinement step is intentionally decoupled from the reconstruction process, REFINE is capable of learning *vertex-based deformations* for a mesh generated by *any reconstruction architecture*. Our experiments show that it can be effectively applied to improve the reconstruction performance of many networks and achieves state of the art results for test-time shape refinement, even outperforming [39] for DeepSDF networks. In summary, unlike prior approaches, REFINE is a black-box technique that can be universally applied to improve reconstruction accuracy, a posteriori.

3. Black-Box Test-Time Shape Refinement

3.1. Formulation and Inputs

Single view 3D reconstruction methods reconstruct a 3D object shape from a single image of the object. This is implemented with a mapping

$$S : \mathbb{R}^{W\times H\times 3} \rightarrow \mathcal{M} \in \mathcal{V} \times \mathcal{E}, \tag{1}$$

where an RGB image $x \in \mathbb{R}^{W\times H\times 3}$ of width W and height H is mapped to a mesh $M = (V, E) = S(x)$ by a reconstruction network, where $V \in \mathcal{V} \subset \mathbb{R}^{N\times 3}$ is a set of vertices and $E \in \mathcal{E} \subset \mathbb{B}^{\binom{N}{2}}$ a set of edges. $\mathbb{B}$ is a boolean domain specifying mesh connectivity. $S(x)$ is usually a coarse shape estimate, whose details do not match the input image, as shown in Figure 2. Performance further degrades when x is sampled from an image distribution different from that used during training [38].

	REFINE	OccNet [34]	MeshSDF [37]	Pix2Mesh [51]	AtlasNet [13]
Params. (M)	0.9	12.7	13.2	18.8	20.3

Table 1. REFINE improves reconstruction by introducing only a small number of parameters, relative to popular networks.

In [39], it was explored whether or not the use of additional auxiliary test-time information could help mitigate these problems. Their approach involved optimizing the parameters of S on-the-fly during inference, given a coarsely reconstructed mesh $S(x) = M_c = (V_c, E_c) \in \mathcal{M}$, an object silhouette x_s, and the camera pose p. We call this problem setting *test-time shape refinement* (TTSR), and we propose to investigate an alternative class of *black-box TTSR* (bbTTSR) solutions which abstracts shape refinement from any black-box reconstruction network S. This consists of introducing a dedicated refinement network R, external to S, to implement the shape refinement. R is trained at test-time, so that the 3D mesh $R \circ S(x)$ more accurately approximates the object shape, as illustrated in Figure 1. We denote the approach as REFINE and R as the REFINEment network. In this formulation, R predicts a set of 3D displacements $V_{dis} \in \mathbb{R}^{N\times 3}$ for the vertices in V_c. These are used to compute the REFINEd mesh $M_r = (V_r, E_r) = (V_c + V_{dis}, E_c)$ whose render best matches the silhouette x_s. Displacements are complemented by a set of symmetry confidence scores $V_{sConf} \in [0, 1]^{N\times 1}$, which regularizes V_{dis} through a symmetry prior, as detailed in Section 3.3.

Several advantages derive from bbTTSR's abstraction of refinement from reconstruction. First, REFINE is a black-box technique, applicable to any network S. In fact, the network does not even have to be available, only the mesh $S(x)$, which gives REFINE a great deal of flexibility. For example, while MeshSDF can only be used with DeepSDF networks, REFINE is applicable even to voxel and point-cloud reconstruction methods, by using mesh conversions [1, 2, 23, 24]. This property is important, as different methods are better suited for different downstream applications. For example, implicit methods [34, 37] tend to produce the best reconstructions but can have slow inference [37]. Meanwhile, AtlasNet [13] is less accurate but much more efficient, and inherently provides a parametric patch representation useful for downstream applications like shape correspondence. Second, because the refinement network R and loss functions used to train it are independent of the reconstruction network S, they can be specialized to the test-time shape refinement goal. This is important because the regularization required to avoid degenerate solutions for the learning of R, which is based on a single mesh instance, is quite different from that of S, which is learned from a large

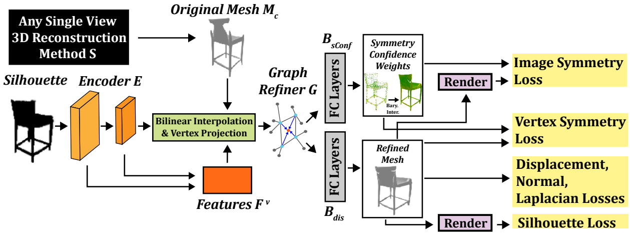

Figure 3. Given an original mesh reconstruction with missing details, a feature map encoder, graph convolutions, and fully connected layers are used to output vertex REFINEments needed to make the mesh consistent with an input image. Several losses, including confidence based 3D symmetry, prevent degenerate solutions. Optimization performed over single examples, at test time.

dataset. In REFINE, several loss functions tailored for test-time training are proposed to achieve this. We also design R to be much smaller than S, to lessen the additional computational overhead for refinement. As shown in Table 1, the REFINE network is at least 10 times smaller than most currently popular reconstruction networks.

3.2. Architecture

Figure 3 summarizes the REFINE architecture. This is a combination of an encoder E and a graph refiner G followed by 2 branches B_{dis} and B_{sConf}, which predict the vertex displacements and vertex confidence scores respectively. The encoder module E contains L neural network layers of parameters $\{\theta_i\}_{i=1}^{L}$, takes silhouette x_s as input, and outputs a set of L feature maps $F(x_s; \Theta_l = \{\theta_j\}_{j=1}^{l}) \in \mathcal{R}^{W_l \times H_l \times C_l}$, of width W_l, height H_l and C_l channels. In our implementation, E is based on ResNet [15]; L is set to 2, where $C_1 = 64$ and $C_2 = 128$.

Given feature map $F(x_s; \Theta_l)$, the feature vector f_l^v corresponding to a vertex v in M_c is computed by projecting the vertex position onto the feature map [12, 51],

$$f_l^v = Proj(v; F(x_s; \Theta_l), p) \in \mathcal{R}^{C_l}, \tag{2}$$

where p is the camera viewpoint and $Proj$ a perspective projection with bilinear interpolation. Vertices are represented at different resolutions, by concatenating the feature vectors of different layers into $F^v = (f_1^v, \dots, f_L^v)^T$. The set $\{F^v\}_{v=1}^{N}$ of concatenated feature vectors extracted from all vertices is then processed by a graph convolution [25] refiner G, of parameters ϕ, to produce an improved set of feature vectors $\{H^v\}_{v=1}^{N} = G(\{F^v\}_{v=1}^{N}; \phi)$. Finally, this set is mapped into the displacement vector V_{dis}

$$V_{dis} = B_{dis}(\{H^v\}_{v=1}^{N}; \psi_{dis}), \tag{3}$$

by a fully connected branch B_{dis} of parameters ψ_{dis} and into the confidence vector

$$V_{sConf} = B_{sConf}(\{H^v\}_{v=1}^{N}; \phi); \psi_{sConf}) \tag{4}$$

by a fully connected branch B_{sConf} of parameters ψ_{sConf}. Overall, the REFINE network implements the mapping

$$R(x_s, M_c; \{\theta_i\}, \phi, \psi_{dis}, \psi_{sConf}, p) = \{V_{dis}, V_{sConf}\}. \tag{5}$$

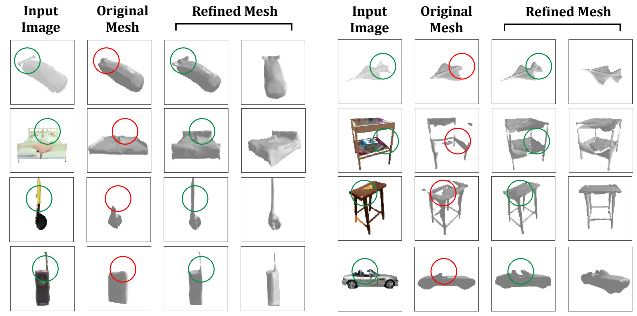

Figure 4. REFINEd reconstructions from an OccNet trained on ShapeNet. Results for objects from ShapeNet (top row), Pix3D (middle rows), & 3D-ODDS (bottom row). REFINE can correct small details as well as generate entirely new parts.

3.3. Optimization

The REFINE optimization combines popular reconstruction losses with novel losses tailored for test-time shape refinement. In what follows we use M^p to denote a differentiable renderer [22, 31] that maps mesh $M \in \mathcal{M}$ into its image captured by a camera of parameters p. We also define sets V_r^s, V_{dis}^s, and V_{sConf}^s of size N, constructed with the rows of V_r, V_{dis}, and V_{sConf} respectively. A set of popular reconstruction losses are used in REFINE, as follows.

Silhouette Loss: Penalizes shape and silhouette mismatch

$$L_{Sil} = L_{BCE}(x_s, \gamma(M_r^p)), \tag{6}$$

where $\gamma(M_r^p)$ is the silhouette of the rendered shape, using the 2D binary cross entropy loss

$$L_{BCE}(a, b) = \sum_{ij} a_{ij} \log(b_{ij}) + (1 - a_{ij}) \log(1 - b_{ij}). \tag{7}$$

Displacement Loss: Discourages overly large vertex deformations, with

$$L_{Dis} = \sum_{v_i \in V_{dis}^s} ||v_i||_2^2. \tag{8}$$

Normal Consistency & Laplacian Losses: L_{Nc} and L_{Lp} are widely used [7, 51] and encourages mesh smoothness.

A second set of losses is introduced to avoid degenerate solutions, namely overfitting to the input view during bbTTSR. These leverage the structural prior that many real world objects are bilaterally symmetric about a reflection plane $\mathcal{Z}$. Symmetry has long been exploited in computer vision, graphics, and geometry [32]. Many methods (e.g. [34, 51]) learn symmetry implicitly from the training data. Since datasets like Shapenet [3] are composed primarily of symmetric objects, a learned bias towards symmetry is almost impossible to avoid. Symmetry can also be explicit, e.g. [59] predicts planes of symmetry given 2D images to improve monocular depth estimation, or used to regularize learning, e.g. with horizontal flips during training [54].

Rather than 2D images, we exploit 3D shape symmetry by imposing two test-time constraints on reconstructed 3D meshes: 1) object vertices should be symmetric, and 2) mesh rendered images should reflect this symmetry.

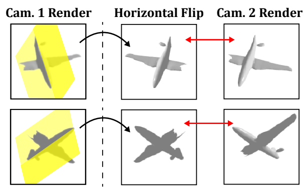

Figure 5. To enforce symmetry, a mesh is differentiably rendered by two cameras; the viewpoint of camera 2 is obtained by reflecting that of camera 1 about the mesh's plane of symmetry (yellow). The second render is compared to the horizontal flip of the first.

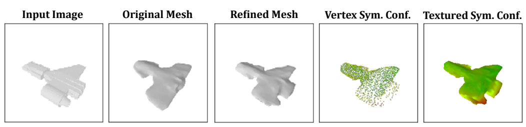

Figure 6. Left to right: image, original mesh, REFINEd mesh, and vertex confidence weights (shown as points or colors on the REFINEd mesh). Green shades indicate higher confidence; red lower, relaxing the symmetry prior on asymmetric object parts.

These leverage horizontal flips, vertex symmetry, and camera symmetry. Reflections are implemented with transformation $T = I - 2\vec{n}\vec{n}^\intercal$, where $\vec{n} \in \mathbb{R}^3$ is the unit normal vector of the reflection plane $\mathcal{Z}$. While there are methods to predict planes of object symmetry [10, 59] we have found that most reconstruction networks output aligned meshes, where $\vec{n} = [0, 0, 1]^\intercal$. To prevent the symmetry prior from overwhelming (6) if some asymmetry is present, confidence scores σ_i are learned during the REFINE optimization per vertex v_i. This enables local deviations from 3D symmetry when appropriate. The symmetry losses are as follows.

Vertex-Based Symmetry Loss: Encourages symmetric mesh vertices according to

$$L_{Vsym} = \frac{1}{N}\sum_{i=1}^{N} \sigma_i \min_{v_j \in V_r^s} \|Tv_i - v_j\|_2^2 + \lambda_{SymB} \ln\left(\frac{1}{\sigma_i}\right), \tag{9}$$

where $v_i \in V_r^s$ are the mesh vertices and $\sigma_i \in V_{sConf}^s$ the associated symmetry confidence scores. The first term penalizes distances between each vertex and its nearest neighbor upon reflection about $\mathcal{Z}$. This is weighted by the confidence score σ_i, which is low for vertices that should be asymmetric based on the object silhouette. The second term penalizes small confidence scores, preventing trivial solutions. The trade-off between the two terms is controlled by hyperparameter $\lambda_{SymB} \in [0, \infty)$. As shown in Figure 6, scores σ_i are large except in areas of clear asymmetry.

Render-Based Image Symmetry Loss: Encourages image projections that reflect object symmetry. Given m camera viewpoints $P_{Isym} = \{p_1, ..., p_m\}$, T is used to obtain differentiably rendered pairs from symmetric camera viewpoints $\{(M_r^{p_1}, M_r^{Tp_1}), ..., (M_r^{p_m}, M_r^{Tp_m})\}$, as shown in the rows of Figure 5. The loss is defined as

$$\begin{aligned} L_{Isym} = \frac{1}{m}\sum_{i=1}^{m}\sum_{j,k}\Big[&\sigma_{j,k}||\gamma(h(M_r^{p_i}))_{j,k} - \gamma(M_r^{Tp_i})_{j,k}||_2^2 \\ &+ \lambda_{SymB}\ln\left(\frac{1}{\sigma_{j,k}}\right)\Big], \end{aligned} \tag{10}$$

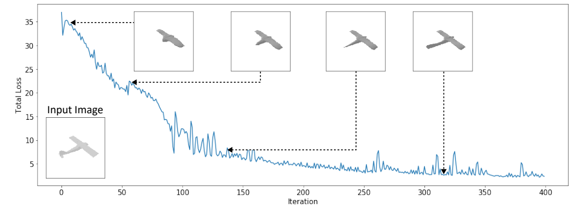

Figure 7. Mesh shape improves as REFINEment optimizes.

where $h(\cdot)$ is an horizontal image flip and j, k are image coordinates. Symmetry is enforced by minimizing the distance between the horizontal flip of each render $M_r^{p_i}$ and the render $M_r^{Tp_i}$ at the symmetrical camera viewpoint. This is akin to comparing a "virtual image" of what the mesh should symmetrically look like. Pixel-based confidence scores $\sigma_{j,k}$ are used as in (9). However, they are not relearned, but derived from the vertex confidences $\sigma_i, i \in \mathcal{V}_{j,k}$, of (9) by barycentric interpolation on the mesh faces, where $\mathcal{V}_{j,k}$ are mesh face vertices projected into pixel j, k.

Overall Loss: REFINE is trained with a weighed combination of the six losses

$$\begin{aligned} L_{total} = &\lambda_{Sil}L_{Sil} + \lambda_{Isym}L_{Isym} + \lambda_{Vsym}L_{Vsym} \\ &+ \lambda_{Dis}L_{Dis} + \lambda_{Nc}L_{Nc} + \lambda_{Lp}L_{Lp}. \end{aligned} \tag{11}$$

L_{Sil} is the main driving factor to ensure input silhouette consistency, while other losses serve as regularizers to prevent degenerate solutions. Figure 7 shows that REFINEd shape quality tracks the evolution of this loss, for an airplane whose body has been truncated in the original reconstruction. As the REFINE loss steadily decreases, the mesh progressively becomes more faithful to the input image; this is seen in the elongated body and corrected wing shape.

3.4. Implementation Details

Several details of our implementation are worth noting. In all experiments we used P_{Isym} of 6 viewpoints, with azimuths in $\{15°, 45°, 75°\}$ and elevations in $\{-45°, 45°\}$. The learning rate is 0.00007, $\lambda_{Sil} = 10$, $\lambda_{Isym} = 80$, $\lambda_{Vsym} = 20$, $\lambda_{SymB} = 0.0005$ $\lambda_{Dis} = 100$, $\lambda_{Nc} = 10$, and $\lambda_{Lp} = 10$. Also, REFINE supports a variable number of vertices per mesh, generally converges in 400 iterations, and takes only seconds to complete when performed in parallel. More details are given in the supplementary.

4. Multiview, Multidomain 2D & 3D Dataset

SVR is usually benchmarked on synthetic CAD datasets [3, 55] because these, albeit unrealistic, allow renders of images from many viewpoints. While real data can also be collected [4, 18, 26, 41], this has various difficulties resulting in datasets with different limitations. For example, Pascal3D [56] contains diverse real indoor/outdoor settings, but meshes are only approximations manually chosen from a CAD library. Pix3D [43] includes ground truth meshes but is relatively small and primarily depicts furniture in indoor locations with uncontrolled viewpoints. No existing real-world dataset enables systematic study of reconstruc-

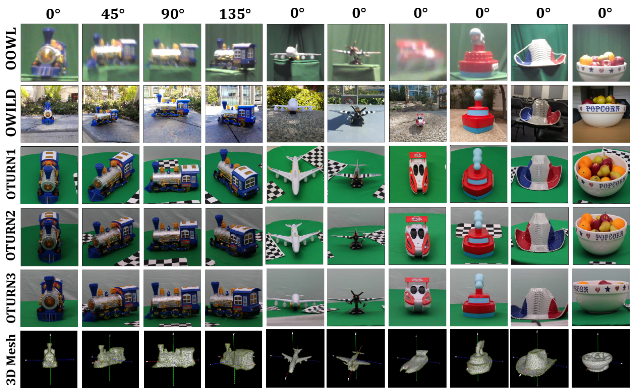

Figure 8. The proposed 3D-ODDS dataset contains 3D meshes and images from 3 domains, 8 azimuth angles, and 16 classes.

tion across *both* controlled viewpoints and domains.

In this work, we introduce the 3D-ODDS dataset to address these two fundamental challenges towards generalizable vision. 3D-ODDS contains DSLR-captured images of 331 objects from 16 different classes with dense pose coverage (72 azimuths, 3 elevations) for 216 images per object, and 71,496 images total. These images were used to generate 3D meshes for each object (331 meshes total, details in supplementary). Crucially, 232 of the objects can also be found in two real-world, multiview image datasets: OOWL [16] and OWILD [17]. They depict the same objects with 45° azimuth increments in different domains. OOWL images were collected using drone cameras during flight, OWILD in diverse indoor/outdoor locations with smartphones.

Note that the relatively small dataset size reflects the difficulty of real-world 3D data collection. While insufficient for large scale SVR network training, 3D-ODDS is ideally suited for tasks such as TTSR, domain adaptation, or few-shot learning, needed to translate shape reconstruction research into real applications. Using synthetic CAD datasets alone is also inadequate in achieving this goal. As illustrated in Figure 8, 3D-ODDS combines OTURN (our collected turntable images and 3D meshes) with OOWL and OWILD images to create a uniquely challenging hierarchical dataset of real images with 3 disentangled factors of variation: pose, object class, and domain. This results in the first real-world dataset to provide both 3D meshes of objects and their images under controlled viewpoints and domains. We believe that 3D-ODDS (to be released publicly) will be an important testing ground to evaluate the real world robustness of SVR methods.

5. Experiments

5.1. Experimental Setup

Metrics: To evaluate bbTTSR performance, the original mesh is first reconstructed by a baseline SVR method, and the reconstruction accuracy is measured. REFINE is then applied to the mesh and its accuracy is measured again. Several metrics of 3D accuracy [46] are used: EMD, $l2$ Chamfer Distance, F-Score, and 3D Volumetric IoU. Lower is better for EMD and Chamfer, while higher is better for IoU and F-Score; for details please refer to the supplementary.

Configuration	EMD↓	CD-l_2 ↓	F-Score↑	Vol. IoU↑	2D IoU
OccNet [34]	4.3	34.0	80	33	69
L_{Sil}	12.2	154.8	51	16	87
$L_{Sil,Dis,Nc,Lp}$	3.7	26.2	80	31	85
$L_{Sil,Dis,Nc,Lp,Vsym}$	3.7	25.8	81	32	86
$\boldsymbol{L_{total}}$	**3.3**	**22.5**	**84**	**35**	85
E & G removed, L_{total}	3.5	24.5	82	33	87
E removed, L_{total}	3.4	24.1	82	34	87
E rand. init, L_{total}	3.4	23.1	83	35	85
OccNet* [34]	11.0	123.3	48	10	53
$L_{total}, \lambda_{SymB} = 1.0*$	8.9	89.1	52	10	72
$\boldsymbol{L_{total}*}$	**7.8**	**85.9**	**55**	**12**	76

Table 2. Ablation study of REFINE. L_{total} indicates that all losses are used; an asterisk indicates results averaged over ShapeNetAsym instead of RerenderedShapeNet.

Datasets: Five datasets are considered, to carefully study domain shift. All baseline models are trained on the synthetic *ShapeNet* dataset [3], with images rendered by [5] using Blender [6]. We also re-rendered the 3D models in the test set of [5] using Pytorch3D [20]. This second dataset, called *RerenderedShapeNet* is designed to create a domain gap due to significant differences in shading, viewpoint, and lighting. The third dataset is motivated by our observation that about 97% of ShapeNet is symmetrical, in the sense that each mesh has a symmetry loss $L_{Isym} < 0.01$ for $\lambda_{SymB} = 1$ and $\sigma_{j,k} = 1$. To ablate how asymmetry affects reconstruction quality, we introduce a subset of RerenderedShapeNet, denoted as *ShapeNetAsym*, containing 1259 asymmetric meshes. Fourth, we use the *Pix3D* dataset [43], which contains real images and their ground truth meshes, to test for large domain shifts. Finally, we use *3D-ODDS* to study invariance to pose and image domain. For bbTTSR experiments, we use a subset of 3D-ODDS consisting of objects with high quality 3D mesh scans and images of 45° increment azimuth angles found in OOWL, OWILD, and OTURN's middle elevation. In total, this subset consists of 212 objects, 3 domains, and 8 viewpoints, for a total of $212 * 3 * 8 = 5088$ images and 212 meshes.

5.2. Ablation Studies

Ablations were performed for different components of REFINE. Here we also measure consistency between the reconstructed mesh's render and the input image silhouette, using 2D IoU, to better understand REFINE's behavior.

The top section of Table 2 shows the effect of different REFINEments of RerenderedShapeNet meshes originally reconstructed by OccNet. The first row is not refined. The second row shows that, using the silhouette loss only ($\lambda_{Sil} = 10$, all other $\lambda = 0$) improves input image consistency (from 69 to 87 2D IoU), but the refined mesh severely overfits to the input viewpoint, leading to decreased 3D accuracy. Adding the popular regularizers (third row, $\lambda_{Dis} = 100$, $\lambda_{Nc} = 10$, $\lambda_{Lp} = 10$) improves 3D reconstruction, but the gains over the baseline are small. The fourth row shows that enforcing vertex

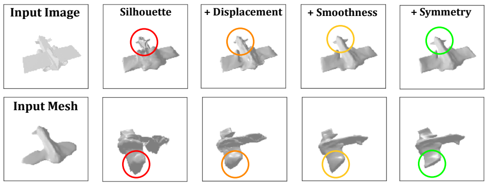

Figure 9. Leftmost column: input image and mesh. Other columns: REFINEment improves with an increasingly larger set of losses (left to right). Best viewed enlarged.

		EMD ↓	CD-l_2 ↓	F-Score ↑	Vol. IoU ↑
SVR	AtlasNet [13]	8.0	13.0	89	30
	Mesh R-CNN [12]	4.2	10.3	90	52
	Pix2Mesh [51]	3.4	8.0	93	48
	DISN [58]	2.6	9.7	91	57
TTSR	MeshSDF [39]	3.0→2.5 (-0.5)	12.0→7.8 (-4.2)	91→95 (+4)	-
	REFINEd OccNet [34]	2.9→**2.3** (-0.6)	12.2→**7.5** (-4.7)	91→**96** (+5)	57→**59** (+2)

Table 3. Reconstruction accuracies with no domain shift. Top: single view reconstruction (SVR) networks. Bottom: test-time shape refinement (TTSR) methods. TTSR results presented by accuracy *before* → *after* refinement, with gain shown in parenthesis.

symmetry ($\lambda_{Vsym} = 20, \lambda_{SymB} = 0.0005$) has marginal improvements by itself. However, when combined with render-based image symmetry (row five, which further adds the image symmetry loss with $\lambda_{Isym} = 80$) it enables significant gains in all metrics (e.g. from 34 to 22.5 CD-l_2).

The middle section of the table uses all losses, ablating architectural components by removing both encoder E and graph refiner G (directly optimizing the mesh deformation with no network), removing only E, and randomly initializing E. These refinements improve on the original mesh, but underperform the implementation of REFINE using G and E with ImageNet weights (row 5). We hypothesize this is because E and G provide a useful high dimensional projection for mesh deformation, similar to the inductive bias from architectural parameterization studied in [14, 48].

The bottom three rows of Table 2 use ShapeNetAsym to study the effect of asymmetry on REFINE performance. The sixth row is not refined. The seventh row shows that when the confidence scores of (9) and (10) are removed (by setting $\lambda_{SymB} = 1$, in which case the confidence scores become approximately 1) the refinement of asymmetrical meshes is significantly less accurate than that of the default configuration (eighth row, $\lambda_{SymB} = 0.0005$). It can also be seen that, when confidence scores are used, the reconstruction quality is significantly superior to that of the original meshes. In summary, the proposed confidence mechanism enables effective REFINEment of non-symmetric objects.

Figure 9 illustrates the contribution of each loss. The leftmost column shows the input airplane image (top) and mesh (bottom). From the second column, we progressively add more losses. With only the silhouette loss, degenerate solutions occur, severely overfitting to the input viewpoint. The displacement loss helps regularize deformation magnitude; the smoothness losses reduce jagged artifacts; the symmetry losses correct shape details (e.g. airplane tail) by

	EMD ↓	CD-l_2 ↓	F-Score ↑	Vol. IoU ↑
REFINEd OccNet [34]	4.3 → **3.3** (**-1.0**)	34.0 → **22.5** (**-11.5**)	80 → **84** (**+4**)	33 → **35** (**+2**)
REFINEd Pix2Mesh [51]	4.8 → **3.5** (**-1.3**)	38.0 → **23.1** (**-14.9**)	67 → **78** (**+11**)	22 → **27** (**+5**)
REFINEd AtlasNet [13]	6.2 → **4.9** (**-1.3**)	62.5 → **32.9** (**-29.6**)	56 → **72** (**+16**)	8 → **13** (**+5**)
REFINEd Pix2Vox [57]	4.5 → **3.3** (**-1.2**)	37.3 → **21.8** (**-15.5**)	70 → **80** (**+10**)	27 → **34** (**+7**)

Table 4. REFINEment in the presence of mild domain shift, namely RerenderedShapeNet reconstructions by ShapeNet trained networks. Gains occur under all networks, classes, and metrics.

enforcing a symmetry prior. These operate intuitively and can be tweaked for target applications. For example, if only symmetric objects are considered λ_{SymB} can be increased.

5.3. bbTTSR Results

We next consider the robustness of REFINE postprocessing to different levels of domain gap. A first set of experiments was performed without domain gap, with reconstruction networks trained and tested on the ShapeNet renders of [5]. Table 3 compares different reconstruction networks and TTSR methods (full per-class results in supplementary). Since the weights used in the state of the art method of [39] are not publicly available, we instead REFINEd OccNet[1] [34]. The REFINE+OccNet combination beats the state of the art, despite a somewhat unfair comparison, since REFINE performs black-box TTSR and is applicable to any network while the MeshSDF refinement of [39] is specific to its network.

Several experiments were next conducted to evaluate the effectiveness of REFINEment in the presence of domain gap. Table 4 gives reconstruction accuracy for RerenderedShapeNet reconstructions, before and after REFINEment, of ShapeNet pretrained networks. Four representatives of different reconstruction strategies are considered: OccNet (implicit functions [34]), Pixel2Mesh (ellipsoid deformation [51]), AtlasNet (surface atlas elements [13]), and Pix2Vox (voxel outputs, converted to mesh [33, 57]). A larger table with per-class results is presented in the supplementary; REFINE provides gains for all classes. The pre-refinement results of Table 4 are generally worse than those of Table 3. While the methods perform well on the training domain, they struggle to generalize to out-of-distribution data. However, REFINEment significantly recovers much of the lost performance for all networks, for relatively little extra computational overhead. Gains are particularly large for the Chamfer distance (-11.5 for OccNet, -14.9 for Pixel2Mesh, and -29.6 for AtlasNet) and increase with network sensitivity to domain gap (e.g. largest for AtlasNet, which has the weakest performance).

We next considered real-world datasets, which have the largest domain gap and are more interesting for applications. Table 5 shows that on Pix3D, REFINE gains are qualitatively identical to those of Table 4. A comparison to

[1] OccNet and the unrefined version of MeshSDF are comparable (both implicit based) and have nearly identical performance prior to refinement.

		EMD ↓	CD-l_2 ↓	F-Score ↑	Vol. IoU ↑
MeshSDF [39]	Chair	11.9 → 9.8 (-2.1)	102.0 → 89.0 (-13.0)	-	-
REFINEd OccNet [34]	Chair	11.0 → **8.5**	110.7 → **74.5**	57 → **62**	18 → **20**
	Bed*	7.5 → **6.1**	70.1 → **47.9**	57 → **62**	22 → **23**
	Bookcase*	7.4 → **4.1**	72.0 → **38.5**	56 → **65**	9 → **12**
	Desk	7.6 → **6.7**	60.6 → **43.7**	71 → **72**	26 → **27**
	Misc*	10.2 → **5.4**	129.6 → **69.8**	46 → **55**	19 → **20**
	Sofa	3.2 → **3.1**	30.8 → **25.5**	75 → **76**	50 → **51**
	Table	6.5 → **5.6**	67.7 → **57.8**	60 → **62**	16 → **17**
	Tool*	10.8 → **8.6**	140.8 → **118.6**	51 → **60**	11 → **14**
	Wardrobe*	5.9 → **3.7**	49.9 → **29.3**	65 → **68**	54 → **55**
	Mean	7.9 → **5.8** (**-2.1**)	81.4 → **56.2** (**-25.2**)	59 → **65** (**+6**)	23 → **28** (**+5**)

Table 5. REFINEment gain for large domain shifts, namely Pix3D reconstructions by ShapeNet trained networks. REFINE achieves gains under all metrics and for all networks. REFINE is even able to improve on classes not seen during training (asterisked).

the TTSR method of [39] on "Chair" shapes (the only class considered in [39]) again shows that REFINE substantially improves on the state of the art. This occurs even though performance prior to refinement is actually worse for OccNet than MeshSDF (Chamfer Distance 110.7 vs 102).

Finally, we studied pose and domain invariance using the 3D-ODDS dataset. For simplicity, we focused on OccNet and the F-score metric (as EMD and CD are unbounded). For each object, we measured accuracy before and after REFINEment using its 24 images as input. Boxplots of example results are shown in Figure 10 (full version in supplementary). Averaged per-object mean accuracy before and after REFINE, over all objects, were 37.2 and 44.4 respectively, while averaged per-object standard deviation were 16.2 and 14.3. This indicates that REFINE improves both reconstruction accuracy and invariance. Figure 11 summarizes averaged performance across pose angle, domain, and object class. REFINE improves reconstruction in all cases.

These results provide insight on the limitations of current reconstruction networks. OOWL (noisiest due to drone camera shake) is the hardest domain on average, followed by OWILD and OTURN (least noisy). Viewpoints at 0 and 180 degrees are most challenging: it is generally more difficult to infer object shape directly from the front or back. Geometrically simple classes like bottles, cans, and bowls perform better than average, with some exceptions like remotes (simple but do not do well). REFINE is beneficial for both classes seen and not seen during training (the latter marked by asterisks). To quantify the relationship between the 3 factors (pose, domain, class) and REFINEd accuracy, we used a 3-way ANOVA [9], with a blocked design to account for object-specific variability. Details are given in the supplementary; all factors were found statistically significant and total variability was decomposed into 13% class, 2% pose, 1% domain, 19% object instance, and 17% from interaction effects between pose/class/domain.

Overall, Tables 3 4, 5, and Figures 10, 11 show bbTTSR with REFINE achieves state of the art reconstruction accuracies, consistently providing performance gains regardless of metric, original base reconstruction network, class, viewpoint, domain, or dataset. Furthermore, gains are consistent

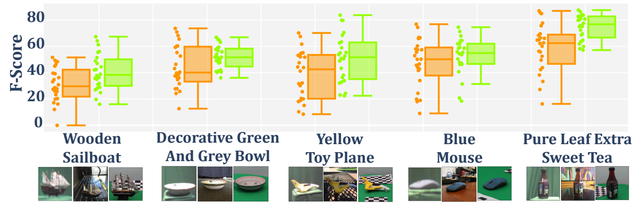

Figure 10. 3D-ODDS objects have 24 images (3 domains, 8 viewpoints). Reconstruction accuracies plotted before (after) REFINE as orange (green). REFINE improves performance & invariance.

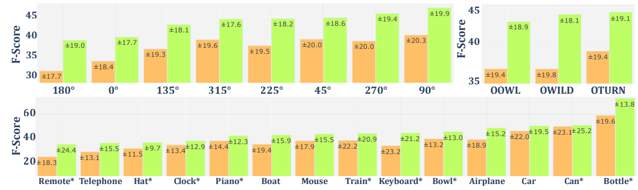

Figure 11. Performance & standard deviation on 3D-ODDS across viewpoint, domain, & class (asterisked unseen during training). Compared to original scores (orange), REFINEing (green) generally improves accuracy while decreasing variability.

or slightly better as domain gap widens; for the best performing OccNet, utilizing REFINE yields F-Score average improvements of 5, 4, 6, 5, 7, and 6 on ShapeNet, ReirenderedShapeNet, Pix3D, OTURN, OWILD, and OOWL. These improvements are illustrated in Figure 4. REFINE can both sharpen details (i.e. airplane's elongated nose) and create entirely new parts (set of wings in the back). It can also recover from very poor reconstructions due to significant domain shift, such as in the table and chair from Pix3D. It especially excels in unusual "outlier" shapes, such as the phone's antenna or convertible car from 3D-ODDS and is successful even for *classes on which the original reconstruction method was not trained*, leading to poor original meshes. This includes the spoon and bed in Figure 4; unseen classes marked by an asterisk in Table 5 and Figure 11. Additional examples provided in the supplementary.

6. Conclusion

In this paper, we demonstrated the effectiveness of blackbox test-time shape refinement (bbTTSR) for single view 3D reconstruction. The proposed REFINE method enforces regularized input image consistency, applicable to any reconstruction network in the literature. Experiments show systematic significant improvements over the state of the art, for many metrics, datasets, and reconstruction methods. A new hierarchical multiview, multidomain image dataset with 3D meshes, 3D-ODDS, was also proposed and shown to be a uniquely challenging benchmark for SVR.

Acknowledgments This work was partially funded by NSF award IIS-1924937, award IIS-2041009, and a gift from Amazon. It is supported by the National Science Foundation Graduate Research Fellowship. We also thank the use of the Nautilus platform for some of the experiments discussed above.

References

[1] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 3

[2] Fatih Calakli and Gabriel Taubin. Ssd: Smooth signed distance surface reconstruction. In *Computer Graphics Forum*, volume 30, pages 1993–2002. Wiley Online Library, 2011. 3

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 2, 4, 5, 6

[4] Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016. 5

[5] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 2, 6, 7

[6] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 6

[7] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 317–324, 1999. 4

[8] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 2

[9] Ronald Aylmer Fisher. Statistical methods for research workers. In *Breakthroughs in statistics*, pages 66–70. Springer, 1992. 8

[10] Lin Gao, Ling-Xiao Zhang, Hsien-Yu Meng, Yi-Hui Ren, Yu-Kun Lai, and Leif Kobbelt. Prs-net: Planar reflective symmetry detection net for 3d models. *IEEE Transactions on Visualization and Computer Graphics*, 27(6):3007–3018, 2020. 5

[11] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4857–4866, 2020. 2

[12] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9785–9795, 2019. 2, 4, 7

[13] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 2, 3, 7

[14] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2mesh: A self-prior for deformable meshes. *ACM Trans. Graph.*, 39(4), 2020. 7

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[16] Chih-Hui Ho, Brandon Leung, Erik Sandstrom, Yen Chang, and Nuno Vasconcelos. Catastrophic child's play: Easy to perform, hard to defend adversarial attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9229–9237, 2019. 6

[17] Chih-Hui Ho, Pedro Morgado, Amir Persekian, and Nuno Vasconcelos. Pies: Pose invariant embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12377–12386, 2019. 6

[18] Tomáš Hodaň, Pavel Haluza, Štěpán Obdržálek, Jiří Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017. 5

[19] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5306, 2019. 3

[20] Justin Johnson, Nikhila Ravi, Jeremy Reizenstein, David Novotny, Shubham Tulsiani, Christoph Lassner, and Steve Branson. Accelerating 3d deep learning with pytorch3d. In *SIGGRAPH Asia 2020 Courses*, pages 1–1. 2020. 6

[21] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018. 1, 2

[22] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018. 2, 4

[23] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 3

[24] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 3

[25] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017. 4

[26] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *2011 IEEE international conference on robotics and automation*, pages 1817–1824. IEEE, 2011. 5

[27] Xueting Li, Sifei Liu, Shalini De Mello, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Online adaptation for consistent mesh reconstruction in the wild. In *Advances in Neural Information Processing Systems*, 2020. 3

[28] Xueting Li, Sifei Liu, Kihwan Kim, Shalini De Mello, Varun Jampani, Ming-Hsuan Yang, and Jan Kautz. Self-supervised single-view 3d reconstruction via semantic consistency. In *European Conference on Computer Vision*, pages 677–693. Springer, 2020. 1, 2

[29] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 2

[30] Chen-Hsuan Lin, Oliver Wang, Bryan C Russell, Eli Shechtman, Vladimir G Kim, Matthew Fisher, and Simon Lucey. Photometric mesh optimization for video-aligned 3d object reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 969–978, 2019. 3

[31] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7708–7717, 2019. 4

[32] Yanxi Liu, Hagit Hel-Or, and Craig S Kaplan. *Computational symmetry in computer vision and computer graphics*. Now publishers Inc, 2010. 4

[33] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 2, 7

[34] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 1, 2, 3, 4, 6, 7, 8

[35] Mateusz Michalkiewicz, Sarah Parisot, Stavros Tsogkas, Mahsa Baktashmotlagh, Anders Eriksson, and Eugene Belilovsky. Few-shot single-view 3-d object reconstruction with compositional priors. In *European Conference on Computer Vision*, pages 614–630. Springer, 2020. 2

[36] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020. 2

[37] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2, 3

[38] Pedro O Pinheiro, Negar Rostamzadeh, and Sungjin Ahn. Domain-adaptive single-view 3d reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7638–7647, 2019. 2, 3

[39] Edoardo Remelli, Artem Lukoianov, Stephan Richter, Benoit Guillard, Timur Bagautdinov, Pierre Baque, and Pascal Fua. Meshsdf: Differentiable iso-surface extraction. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22468–22478. Curran Associates, Inc., 2020. 1, 2, 3, 7, 8

[40] Tomas Sakinis, Fausto Milletari, Holger Roth, Panagiotis Korfiatis, Petro Kostandy, Kenneth Philbrick, Zeynettin Akkus, Ziyue Xu, Daguang Xu, and Bradley J Erickson. Interactive segmentation of medical images through fully convolutional neural networks. *arXiv preprint arXiv:1903.08205*, 2019. 3

[41] Arjun Singh, James Sha, Karthik S Narayan, Tudor Achim, and Pieter Abbeel. Bigbird: A large-scale 3d database of object instances. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 509–516. IEEE, 2014. 5

[42] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking backpropagating refinement for interactive segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8623–8632, 2020. 3

[43] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018. 2, 5, 6

[44] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning (ICML)*, 2020. 3

[45] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096, 2017. 2

[46] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019. 1, 2, 6

[47] Hsiao-Yu Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems*, pages 5236–5246, 2017. 3

[48] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018. 7

[49] Bram Wallace and Bharath Hariharan. Few-shot generalization for single-image 3d reconstruction via priors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3818–3827, 2019. 2

[50] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. 3

[51] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 1, 2, 3, 4, 7

[52] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017. 2

[53] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017. 2

[54] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2020. 4

[55] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 5

[56] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2014. 5

[57] Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2690–2698, 2019. 1, 2, 7

[58] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems*, pages 492–502, 2019. 2, 7

[59] Yichao Zhou, Shichen Liu, and Yi Ma. NeRD: Neural 3d reflection symmetry detector. In *CVPR*, 2021. 4, 5

[60] Silvia Zuffi, Angjoo Kanazawa, Tanya Berger-Wolf, and Michael Black. Three-d safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5358–5367. IEEE, 2019. 3