Comparative Evaluation of 3D Reconstruction Methods for Object Pose Estimation (Supplementary material)

Varun Burde* Assia Benbihi* Pavel Burget Torsten Sattler
Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague
firstname.lastname@cvut.cz

The supplementary material is organized as follows: Sec. A describes the data acquisition and the information related to the release as announced in Sec.3 of the main paper. Sec. B provides implementation details relative to the reconstructions, the pose estimation, and the evaluation, as announced in Sec. 4 of the main paper. Sec.C reports further quantitative results that support the conclusions drawn in Sec. 5 of the main paper.

A. Dataset

The proposed benchmark answers the question "To what degree 3D models reconstructed by current 3D reconstruction algorithms can replace the CAD models commonly used for object pose estimation?" To this end, we compare the performance of two State-of-the-Art (SotA) object pose estimators, FoundationPose and Megapose [15, 29], when using the 3D reconstructed models against the use of classical CAD models. Thus, we mainly measure 3D reconstruction performance by the accuracy of the estimated poses rather than the accuracy of the resulting 3D models itself.

The evaluation of the pose estimators runs on the classic YCB-V [30] pose estimation dataset and follows the standard guidelines defined in the BOP benchmark [12, 13]. The YCB-V dataset [30] is made of 21 objects (Fig. 2) selected from the YCB dataset [5], including small objects, objects with low texture, complex shapes, high reflectance, and multiple symmetries. For each object, our benchmark consists of two components: **(1)** Images of the object captured by a robot arm that can be used for 3D reconstruction. **(2)** A set of test images for object pose estimation with ground truth poses registered with the image sets from (1). Compared to the data for the first component, which we captured ourselves, we use the test images provided by the YCB-V [30] dataset for the second component. Given the two components, we evaluate multiple state-of-the-art 3D reconstruction methods.

*Equal Contribution.

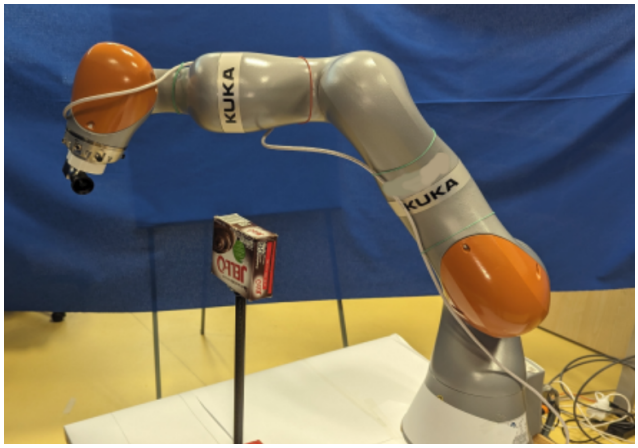

Figure 1. **Data Acquisition.** The objects are mounted on a tripod. A Basler Ace camera mounted on a 7DoF KUKA arm autonomously captures views from all sides of the YCB-V [5, 30] objects.

In this section, we describe how the images in **(1)** were collected (Sec. A.1) and how the data is released (Sec. A.2).

A.1. Data Collection Setup

Data Acquisition. The images are collected with a Basler Ace camera acA2440-20gc[1] mounted on the flange of a 7 Degrees-of-Freedom (DoF) KUKA LBR IIWA 14 R820 (Fig. 1). The image resolution prior to undistortion is 2448x2048 pixels and the field of view covers the whole object. The acquisition is done in an open-floor indoor environment exposed to neon lighting and varying sunlight through ceiling-high windows, which is typical of daily life and industrial scenarios. The camera exposure remains fixed during the object's scan.

The camera is calibrated using open-source solutions: the OpenCV routine for the camera calibration [4] and the

[1]https://docs.baslerweb.com/aca2440-20gc

MoveIt [6] routine for the hand-eye calibration[2]. Afterward, the camera can be positioned with an error of up to 5mm in translation and 0.1° in rotation. The resulting camera calibration is used to undistort the images using the open-source COLMAP [24] library. The camera poses obtained from the kinematic chain of the robot are later refined with an offline optimization (see the paragraph 'Pose refinement' below).

The objects are positioned on a tripod such that the centroid of the object is approximately at the center of the reachability domain of the robot. For each object, dense outside-in views are captured by positioning the camera on a sphere centered on the object's centroid and with a radius ∼30cm. The positions are evenly distributed on the sphere with latitude and longitude spanning over [0°,150°] and [0°,360°] respectively, with intervals of 10°. The lowest camera position at latitude 150° allows to capture views of the objects from all sides. Since the object is a source of collisions, the number of reachable views varies depending on the object's shape and size. The number of images per object varies between 397 and 505 images with an average of 480. The motion planning and the collision avoidance are operated with the proprietary IIWA stack [11] and the open-source MoveIt [6] framework using the CHOMP planning algorithm [21].

Image Registration. The YCB-V dataset [30] provides test images depicting the objects and their ground-truth poses. These poses are the rotation and the translation between the camera coordinate frame and the object's coordinate frame, which is the coordinate frame of the original CAD model. This coordinate frame is different from the 'world' coordinate frame where we collect the images and reconstruct the objects. Thus, the YCB-V ground-truth poses are not compatible with the coordinate frame of the 3D reconstructions. Consequently, we register the collected images to the original CAD model for each object. To do so, we first generate 3D reconstructions for all objects in their coordinate frame with COLMAP [24] then estimate the rigid transform between the reconstruction's coordinate frame and the CAD model's coordinate frame with Iterative Closest Point (ICP) [3, 36]. The ICP is initialized with user-defined coarse alignments.

Pose Refinement. The camera extrinsics of the collected images are obtained from the kinematic chain of the robot that can be relatively noisy. One source of such noise is in the robotic arm's positional encoders: they are known to be noisier in configurations that are close to degenerate, *e.g.*, when the arm is extended. This can affect the quality of the reconstruction methods so we refine the camera extrinsics in two steps: i) we use sequentially the pose refinement capabilities of iNGP [19] that update the poses consistently with the trained 3D reconstruction; ii) we then use the iNGP-refined poses to generate a sparse 3D reconstruction and refine the camera poses again with bundle adjustment using COLMAP [24]. After this step, the camera extrinsics remain fixed. Although some evaluated methods have pose refinement capabilities, we turn them off so that all reconstruction methods share the same coordinate frame. Without this, the reconstruction's coordinate frames may be slightly different for each method depending on the amplitude of the pose refinement, which would introduce inconsistencies in the evaluation.

Object masks. Once the images are registered with the original YCB-V [30]'s mesh, we generate object masks by projecting the mesh onto the image. We sample a dense set of points on the mesh and project them on the image using the intrinsics and the registered extrinsics. The pixels where the projected points fall form the object's mask.

Subset Generation. In addition to the full set of captured images, we define image sets of different sizes as, in certain applications, data capture and 3D reconstruction times can be important. Providing smaller subsets of images allows us to simulate such scenarios. We provide subsets of the following sizes: 25, 50, 75, 100, 150, and 300. The subsets are generated from the full set of images with Fibonacci sampling [10] to ensure that the subset views cover the object uniformly.

A.2. Data Release

The released data is available on the project's website:

`reconstruction_pose_benchmark`

The webpage contains instructions on how to download the data from an Apache server, either through a web page or a simple command line tool, *e.g.*, `wget`.

Released Data and Format. We release the calibrated undistorted images of the 21 YCB-V [30] objects collected in Sec. A.1, the objects' masks, and the meshes generated by the reconstruction methods evaluated in the benchmark.

For each object, the undistorted images, their extrinsics, the camera intrinsics, and the object masks are packed in a single zip file. The images and the masks are in `png` format, the extrinsics and intrinsics are released under both the Nerfstudio [26] camera pose convention[3] and the COLMAP [24] camera pose convention[4]. These are two of the most commonly used camera pose conventions for 3D reconstruction. Dataparsers for these two formats are available at the respective repositories.

For each reconstruction method, a single zip file contains all reconstructed objects. The 3D reconstructions are saved

[2] https://github.com/ros-planning/moveit_calibration

[3] docs.nerf.studio/quickstart/data_conventions.html

[4] colmap.github.io/format.html#text-format

Figure 2. **The 21 YCB-V [5, 30] objects** rendered from original CAD models.

under the standard `obj` format usually used by 3D processing libraries, *e.g.*, Meshlab [7], Open3D [36], trimesh [8], and Pytorch3D [22].

License. The collected images and the reconstructed meshes are released under the CC BY 4.0 license: the license is conditioned on the license of the YCB [5] objects depicted in the images. The YCB objects are released under the Creative Commons Attribution 4.0 International (CC BY 4.0) [5] so the data is released under the same license.

Benchmark Reproducibility. The benchmark uses open-source code for the reconstructions, the pose estimation, and the evaluation. The only exception may be the code of the reconstruction method RealityCapture [23], which is free except for companies with high earnings[6]. Unless mentioned otherwise in Sec.B1, the default parameters of the reconstruction methods are used.

B. Evaluation Setup

Compared to existing benchmarks for 3D reconstruction, our benchmark does not treat 3D reconstruction as a task unto itself but rather evaluates the resulting 3D models inside a higher-level task, *i.e.*, object pose estimation. Thus, we mainly measure 3D reconstruction performance by the accuracy of the estimated poses rather than the accuracy of the resulting 3D models itself. To this end, we compare the performance of two SotA object pose estimators [15, 29] when using the 3D reconstructed models against the use of classical CAD models.

[5] http://ycb-benchmarks.s3-website-us-east-1.amazonaws.com/
[6] https://www.capturingreality.com/pricing-changes

In the rest of this section, we report the implementation details relative to the evaluated 3D reconstructions (Sec B.1), the pose estimators used for the evaluation (Sec B.2), the pose evaluation metrics (Sec. B.3), and the nature of the objects in the evaluation dataset (Sec B.4).

B.1. Reconstruction Implementation Details

We describe the experimental setup related to the 3D reconstruction.

Colmap [24, 25]. We use the default parameters of COLMAP's triangulation [24, 25], which are already optimized for the purpose of reconstruction. The feature extraction returns RootSIFT [2, 18] features and the feature matching runs only between covisible image pairs: two images are covisible if they form an angle smaller than 45° with respect to the object's center. Before the Poisson [14] mesh reconstruction, we filter out the room's background by roughly cropping the dense point cloud around the object. We run the Poisson surface reconstruction with parameters `depth=10, trim=1`.

Reality Capture [23]. Given the known camera extrinsics and intrinsics, we first obtain a sparse point cloud model using the "draft" mode, which downsamples the images by a factor of two. For the following processing steps, a reconstruction region is defined as a bounding box (roughly) around the camera positions. All scene parts outside the region are ignored. A 3D mesh is then computed for this region in "normal detail", which also uses images downsampled by a factor of two. This reconstruction stage first computes depth maps to obtain a dense point cloud, from which a mesh is then extracted. Typically, there are artifacts

in the form of additional connected components. We only keep the largest connected component, which in all cases corresponded to (parts of) the object. The remaining mesh is then textured.

Implicit Methods. Nerfacto [26] is trained for 30K iterations without pose refinement, with normal prediction, and with the default parameters set by Nerfstudio [26]. iNGP [19] is trained with the default parameters set by the authors for 30K iterations. A mesh is extracted from the resulting models using the Poisson surface reconstruction [14].

MonoSDF [34], VolSDF [31], BakedSDF [32], and Neus [28] are trained with the default configuration set in SdfStudio [33], NeuralAngelo [16] and Plenoxel [9] with the configurations set by the authors. VolSDF and Neus are trained for 100k steps, MonoSDF for 200k iterations, BakedSDF for 250k iterations, NeuralAngelo for 500K iterations, and Plexoxel for 128K iterations, as recommended by their respective authors. These methods output an Signed Distance Function (SDF) from which the mesh is extracted using the marching cube algorithm [17] with SDF values sampled on a 1024x1024x1024 grid and a subsampling factor of 8. The NeuralAngelo [16] mesh is extracted with a 2048-resolution grid, and the Plexoxel [9] one with a 256-resolution grid, as recommended by their respective authors. For all methods, the SDF is initialized with a unit sphere. The same setup is adopted for Unisurf [20], except that it outputs an occupancy grid instead of an SDF.

iNGP [19] implicit rendering. In addition to evaluating the mesh derived with iNGP [19], we also evaluate how well the novel view synthesis of iNGP can replace the rendering of CAD models for pose estimation. We used the default settings of the iNGP renderer, except for the number of samples per pixel that is decreased to 1. We edit the Megapose [15] codebase to replace the CAD renderings with the iNGP implicit renderings.

Hardware. To provide fair runtime comparisons, all reconstruction methods are run on a single NVIDIA A100 GPU, 32x Intel Xeon-SC 8628, 24 cores, 2,9 GHz with 256GB of RAM.

One exception though is for RealityCapture [23] that runs on a GeForce RTX 3060 with 12 GB of RAM, *i.e.*, a weaker GPU than the other methods. This exception is because RealityCapture required a Windows installation that we had available only on a machine equipped with GeForce.

B.2. Pose Estimators: FoundationPose and Megapose

We recall how the two pose estimators used in the evaluation operate and refer the reader to their respective papers for more details.

FoundationPose [29][7] is a generalizable render-and-compare pose estimator that takes as input an RGBD image with an object of interest and a 3D representation of that object to output a pose.

The pose estimation proceeds in 2 steps: it first generates a set of pose hypotheses all around the object that are later refined by a network. This refinement network is given both the RGBD test image and the RGBD renderings from the pose hypotheses generated with the 3D representation. A second network then ranks the refined pose hypotheses using the RGBD test image and the RGBD rendering produced from the refined poses. The best-ranked pose is the final output.

One of the main contributions of FoundationPose is that the 3D representation can take two forms as long as color and depth renderings can be generated from it: a traditional CAD model or an implicit representation. The pose estimation is agnostic to the 3D representation as it only uses the RGBD renderings. Also, it is jointly trained with renderings from the CAD models and the implicit representations, which reduces any possible distribution shift between the two types of renderings.

Whenever the 3D representation of the object at hand is not available, FoundationPose generates a 3D implicit representation of the object at test time. Thus FoundationPose can be deployed on any object even when the 3D representation is not known beforehand, which reflects the practical conditions in which pose estimators are deployed.

The 3D implicit model can generate RGBD renderings in two ways: either with novel-view synthesis directly or by first extracting a mesh out of the implicit representation and then rastering the mesh. The latter is computationally more efficient, as observed by the FoundationPose authors [29]. We adopt a similar derivation where we extract a mesh out of the evaluated implicit 3D reconstruction methods to produce RGBD renderings.

Megapose [15][8] is a also generalizable render-and-compare pose estimator that takes as input an RGB image depicting an object and a 3D model of that object and outputs the pose of the object. Megapose is made of two modules: a coarse pose estimator and a pose refiner. Given a test image cropped around the object, **the coarse module** generates n pose hypothesis all around the objects from which renderings are generated. The concatenation of the test image and the renderings are fed to a classification network which logits are interpreted as a score for each pose hypothesis. The top-K coarse poses are kept and refined: for each coarse pose, the refiner renders the object's mesh from that coarse pose and from three additional views. The views aim at creating parallax and are generated by moving the camera laterally while ensuring that the camera-z axis goes through

[7] https://github.com/NVlabs/FoundationPose
[8] https://github.com/megapose6d/megapose6d

the object's centroid. **The refiner network** is a regression network that takes as input the test image and the four renders defined previously and outputs a pose update to apply to the coarse pose. The final pose is the composition of the coarse pose with the pose update. This final pose can also be refined by repeating the refinement step. We adopt the author's recommendation and set the number of refinement iterations to 5. We feed the coarse module with the maximal number of initial pose hypothesis $n = 576$ and we keep the top-1 coarse pose, *i.e.*, setting $K = 1$, since we observe marginal improvement with $K = 5$ (see Fig. 3) and it runs faster.

Before being fed to the network, the image is cropped around the object of interest. One can either use a box detector or the ground-truth box to define the region of interest and we use the latter for both Megapose and FoudationPose.

B.3. BOP Evaluation

As is custom for object pose estimation, we report the three standard pose errors defined in the BOP benchmark [12, 13]. The metrics reported are the Visible Surface Discrepancy (VSD), the Maximum Symmetry-Aware Surface Distance (MSSD), and the Maximum Symmetry-Aware Projection Distance (MSPD). We refer the reader to the well-documented BOP benchmark methodology for details[9]. We recall the definitions here for the sake of completeness. **Note that all these errors measure the pose error, hence the lower, the better.**

The **Maximum Symmetry-Aware Surface Distance (MSSD)** measures the maximum surface misalignment between the surface of the object when it is positioned with the ground-truth pose and when it is positioned with the estimated pose. In both cases, the object is positioned with respect to the camera coordinate frame. As indicated in the name, the MSSD is symmetry-aware, *i.e.*, it does not penalize the estimated pose for ambiguous symmetries (*e.g.* a textureless sphere has an infinite number of non-resolvable symmetries). *Derivation details:* Let M be the CAD object model, $\hat{\mathbf{P}}$ the estimated pose, $\bar{\mathbf{P}}$ the ground-truth pose, V_M a set of points sampled on the reference CAD model M (*i.e.* the YCB-V [5, 30] one), S_M the set of symmetries of the object. The MSSD is computed as:

$$e_{\text{MSSD}}(\hat{\mathbf{P}}, \bar{\mathbf{P}}, S_M, V_M) = \min_{\mathbf{S} \in S_M} \max_{\mathbf{x} \in V_M} \|\hat{\mathbf{P}}\mathbf{x} - \bar{\mathbf{P}}\mathbf{S}\mathbf{x}\|_2 \ .$$

The metric is computed over the set of points V_M sampled on the reference mesh positioned with the ground-truth pose and with the estimated pose. Corresponding points form pairs between which the distance is computed. The MSSD reports the maximum distance over all pairs. To account for symmetries, the derivation computes several distances for each pair of points where the ground-truth point is additionally transformed with an isometry to account for the symmetry ($\min_{\mathbf{S} \in S_M}$). The actual distance between two points in the minimum over all symmetries. The symmetries are provided by the evaluation dataset as annotations. **Interpretation:** as an upper bound on the surface misalignment, the MSSD is very sensitive as even a small pose error can induce high surface discrepancies around high-curvature regions of the object. In practice, such high-curvature regions are suitable grasping points for **robotic manipulation**, so a high AR-MSSD (Average Recall-MSSD, see below) suggests that the object pose estimation is suitable for robotic grasping.

The **Maximum Symmetry-Aware Projection Distance (MSPD)** measures the maximum pixel displacement induced by an inaccurate pose estimate when rendering the object. *Derivation details:* The notation is the same as for the MSSD metric. Let M be the CAD object model, $\hat{\mathbf{P}}$ the estimated pose, $\bar{\mathbf{P}}$ the ground-truth pose, V_M a set of points sampled on the reference CAD model M (*i.e.* the YCB-V [5, 30] one), and S_M the set of symmetries of the object. The MSPD is computed as:

$$e_{\text{MSPD}}(\hat{\mathbf{P}}, \bar{\mathbf{P}}, S_M, V_M) = \min_{\mathbf{S} \in S_M} \max_{\mathbf{x} \in V_M} \|\text{proj}(\hat{\mathbf{P}}\mathbf{x}) - \text{proj}(\bar{\mathbf{P}}\mathbf{S}\mathbf{x})\|_2 \ .$$

As for the MSSD, the metric is computed on the set of points V_M sampled on the reference CAD model when it is positioned with the ground-truth pose and with the estimated poses, respectively. The points are projected onto the image plane and the MSPD reports the maximum distance between the projected points over all pairs. **Interpretation:** Note that the projective nature of the MSPD makes it unsuitable for applications that require physical interactions with the world. Instead, this perceptual error is better suited for vision applications such as **Augmented and Virtual Reality** that exploit the rendering of positioned objects.

The **Visible Surface Discrepancy (VSD)** measures the average misalignment of the visible surface of the object along the camera-z axis. *Derivation details:* Let $\hat{V}$ and $\bar{V}$ be visibility masks, *i.e.*, the set of pixels where the reference model M is visible in the image when the model is positioned at the estimated pose $\hat{P}$ and the ground-truth pose $\bar{P}$. Let $\hat{D}$ and $\bar{D}$ be the distance maps obtained by rendering the reference object model M in the estimated pose $\hat{P}$ and the ground-truth pose $\bar{P}$, respectively. The VSD is computed as:

$$e_{\text{VSD}}(\hat{D}, \bar{D}, \hat{V}, \bar{V}, \tau) = \text{avg}_{p \in \hat{V} \cup \bar{V}} \begin{cases} 0 & \text{if } p \in \hat{V} \cap \bar{V} \wedge \|\hat{D}(p) - \bar{D}(p)\| \leq \tau \\ 1 & \text{otherwise} \end{cases} .$$

[9] https://bop.felk.cvut.cz/challenges/bop-challenge-2019/#evaluationmethodology

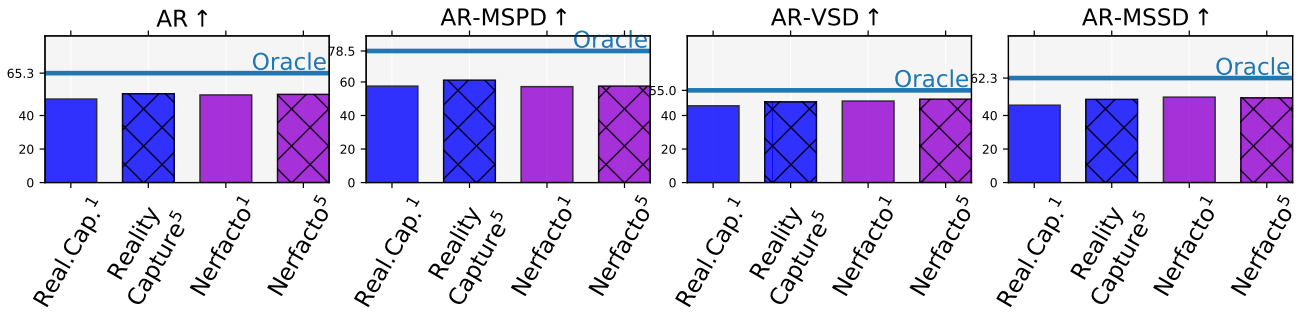

Figure 3. **Megapose Parameter Tuning**: comparison of the pose performance when keeping the top-1 (uniform bar) and top-5 (crossed bar) pose hypothesis for pose refinement. The experiment runs on two of the best-performing reconstruction methods RealityCapture [23] and Nerfacto [26]. The results show that using a higher number of coarse hypotheses marginally improves the scores for RealityCapture (+3%) and barely affects Nerfacto.

Category	Objects
Lambertian-Textured-Large	02-cracker-box, 03-sugar-box, 16-wooden-block
Lambertian-Textured-Small	07-pudding-box, 08-gelatin-box, 18-large-marker
Shiny-Textured	01-master-chef-can, 04-tomato-soup-can, 06-tuna-fish-can, 09-potted-meat-can,13-bowl, 14-mug
Uniform-Texture	10-banana, 11-pitcher-base, 17-scissors, 19-large-clamp, 20-extra-large-clamp, 21-foam-brick
Low-Texture	05-mustard-bottle, 12-bleach-cleanser
Scissors-Like	17-scissors, 19-large-clamp, 20-extra-large-clamp
Legacy-Objects	03-sugar-box, 04-tomato-soup-can, 10-banana, 12-bleach-cleaner, 13-bowl, 14-mug, 16-wooden-block, 17-scissors, 18-large-marker, 19-large-clamp, 20-extra-large-clamp, 21-foam-brick
Updated-Objects	01-master-chef-can, 02-cracker-box, 05-mustard-bottle, 06-tuna-fish-can, 07-pudding-box, 08-gelatin-box, 09-potted-meat-can, 11-pitcher-base, 15-power-drill

Table 1. **Object Categories.** The categories are characteristic of object properties such as size, shape, texture, and materials. We also distinguish between the objects that are the same between the YCB-V datasets [5, 30] and our data collection ('legacy') and the ones that got updated by their manufacturer ('updated').

The VSD is also a perceptual metric but the definition of the distance is slightly different. It measures a distance in 3D but contrary to the MSSD, it does not measure it between the corresponding points sampled on the mesh. Instead, the pairs are formed by taking a visible point from each mesh that projects onto the same pixel. In practice, the CAD model is rendered from the two poses to generate distance maps (*i.e.*, depth maps) and the VSD corresponds to the average number of misplaced rendered pixels. Misplacement can either mean an incorrect distance map value or an incorrect position in the image, *e.g.*, if the estimated pose translates the object too much in the camera-x or camera-y directions, the rendered pixels will be disjoint on the image space. **Interpretation:** a low VSD indicates that the object is well positioned, yet can miss to report errors in the estimated pose's rotation. If the object is mostly symmetric, a low VSD can indicate that the centroid of the object is well estimated but it provides less information on the rotation's error. For example, take the 14-mug object and assume that the estimated pose is the ground-truth pose flipped so that the mug is upside-down. Then the position of the CAD models positioned with the ground-truth pose and the estimated pose differ only on the side in which the 14-mug handle is. Instead, the surface of the 14-mug's "body" will be aligned. Thus, the distance maps of these points will be mostly equal and will dominate the metric, indicating a good position. Thus, the VSD is better suited **for applications with tolerance to such edge-cases such as robotic navigation.**

Throughout the evaluation, the metrics are derived

on the same model M which is the original CAD model provided in the YCB-V [5, 30] dataset, *i.e.*, the reconstructed meshes are not used when computing the metrics. This ensures that the results for poses obtained from different meshes are comparable.

Note that the BOP benchmark does not report these errors as is but reports the **Average Recall (AR)** on these errors. Given an error threshold, the recall is the fraction of the estimated poses for which the error falls below the threshold. **Since these metrics quantify the pose error, the lower, the better.** The recall is computed separately over each metric and as usual **for recalls, the higher is better**. The Average Recall (AR) for a given metric, which we report as AR-VSD, AR-MSPD, AR-MSSD, is the average of several recalls computed over a range of error thresholds. Finally, we also report the **global average recall AR** computed as the average of the three average recalls AR-VSD, AR-MSPD, AR-MSSD. As specified in the BOP benchmark, it measures a method's overall performance.

The recall averaging used in the benchmark slightly differs from the averaging of the BOP benchmark. Instead of averaging the metrics over the set of all test images, we averaged the recall over the test images depicting a given object instance (*e.g.* 02-cracker-box) to output the object-specific AR-MSPD, AR-MSSD, AR-VSD, AR. The global performance of the method is the average of these metrics over the set of objects. The performance of the methods of specific object categories is the average of the subset of objects that make that category. Because of the change in the averaging order, the metrics we report can be slightly lower than the ones reported in the BOP benchmark by about 5% on average. This edit is not critical as we are interested in the relative pose performances rather than the absolute ones.

B.4. Object Categorization

We split the YCB-V objects into categories with specific properties depending on their size, their texture and their materials (Tab. 1).

Lambertian-Textured-Large / Small refers to large / small objects strongly textured, with diffuse reflections.

Shiny-Textured objects are medium and small objects which material has high reflectivity.

Uniform-Texture objects have no texture, *i.e.*, they are color-uniform whereas **Low-texture** objects have both high-texture areas and textureless areas.

Scissors-like: Objects with scissors-like shapes that may have holes inside the mesh and have symmetries from different views.

Updated Objects: The YCB-V [30] are made of objects commonly found in grocery stores. However, brands regularly update their product packaging, which means that the object's texture gets updated. Some of the original YCB-V [30] objects are not available on the market anymore so we use the alternatives suggested by the official YCB [5] website[10]. The newer versions of the objects differ in texture and scale, with an average change estimated at 4% of the object's dimensions. As for the texture update, the 01-master-chef-can, 11-pitcher-base, and 15-power-drill underwent full texture updates and other objects have only minor updates.

Legacy Objects are the objects that are the same between the YCB-V [30] dataset and our data collection, *i.e.*, all objects minus the updated objects.

C. Additional Results

We report the quantitative results that support the conclusions drawn in the main paper and that could not fit due to the page limit: the average recall on each of the three pose errors (Sec. C.1), the influence of the size of the training data on the performance (Sec. C.2), further results on the relation between the object categories and the pose performance (Sec. C.3), the importance of the texture for pose estimation (Sec C.4), the detail of the pose performances per objects (Sec C.5), and qualitative results showing renderings from the reconstructed meshes (Sec C.6).

C.1. Pose Evaluation: Average Recall

The BOP benchmark defines multiple errors relevant to assess how suitable a reconstruction is for a given application such as augmented and virtual reality (MSPD), navigation (VSD), and object manipulation (MSSD) [1, 12, 13]. In the main paper, the pose performances are measured with the Average Recall (AR) averaged over all three VSD, MSSD, and MSPD errors We now report the AR for each error separately (Fig. 4) to support the conclusions of the main paper (Sec.5 "Finer Pose Evaluation").

The performance gap between Megapose [15] and FoundationPose [29] when they are evaluated with the original CAD models is relatively smaller for the MSPD than for the other errors. One interpretation is the improvement of FoundationPose over Megapose is particularly useful for 3D spatial tasks, as measured by the VSD and the MSSD, *e.g.*, for navigation and object manipulation. The improvement is smaller for tasks involving object rendering for which Megapose is close to FoundationPose.

The main paper mentions the performance drop induced when replacing the original CAD models with the reconstructed ones and how it is consitent between the two pose estimators for the best reconstruction methods, *i.e.*, RealityCapture [23], Nerfacto [26], VolSDF [31], MonoSDF [34], BakedSDF [32], and NeuralAngelo [16]. This remains the case for the individual ARs and we also observe that the relative performances of the reconstruction methods are consistent across the different errors. So one can expect that im-

[10] https://www.ycbbenchmarks.com/object-set/

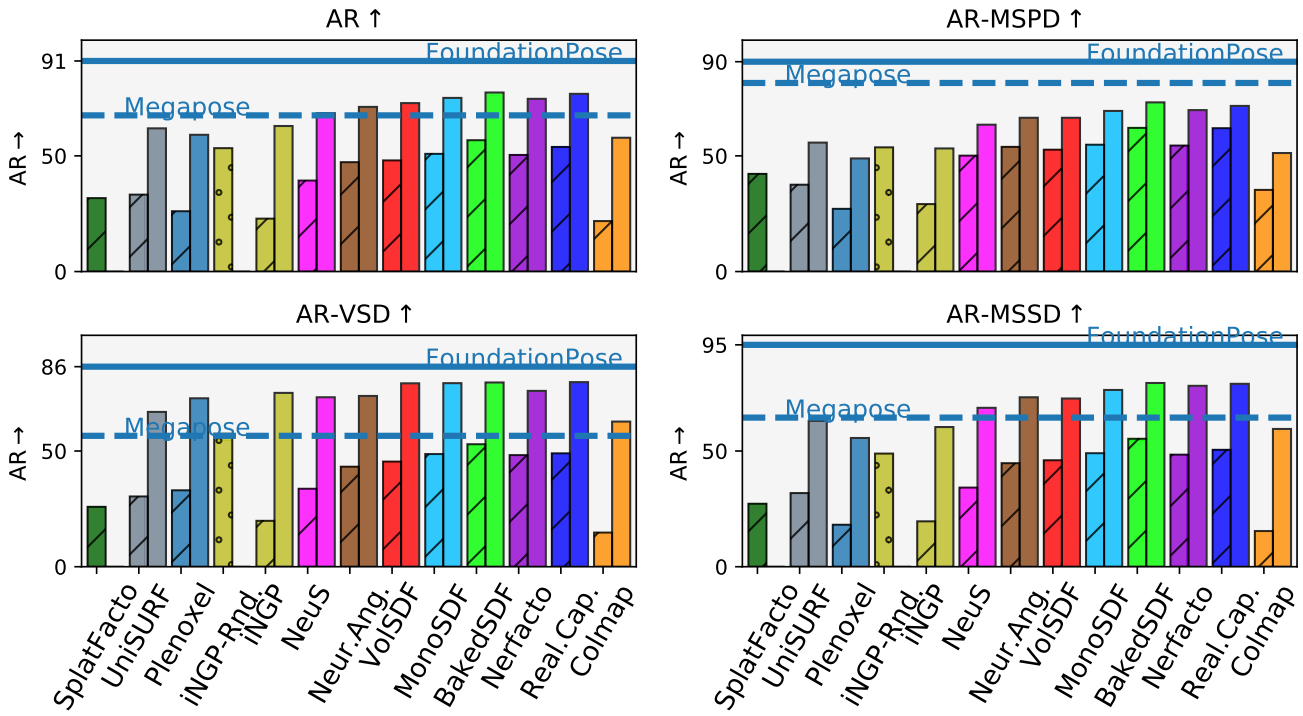

Figure 4. **Detailed Pose Evaluation of the 3D reconstructions.** We measure the performance of FoundationPose [29] (uniform bar) and Megapose [15] (lined bar) when replacing the CAD models with the 3D reconstructions. The blue lines indicate the performance of the pose estimators when using the original CAD models (FoundationPose: **full line**, Megapose: **dashed line**). Overall, a gap remains between CAD models and 3D reconstructions for pose estimation. This gap is smaller on the VSD, which indicates that 3D reconstructions can be suitable for pose estimation in applications with tolerance to pose errors.

proving the 3D reconstructions might benefit all three pose estimation errors, hence various applications involving object pose estimation.

One interesting observation is the gap between the original CAD models and the best reconstruction methods is relatively small on the VSD compared to the other errors. The VSD indicates that the object is overall well positioned so this suggests that 3D reconstructions can be suitable replacements for the CAD model when the pose estimator is used in tasks that are tolerant to the pose errors, *e.g.*, visual navigation. However, the gaps on the MSSD and MSPD remain large so there is room for improvement for 3D reconstructions to replace CAD models in tasks with low tolerance in the pose errors, *e.g.*, object manipulation.

C.2. Reconstruction with varying training size

Besides the quality of the pose estimation, an important property of the reconstruction methods is their data requirement. To evaluate the 'data-greediness' of the reconstruction methods, we reconstruct the YCB-V [30] objects from subsets of images sampled uniformly around the object [10]. Fig. 5 complete the results in the main paper with the pose performance of FoundationPose [29] with reconstructions from image subsets and the 3D reconstruction's geometric accuracy.

As for Megapose [15], the performance of FoundationPose [29] improves as more images are used for the 3D reconstruction (top-left). Nerfacto [26] plateaus with only 75-100 images and the performance of RealityCapture [23] remains relatively stable as the number of images decreases to 25. This further demonstrates the practical advantage of RealityCapture that is both fast and data-efficient over other reconstruction methods: the reconstruction takes less than 1 min. on 25 images and $\sim$2 min. on 50 images. Still, there remains a gap between the 3D reconstructions and the original CAD models for pose estimation that calls for further improvements in 3D reconstruction.

We observe that FoundationPose [29] seems more resilient to innacurate 3D reconstructions than Megapose [15]. For example, the geometric quality of Nerfacto [26] drops slightly at 150 images which is reflected in the Megapose performance but not in the FoundationPose ones. A similar observation holds for BakedSDF [32] at 100 and 150 images.

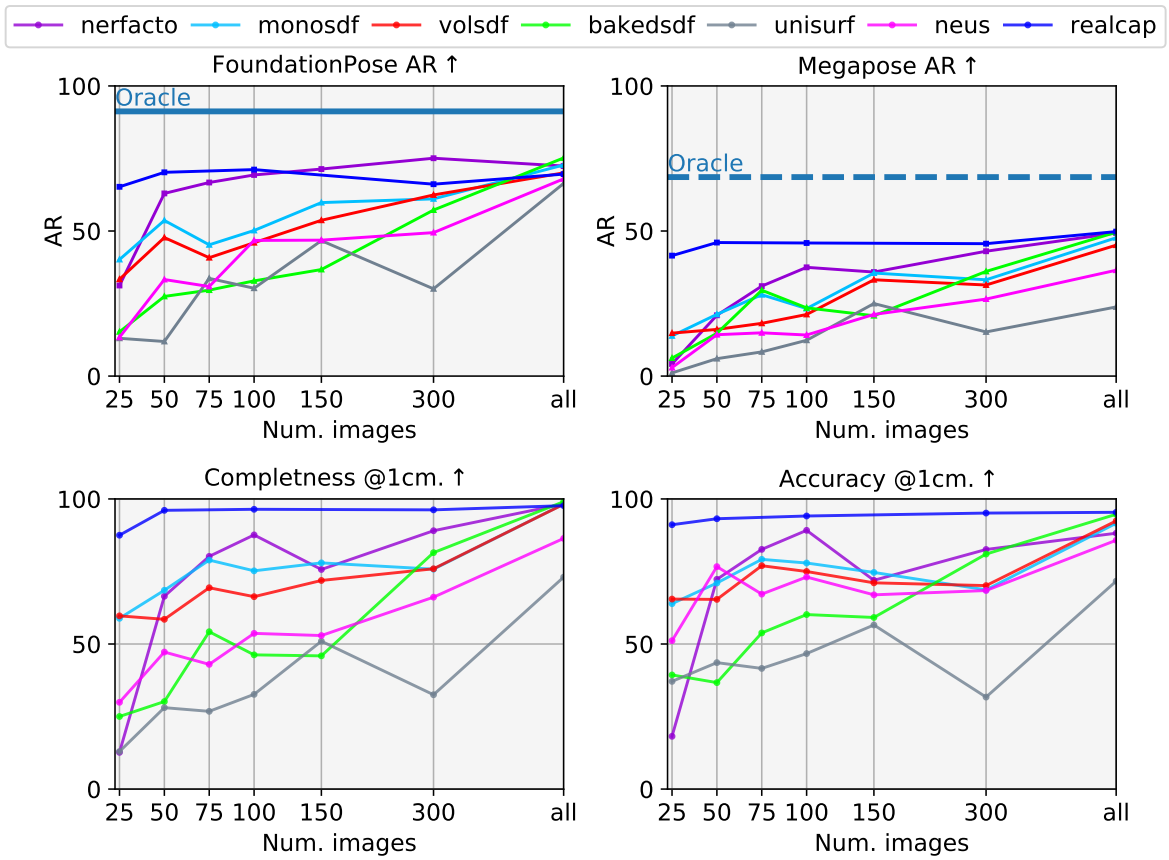

Figure 5. **Pose Evaluation of the 3D reconstructions from subsets of images. Top:** We measure the performance of FoundationPose [29] and Megapose [15] when replacing the CAD models with the 3D reconstructions generated from image sets of varying size. The blue lines indicate the performance of the pose estimators when using the original CAD models (FoundationPose: **full line**, Megapose: **dashed line**). As expected, the more data the better the results and 75-100 images are enough to get reasonable reconstructions. The performance of Reality Capture [23] is relatively stable as the number of images decreases: **the reconstruction from 50 images takes as little as 2 minutes** and is already close to the best RealityCapture performance. **Bottom:** Evaluation of the reconstructed geometry with the traditional completeness and accuracy metrics.

C.3. Object Categories

We split the YCB-V [30] objects into 8 groups defined in Sec. B.4 based on their shape, texture, material properties, and the degree of change between the collected objects and the original ones. We report results for all object categories in Fig. 6 (including the 3 reported in the main paper) and observe that the relative pose performance across object categories is telling of some of the current challenges in 3D reconstruction.

The variation in pose performance is relatively large between the most simple categories and the most challenging ones. Simple object categories include large objects with Lambertian texture and objects with little or uniform texture. For these objects, replacing the CAD model with the 3D reconstructions has little or no impact on the performances. We also note that some of the reconstruction methods that perform relatively lower than others, *e.g.* Neus [28], NeuralAngelo [16], achieve very good results on some of the simple object categories. Another interesting result is the high performance of the MVS-based RealityCapture [23] on objects with little texture: one could have expected lower results since MVS relies on feature matching which accuracy usually drops when the image content exhibits little texture.

As mentioned in the main paper, the most challenging object categories are small textured objects and objects with a 'shiny' texture, *i.e.*, a texture with high reflectance. For these categories, the performance bottleneck often lies in the texture. The resolution of small reconstructed objects is relatively lower than for larger objects and that loss of information can impede the pose estimation. As for shiny objects, the reflection of light sources on the object during the data collection can cause visual discrepancies between

the reconstruction and the object in the test images.

Since the YCB-V dataset [5, 30] is made of objects commonly found in grocery stores, whenever a brand updates its packaging, the texture gets updated. The YCB-V objects we captured hence fall into two categories: the 'legacy' objects which texture did not change and the 'updated' objects which texture has changed. Only the coffee box '01-master-chef-can', the '11-pitcher-base, and the '15-power-drill' have undergone a full texture update and the other objects have undergone relatively minor updates. Still, the gap between the 3D reconstructions and the CAD models is lower on the 'legacy' objects than the 'updated ones'. BakedSDF [32] is even on par with the original CAD models with Megapose [15]. There still remains a gap between 3D reconstructions and CAD models though. Note that the integration of a given 3D reconstruction with FoundationPose [29] often outperforms or is on par with Megapose [15] running with the original CAD model: this suggests that the gap between CAD models and 3D reconstructions could be closed not only by improving 3D reconstructions but also pose estimators.

C.4. Influence of the Texturing Algorithm

Previously, we observed that the performance of the pose estimation drops on small objects or shiny objects and we argued that the performance bottleneck lies in the texturing. So we now analyze the influence of the texture on pose estimation (Fig. 7). In the first experiment, we compare the **pose performance of the 3D reconstructions with and without texture**. A second experiment assesses the influence of the texturing algorithm by **comparing the 3D reconstruction with native texturing and MVS-texturing** [27]. These reconstructions are compared to the original CAD models with texture **(dashed blue line)** and without texture **(dotted blue line)**.

We run this ablation on Megapose [15] with a subset of methods: Nerfacto [26], VolSDF [31], MonoSDF [34] and BakedSDF [32], the colored but textureless reconstructions from COLMAP [24, 25], Neus [28] and UniSURF [20].

We compare the reconstructions under three texturings: the texturing **native** to each reconstruction method, **no texturing** at all, and texturing with the **MVS-texturing** algorithm [27]. Note that COLMAP does not texture the models but only provides vertex colors so for COLMAP we compare the mesh colored with the point cloud colors, the colorless mesh, and the mesh textured with MVS-texturing. Native texturing refers to the texturing algorithm as implemented in NerfStudio [26] and SDFStudio [33]: the UV map is generated by querying the network for the color at the position of the mesh's faces. The textureless mesh has exactly the same geometry as the textured one but without the texture map and with no color. Finally, the MVS-textured mesh is the output of the MVS-texturing algorithm [27] run on the geometry of the reconstructed mesh.

With vs. Without Texture (circles). For all object categories, the gap between the textureless CAD model (dotted line) and the textureless reconstruction (circle bars) is relatively small or non-existent. This is the case even for the object categories that are the most challenging under the regular pose evaluation (small and shiny), which supports the assumption that the performance bottleneck for these categories is the texture. This suggests that the geometric quality of the 3D reconstructions and the CAD models are comparable as measured by pose estimation performance. This is in line with the results obtained with traditional geometric evaluation of the 3D reconstructions measured with the completeness and accuracy metrics.

When the texture is removed, whether from the CAD models or the 3D reconstructions, the pose performance drops less for objects with low or uniform texture than for textured objects. A reasonable explanation is that the pose estimation relies more on the geometry than on the texture for objects with little texture information. Hence, the absence of texture or even incorrect texturing may not be prohibitive for pose estimation on such objects as long as the reconstructed geometry is good enough.

Native vs. MVS-Texturing. The texturing native to each method (uniform bar) leads to either comparable or better results than MVS-texturing [27] (crossed bar). Even the textureless but colored meshes of COLMAP [24, 25] leads to better pose performance than with the MVS-texturing. This may appear counter-intuitive since MVS-Texturing [27] produces sharper and more visually appealing textures (see Fig. 8-right). However, render-and-compare methods usually operate on low-resolution renderings so that they can be fed efficiently to the network, so the lower resolution of the native texturing is not penalized. Also, MVS-Texturing [27] can generate strong color artifacts, such as color saturation or color mismatch that decrease the texture's quality (see the green tint as shown in Fig. 8-right). That could be due to multiple reasons: noise in the camera pose which leads to blurred texture (Fig. 8-left), the surface extracted from SDFs may not been extracted with the optimum isosurface value, or MVS-Texturing can not handle well reflections.

C.5. Per-Object Pose Evaluation

We report the pose performances separately for each YCB-V object [5, 30], 3D reconstruction, and pose estimators in Fig. 9 and Fig. 10. The results are consistent with the previous conclusions that the most challenging objects are the small ones or shiny ones, *e.g.*, 06-tuna-fish-can or the 18-large-marker.

As observed previously, the combination of FoundationPose [29] and the 3D reconstructions is often better than Megapose [15] with the original CAD models, *e.g.*, 02-

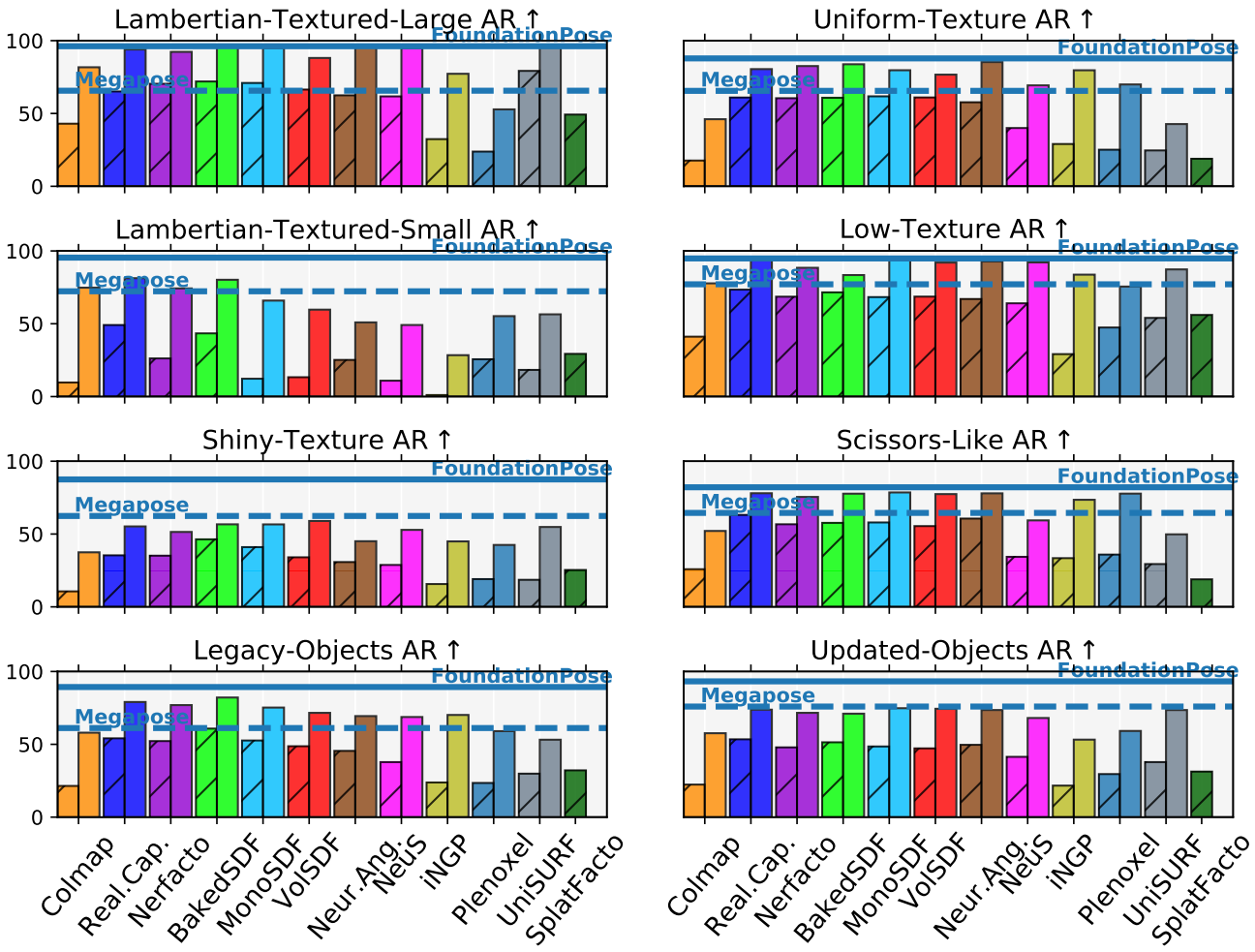

Figure 6. **Pose Evaluation of the 3D reconstructions for various Object Categories.** We measure the performance of FoundationPose [29] (uniform bar) and Megapose [15] (lined bar) when replacing the CAD models with the 3D reconstructions. The blue lines indicate the performance of the pose estimators when using the original CAD models (FoundationPose: **full line**, Megapose: **dashed line**). Some categories we expected to be challenging, such as objects with uniform texture are not. Instead, the most difficult categories are small and shiny objects.

cracker-box, 10-banana. This suggests that bridging the gap between 3D reconstructions and CAD models for pose estimations has two directions for improvements: improving the 3D reconstruction itself and improving pose estimators.

C.6. Qualitative Results

The SSIM, PSNR, and LPIPS [35] metrics are evaluated by comparing the rendering of the CAD model and the masked YCB-V images, as illustrated in Figure 11. We report the average values across all test images.

The 3D reconstructions are displayed in Figure 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32. Each figure showcases the object's reconstructions, the pose performances and the texture quality for various reconstruction methods.

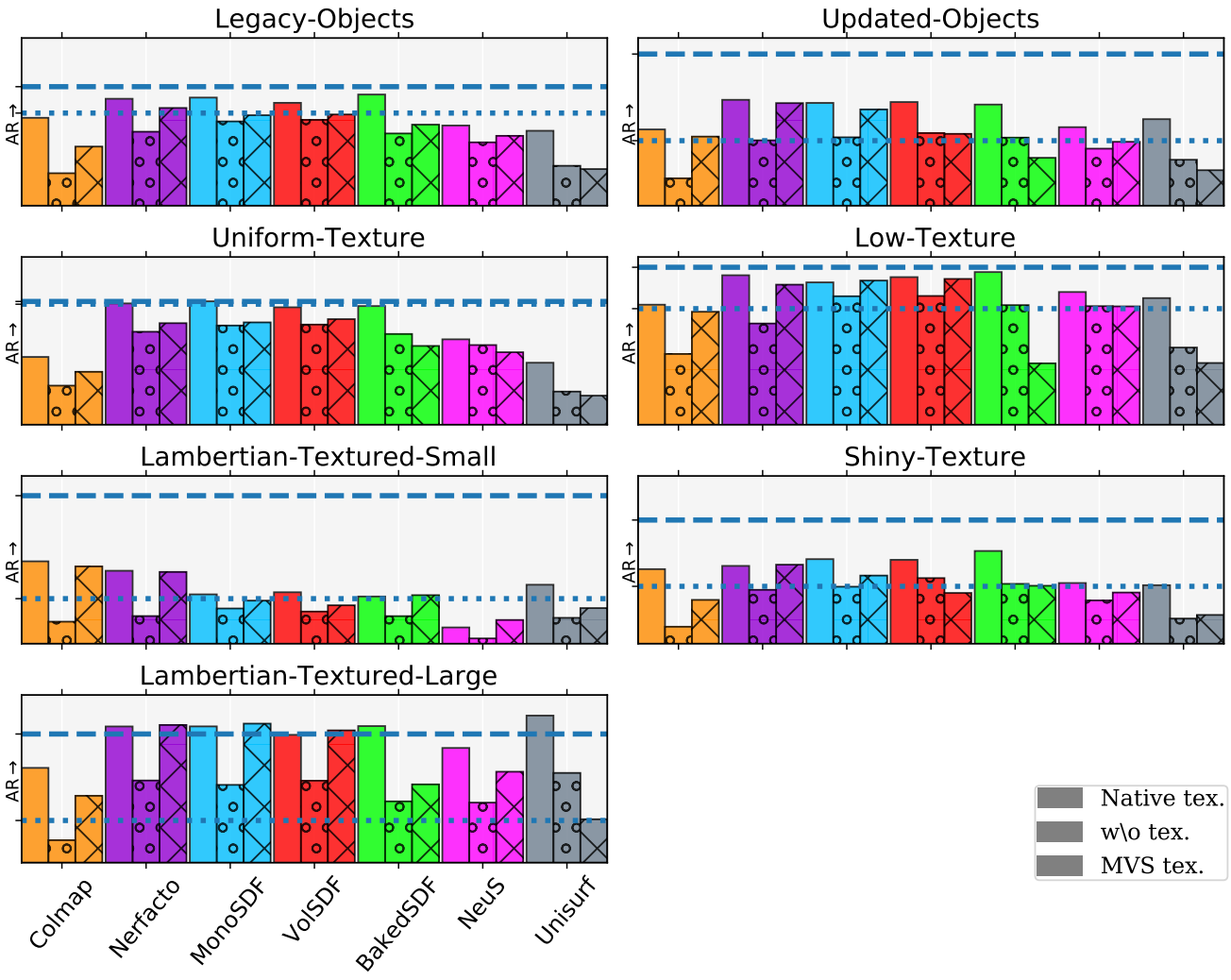

Figure 7. **Pose Evaluation of the 3D reconstructions under different texturings.** We measure the performance of Megapose [15] when replacing the CAD models with 3D reconstructions. The reconstructions are compared to the original CAD models with texture **(dashed line)** and without texture **(dotted line)**. The performances of the textureless CAD models and the textureless 3D reconstructions are on par, which suggests that the 3D reconstructions have geometry suitable for pose estimation.

Figure 8. Examples of texturing artifacts generated with MVS-texturing [27]. **Left**: Illustration of MVS-Texturing mirage artifact. The text in the 03-sugar-box is distorted and repeated. We also observe color saturation on the blue separation line between the top of the box and the bottom. **Right**: Illustration of the MVS-Texturing color artifacts. The top of the 02-cracker-box on the left (COLMAP + MVS-Texturing) exhibit green areas that do not exist in the original model and are not produced by native texturing of other methods (*e.g*. Nerfacto on the right).

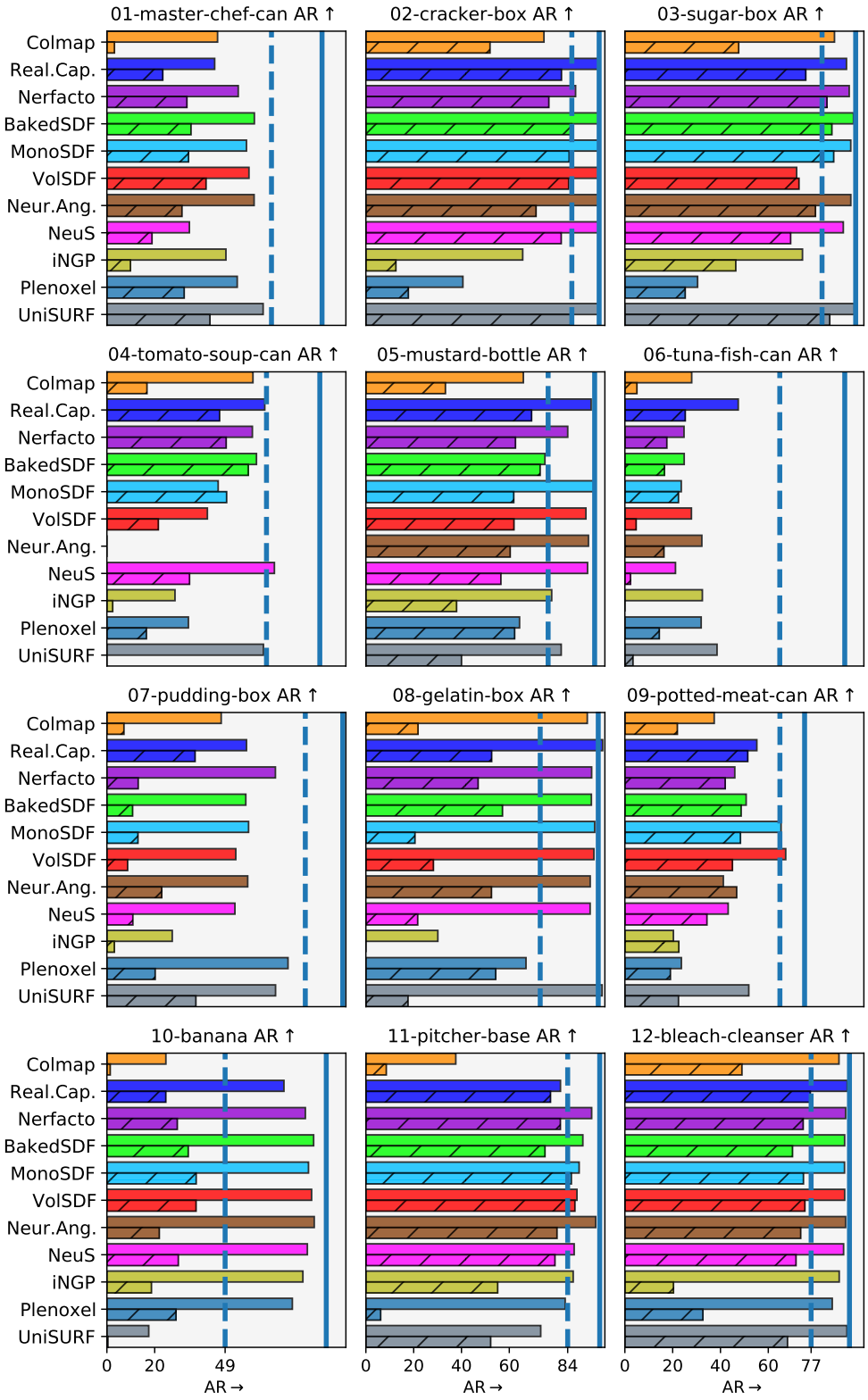

Figure 9. **Pose Evaluation of the 3D reconstructions for each YCB-V object [30] (1/2).** The 3D reconstructions are evaluated based on the performance of two pose estimators, **FoundationPose [29] (uniform bar)** and **Megapose [15] (lines bar)**, when the CAD model is replaced with 3D reconstructions. The blue vertical lines indicate the performance of the pose estimators when using the original CAD models (FoundationPose: **full line**, Megapose: **dashed line**). The objects for which it is the most challenging to replace the CAD model are the small and / or shiny objects, *e.g.*, 06-tuna-fish-can, 07-pudding-box. For a given 3D reconstruction, the use of a high-performance pose estimator can compensate for some of the limitations of the 3D reconstruction: for example, FoundationPose [29] induces a strong boost on 07-pudding-box compared to Megapose [15].

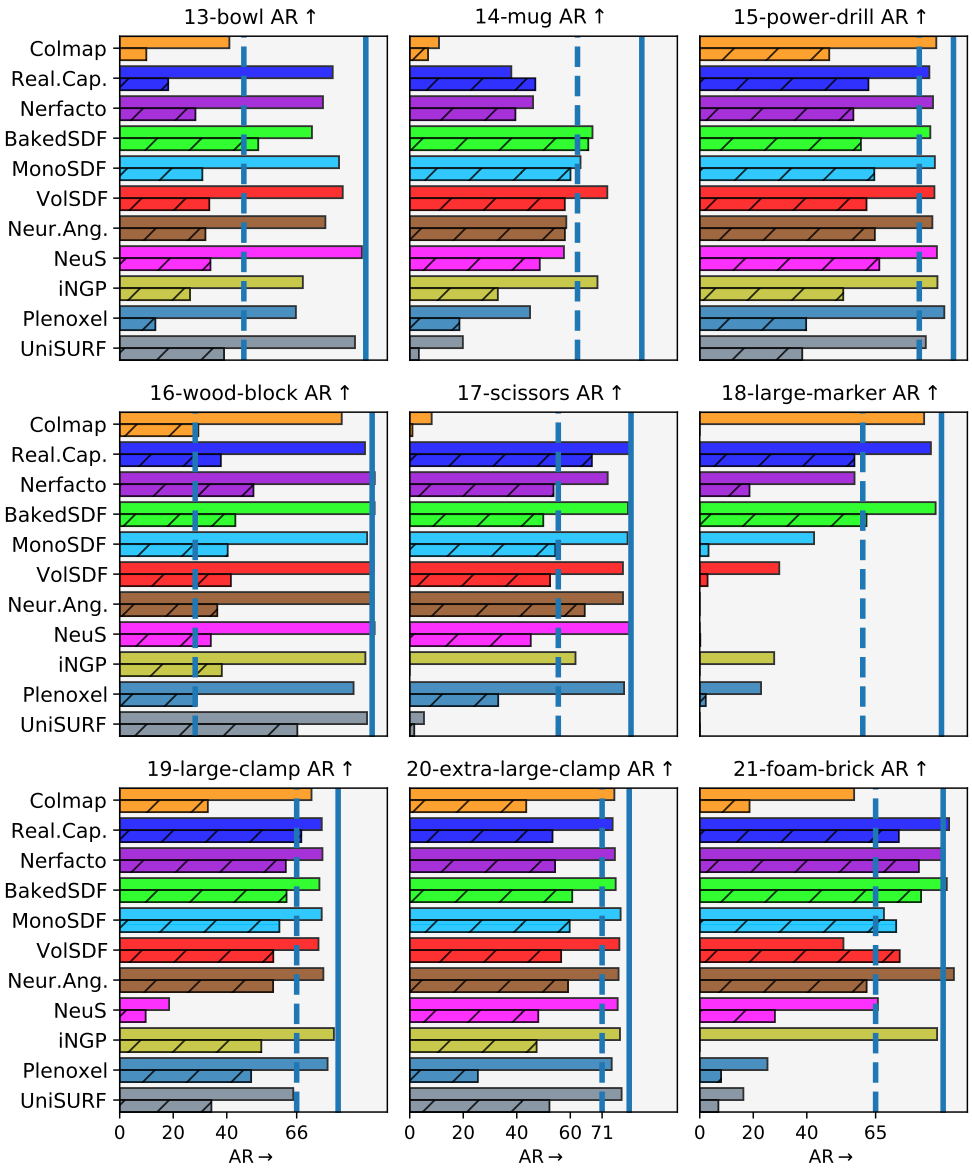

Figure 10. **Pose Evaluation of the 3D reconstructions for each YCB-V object [30] (1/2).** The 3D reconstructions are evaluated based on the performance of two pose estimators, **FoundationPose [29] (uniform bar)** and **Megapose [15] (lines bar)**, when the CAD model is replaced with 3D reconstructions. The blue vertical lines indicate the performance of the pose estimators when using the original CAD models (FoundationPose: **full line**, Megapose: **dashed line**).

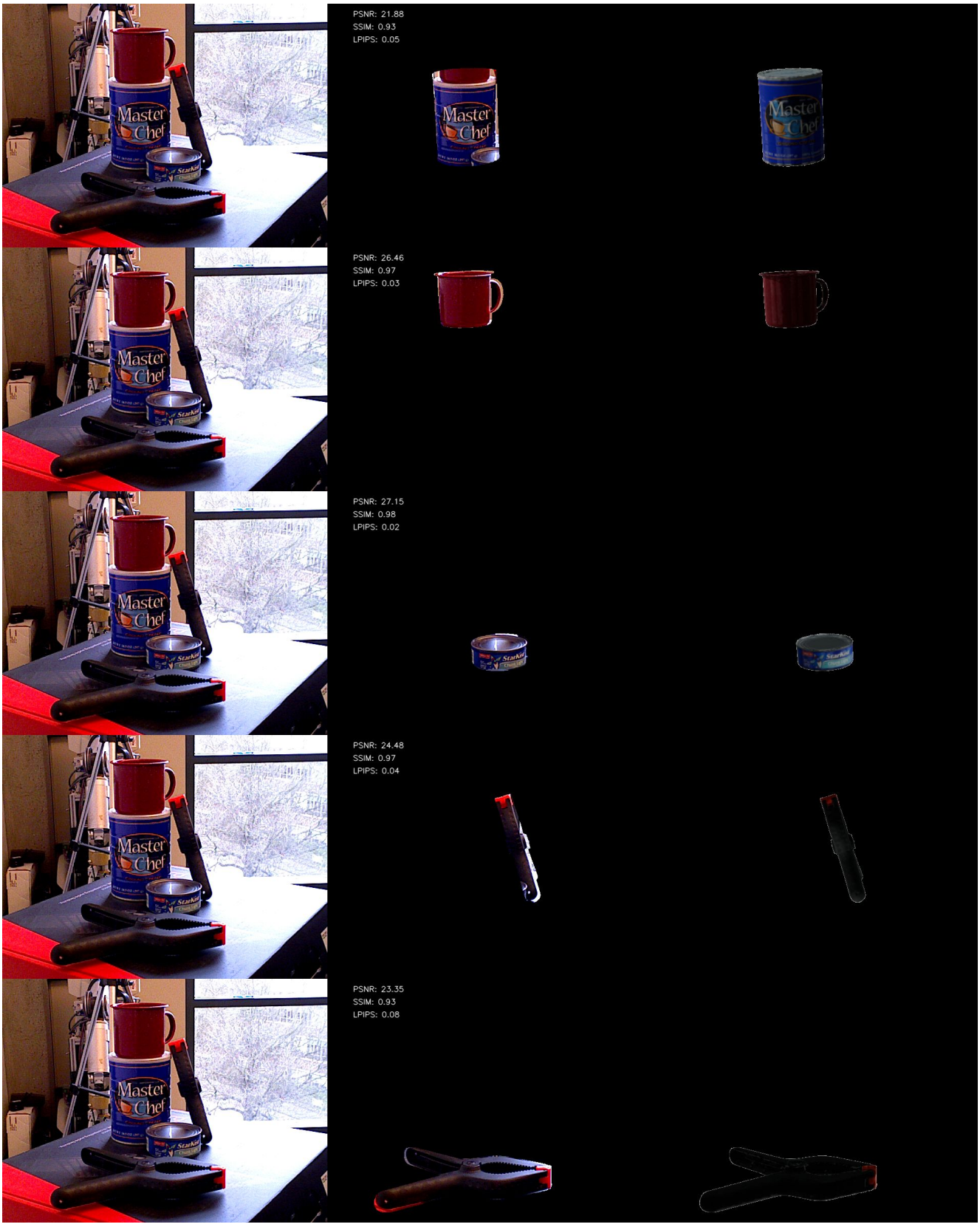

Figure 11. Methodology for evaluating SSIM, PSNR, and LPIPS metrics. On the left is the test YCB-V image, in the middle is the masked YCB-V object, and on the right is the rendering of the reconstructed mesh overlay on the masked image.

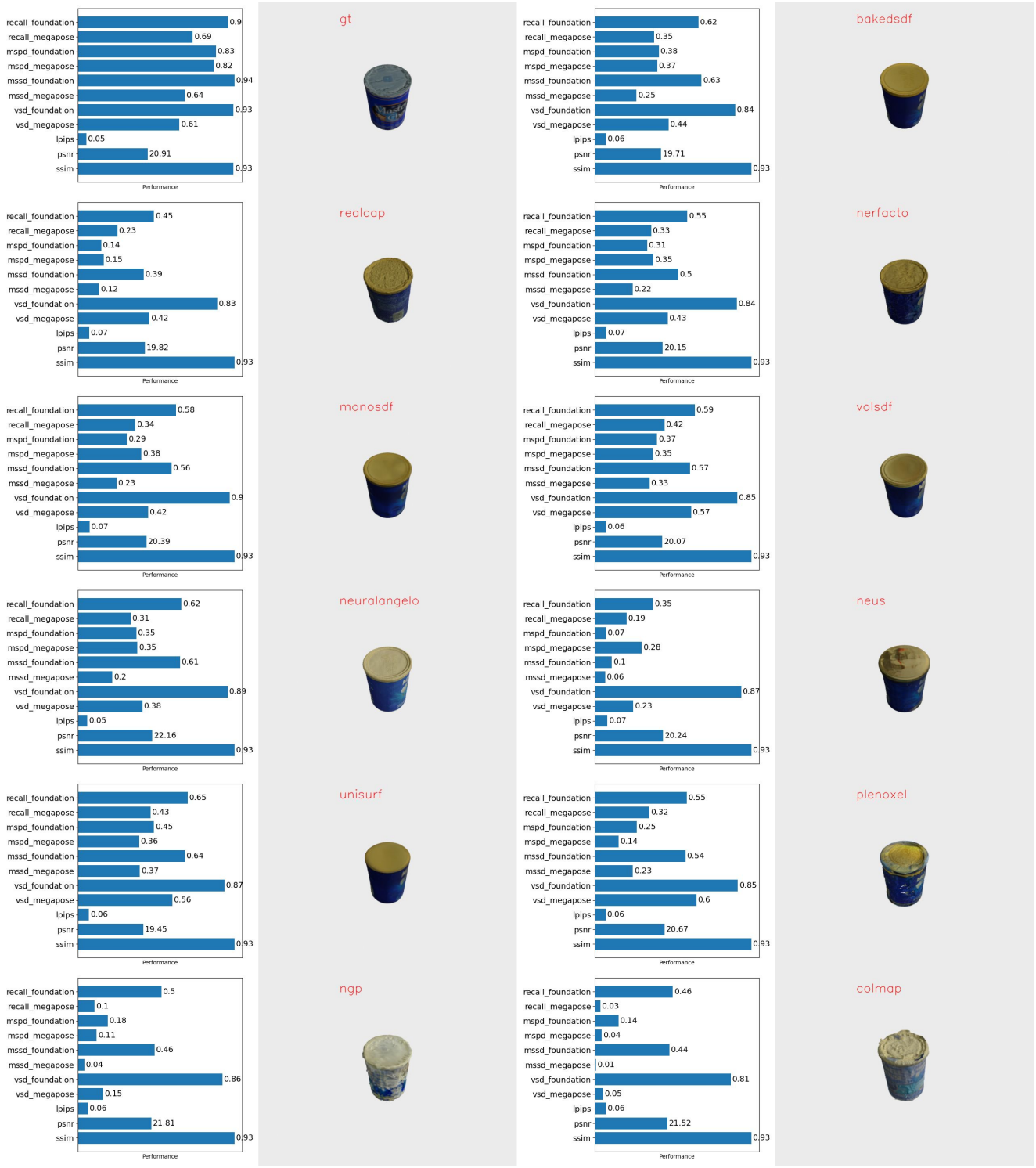

Figure 12. Pose performance and texture scores (left) and object renderings (right) of 01-master-chef-can reconstructed with various reconstructed methods.

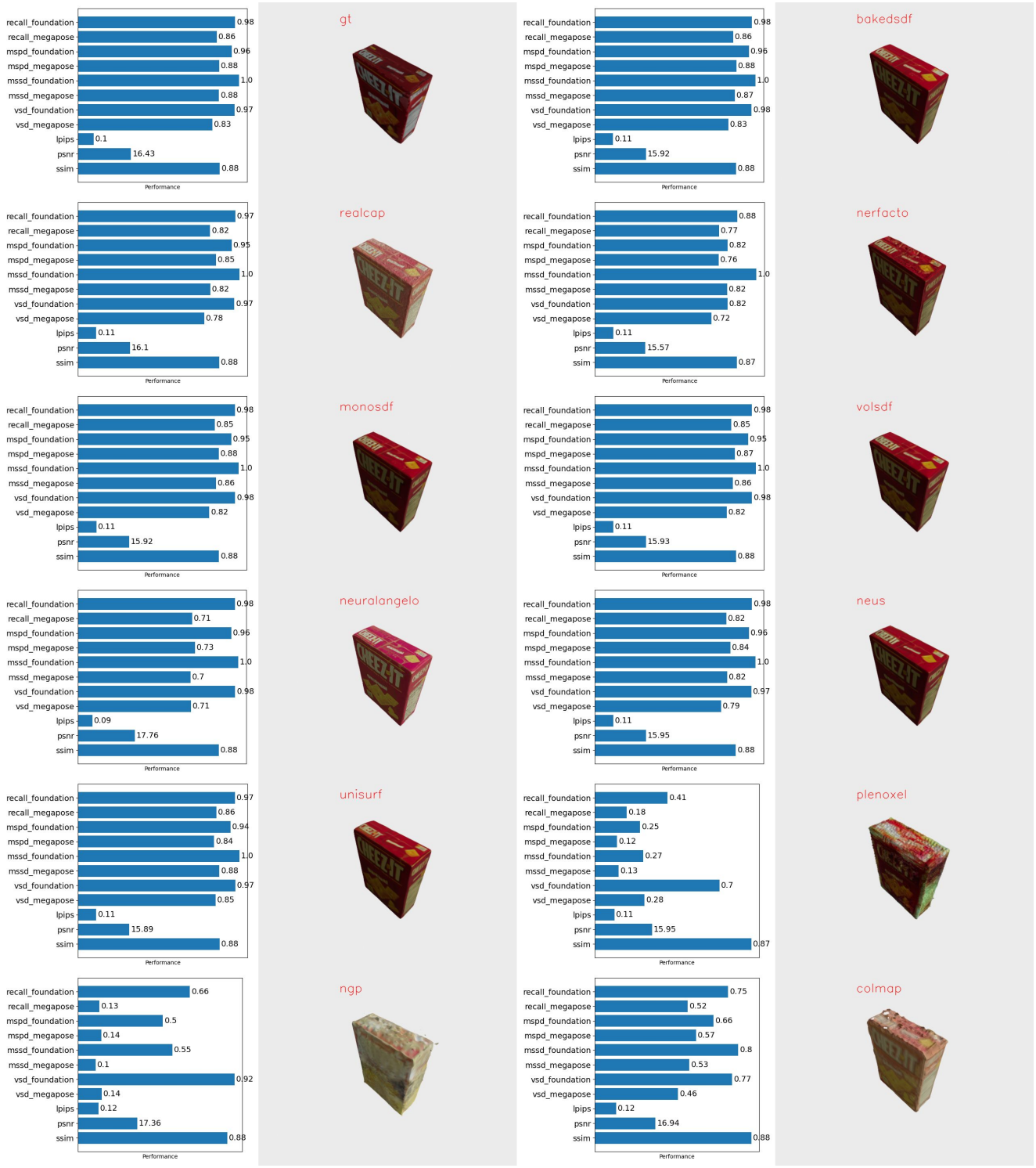

Figure 13. Pose performance and texture scores (left) and object renderings (right) of 02-cracker-box reconstructed with various reconstructed methods.

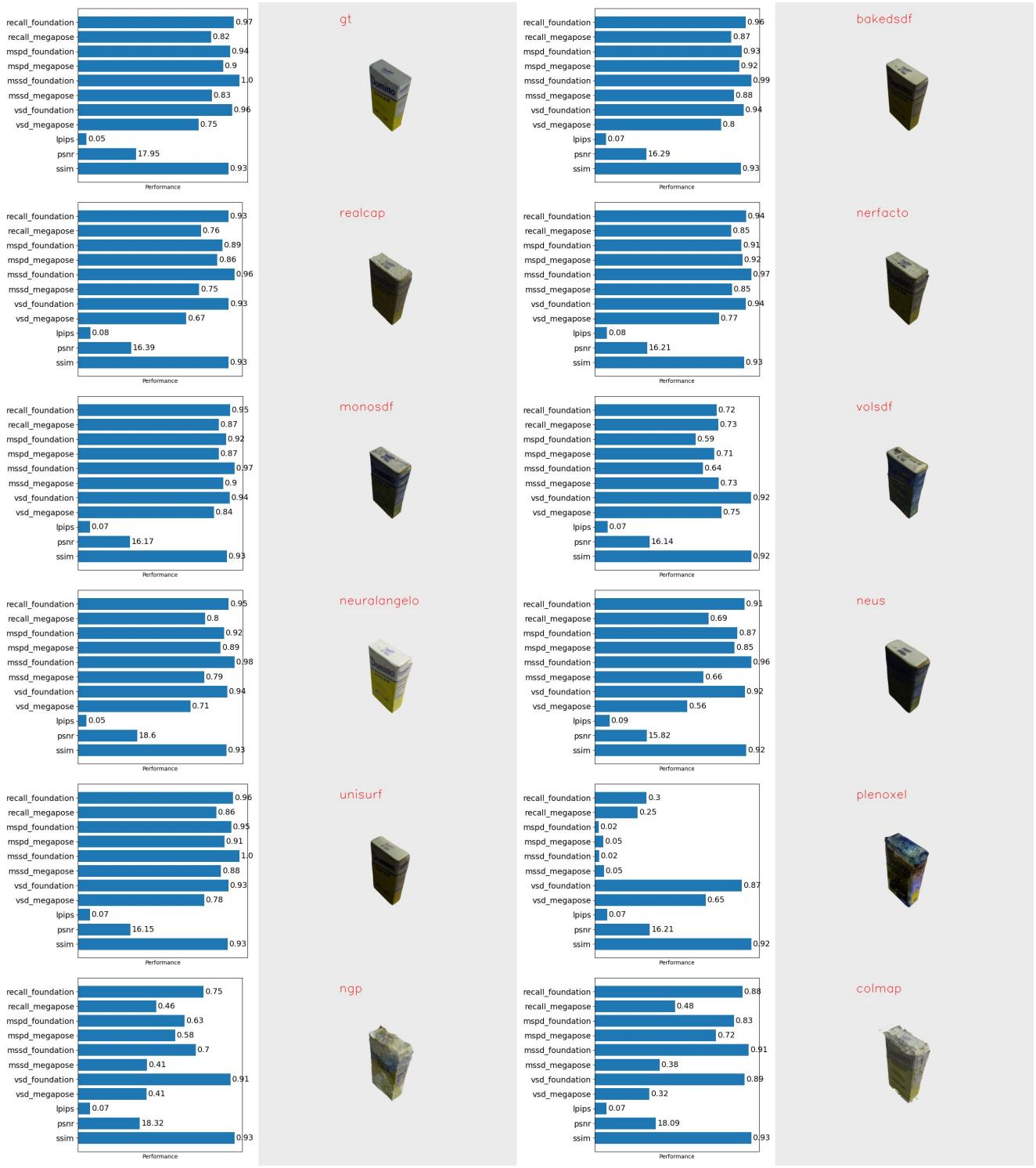

Figure 14. Pose performance and texture scores (left) and object renderings (right) of 03-sugar-box reconstructed with various reconstructed methods.

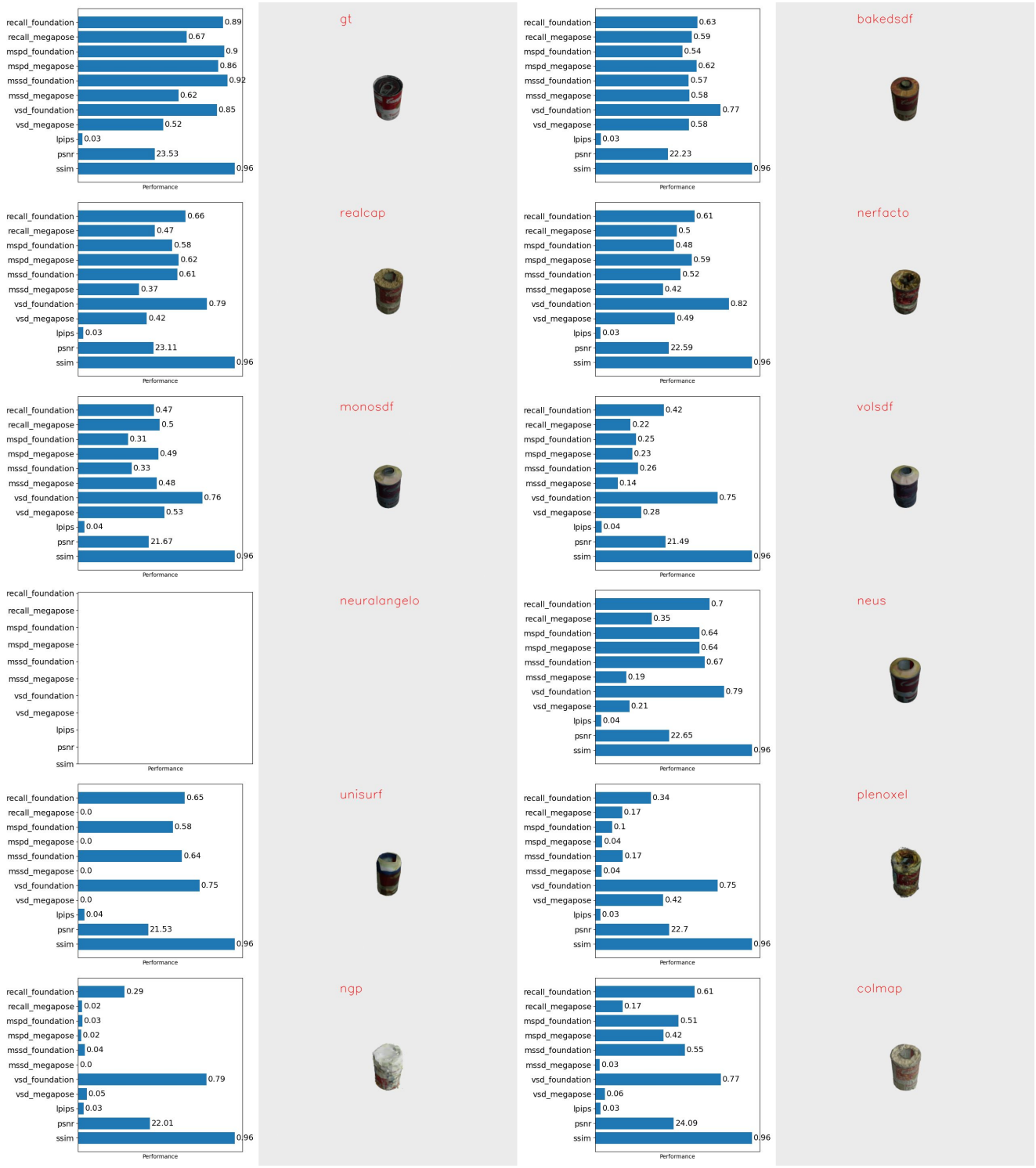

Figure 15. Pose performance and texture scores (left) and object renderings (right) of 04-tomatoe-soup-can reconstructed with various reconstructed methods.

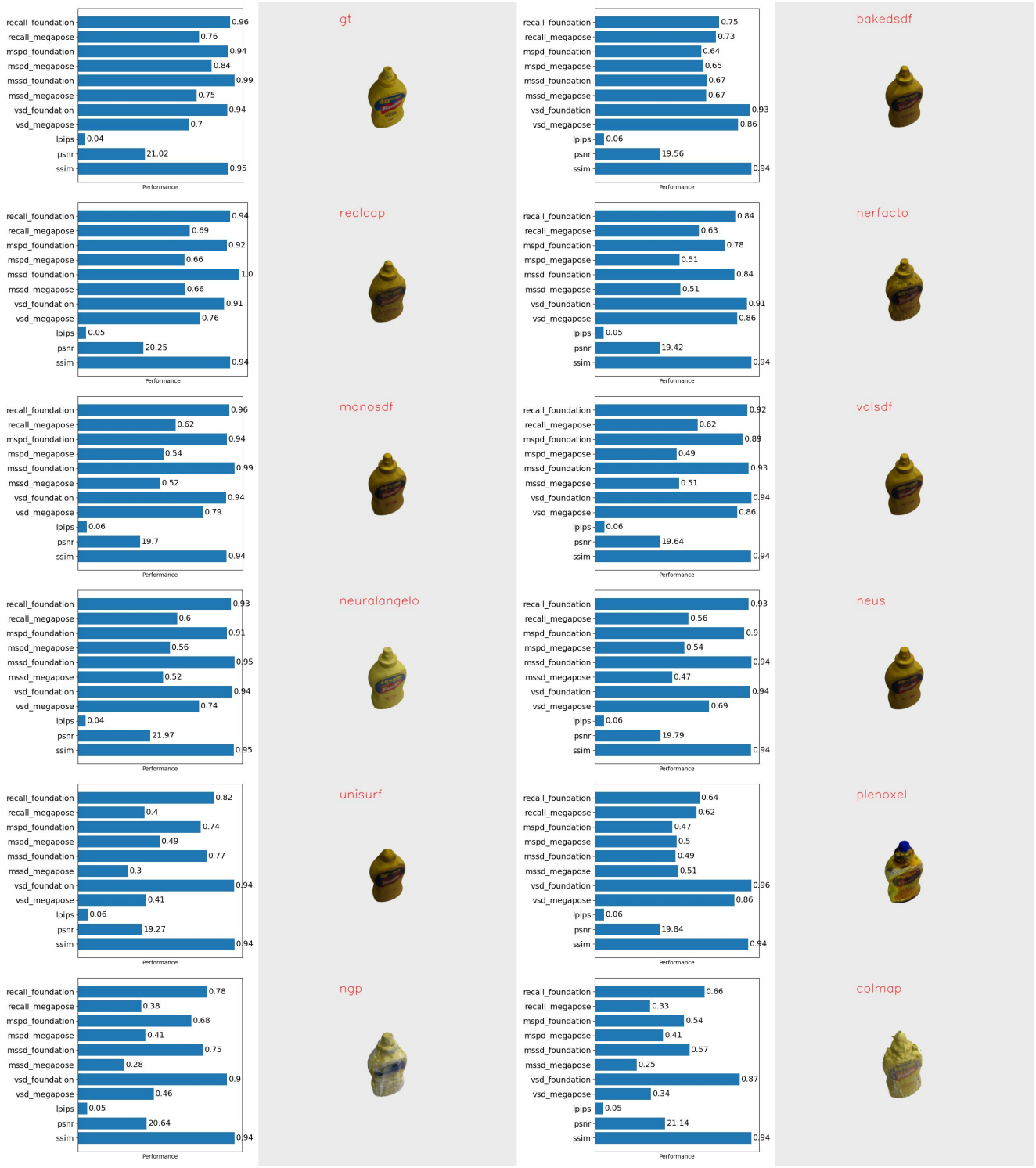

Figure 16. Pose performance and texture scores (left) and object renderings (right) of 05-mustard-bottle reconstructed with various reconstructed methods.

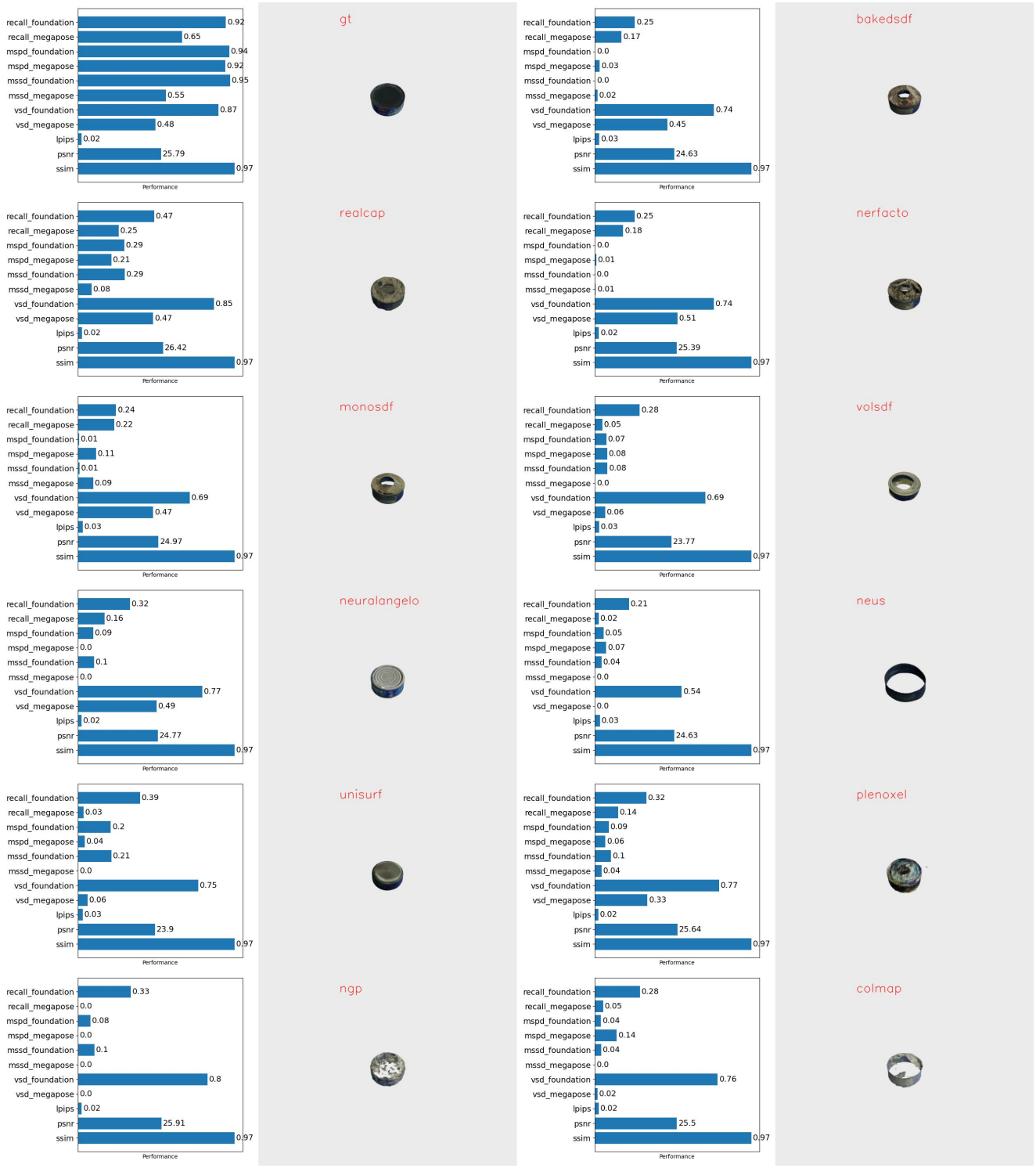

Figure 17. Pose performance and texture scores (left) and object renderings (right) of 06-tuna-fish-can reconstructed with various reconstructed methods.

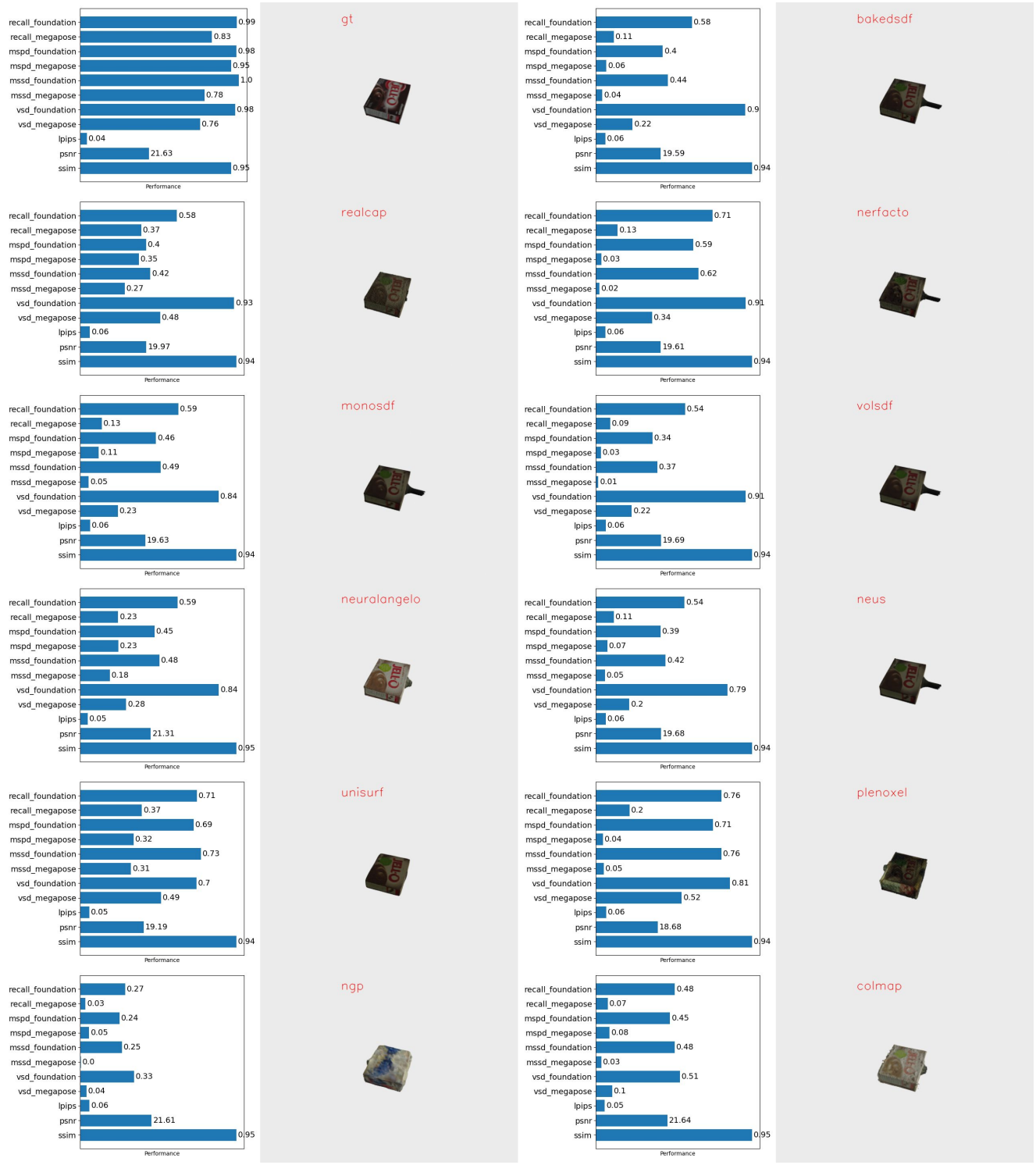

Figure 18. Pose performance and texture scores (left) and object renderings (right) of 07-pudding-box reconstructed with various reconstructed methods.

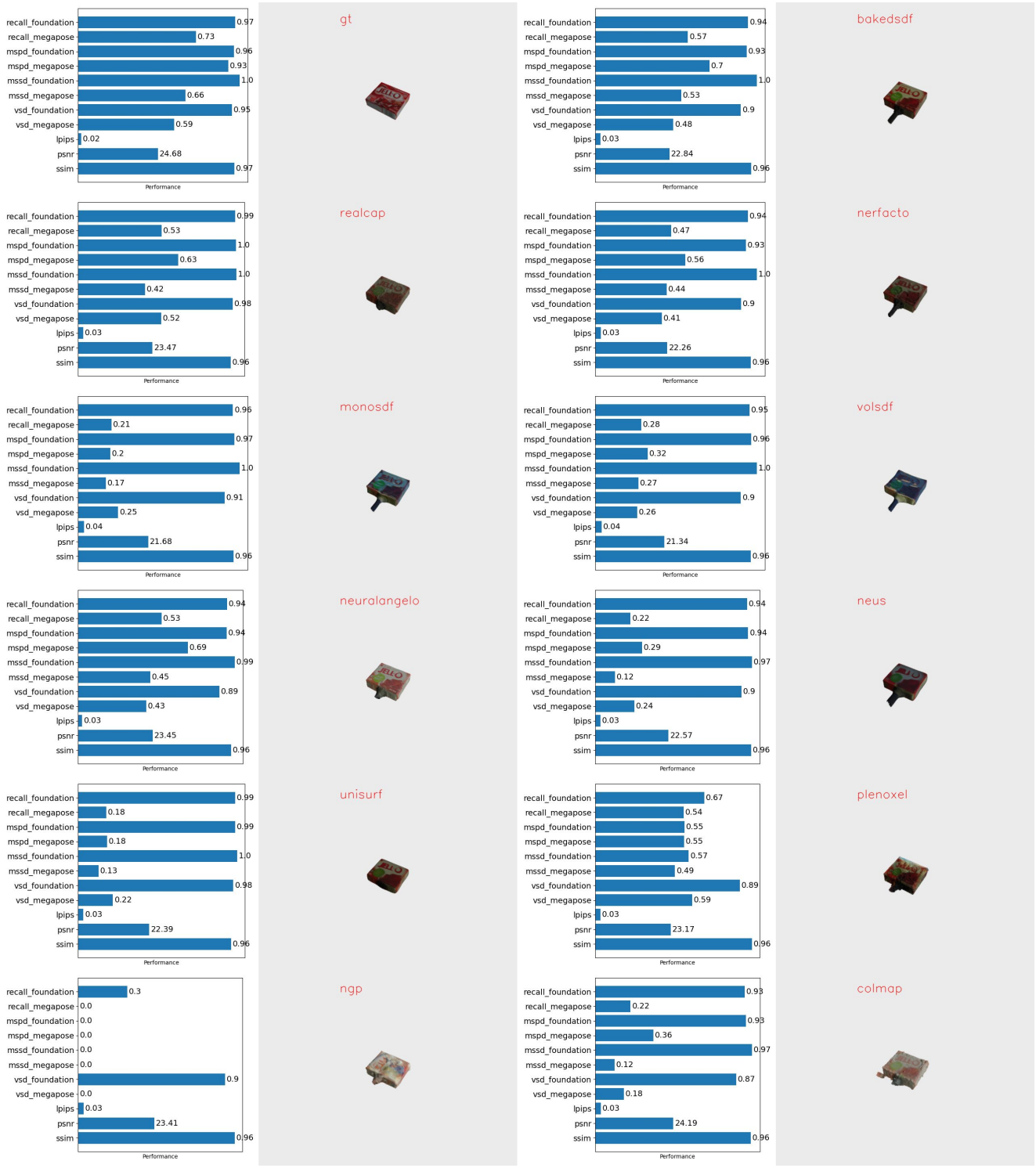

Figure 19. Pose performance and texture scores (left) and object renderings (right) of 08-gelatin-box reconstructed with various reconstructed methods.

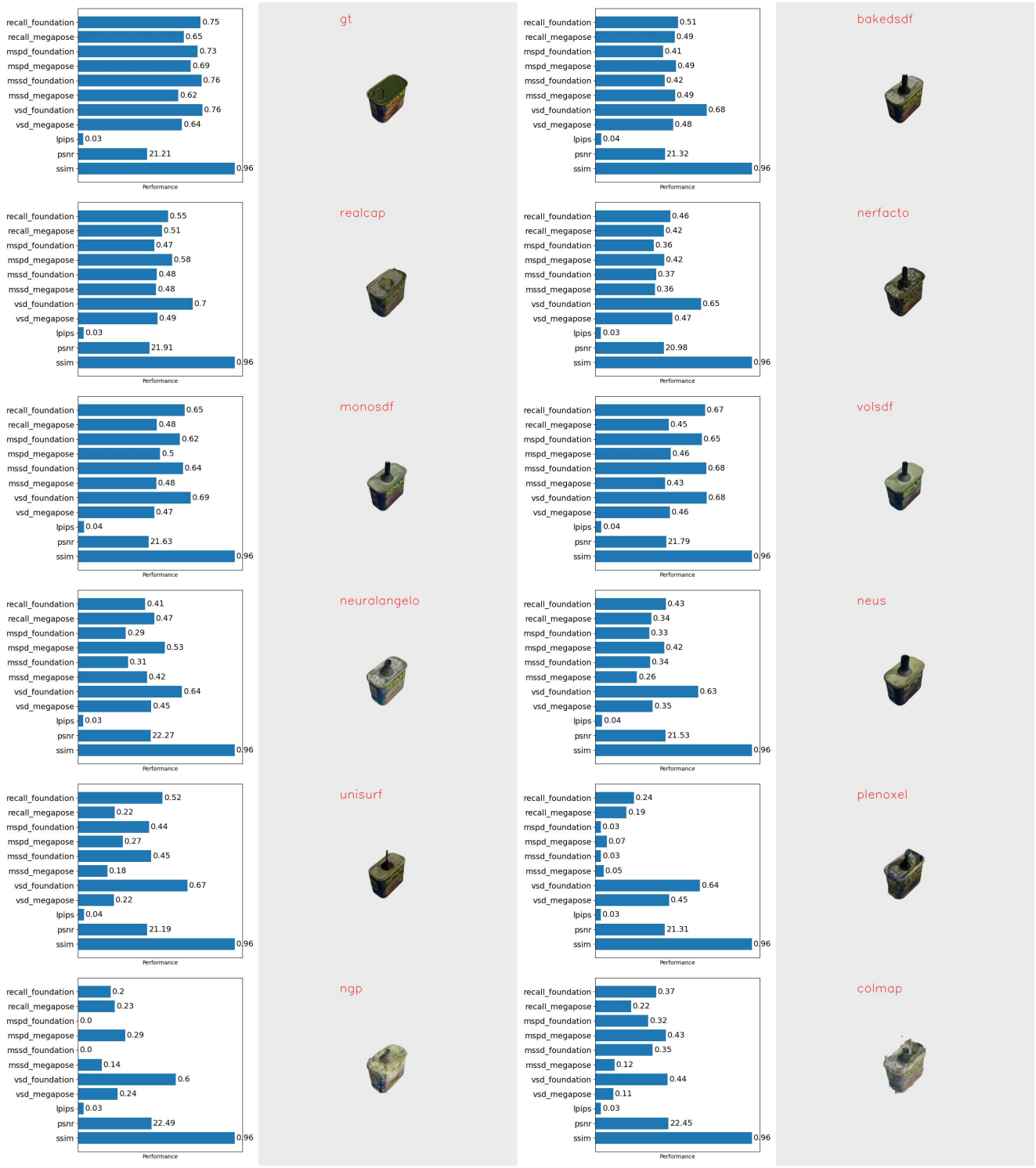

Figure 20. Pose performance and texture scores (left) and object renderings (right) of 09-potted-meat-can reconstructed with various reconstructed methods.

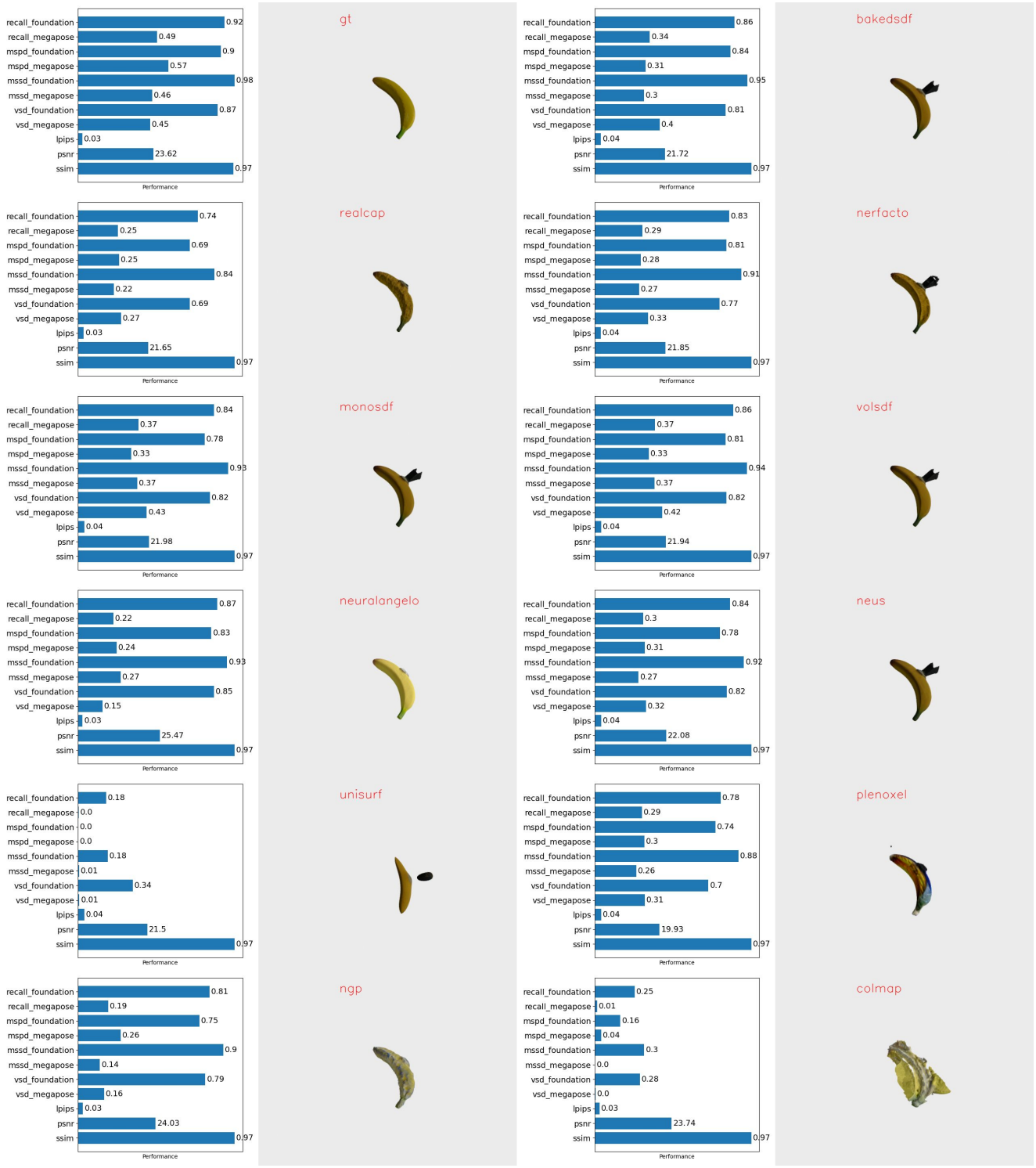

Figure 21. Pose performance and texture scores (left) and object renderings (right) of 10-banana reconstructed with various reconstructed methods.

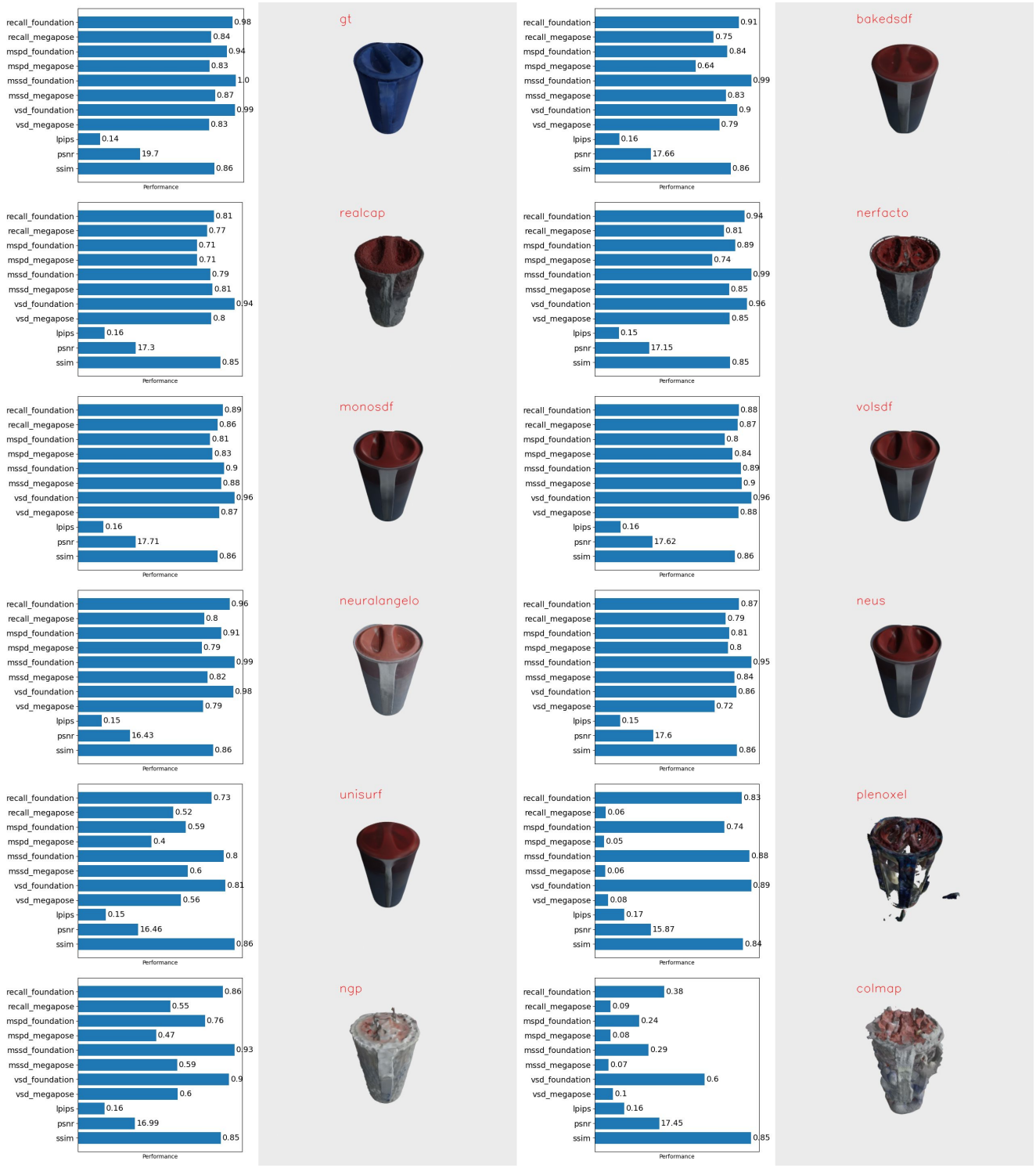

Figure 22. Pose performance and texture scores (left) and object renderings (right) of 11-pitcher-base reconstructed with various reconstructed methods.

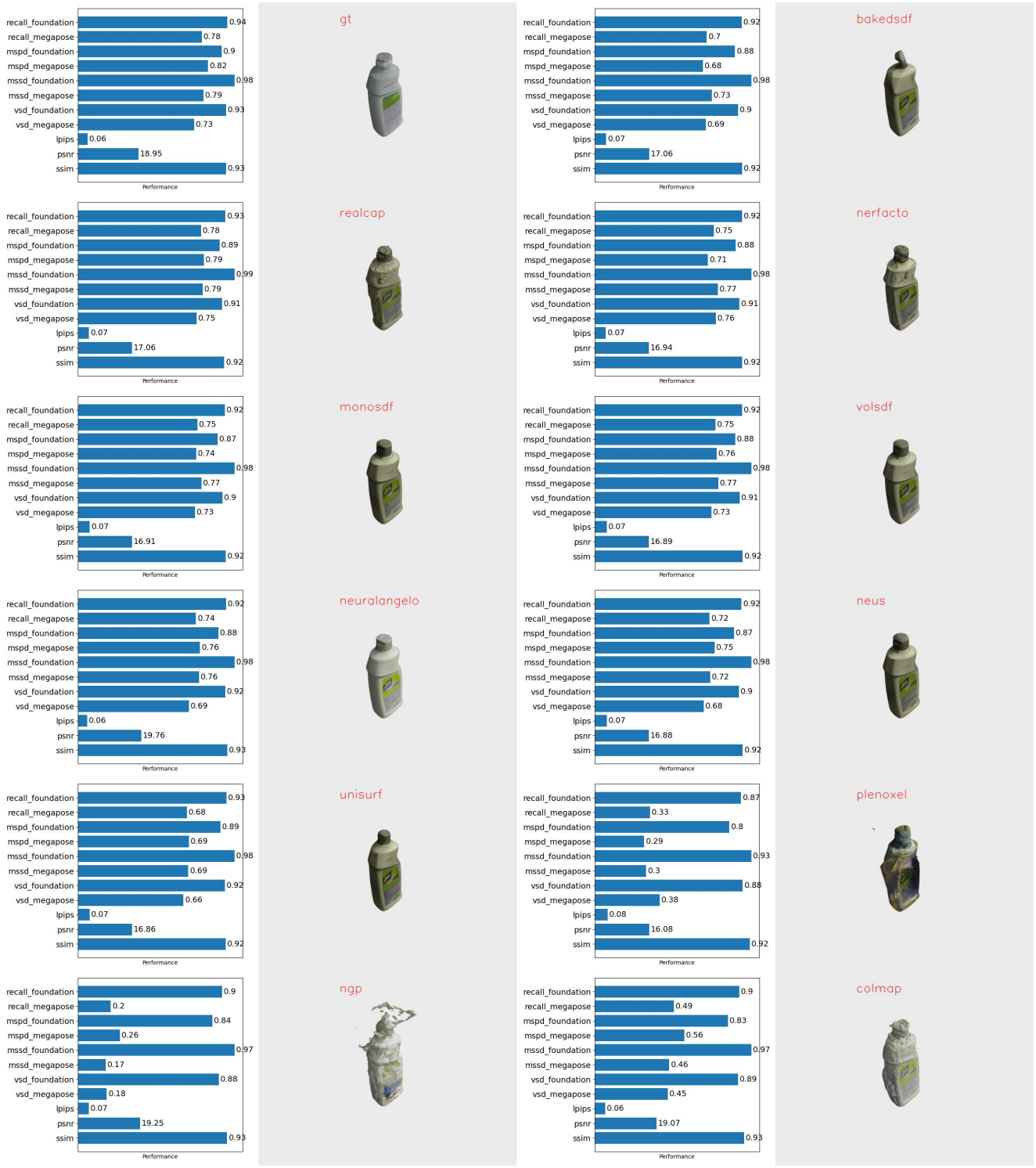

Figure 23. Pose performance and texture scores (left) and object renderings (right) of 12-bleach-cleanser reconstructed with various reconstructed methods.

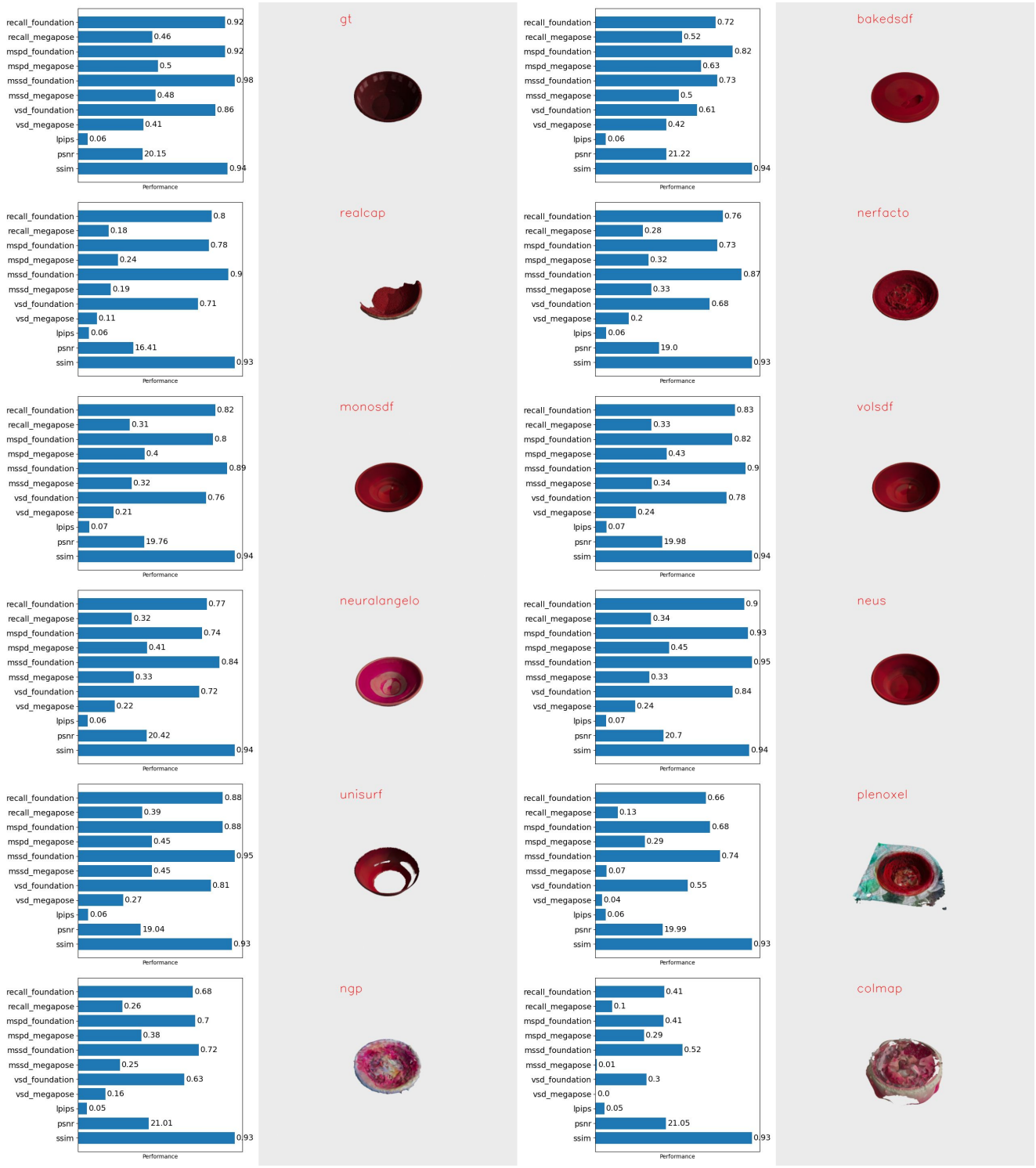

Figure 24. Pose performance and texture scores (left) and object renderings (right) of 13-bowl reconstructed with various reconstructed methods.

Figure 25. Pose performance and texture scores (left) and object renderings (right) of 14-mug reconstructed with various reconstructed methods.

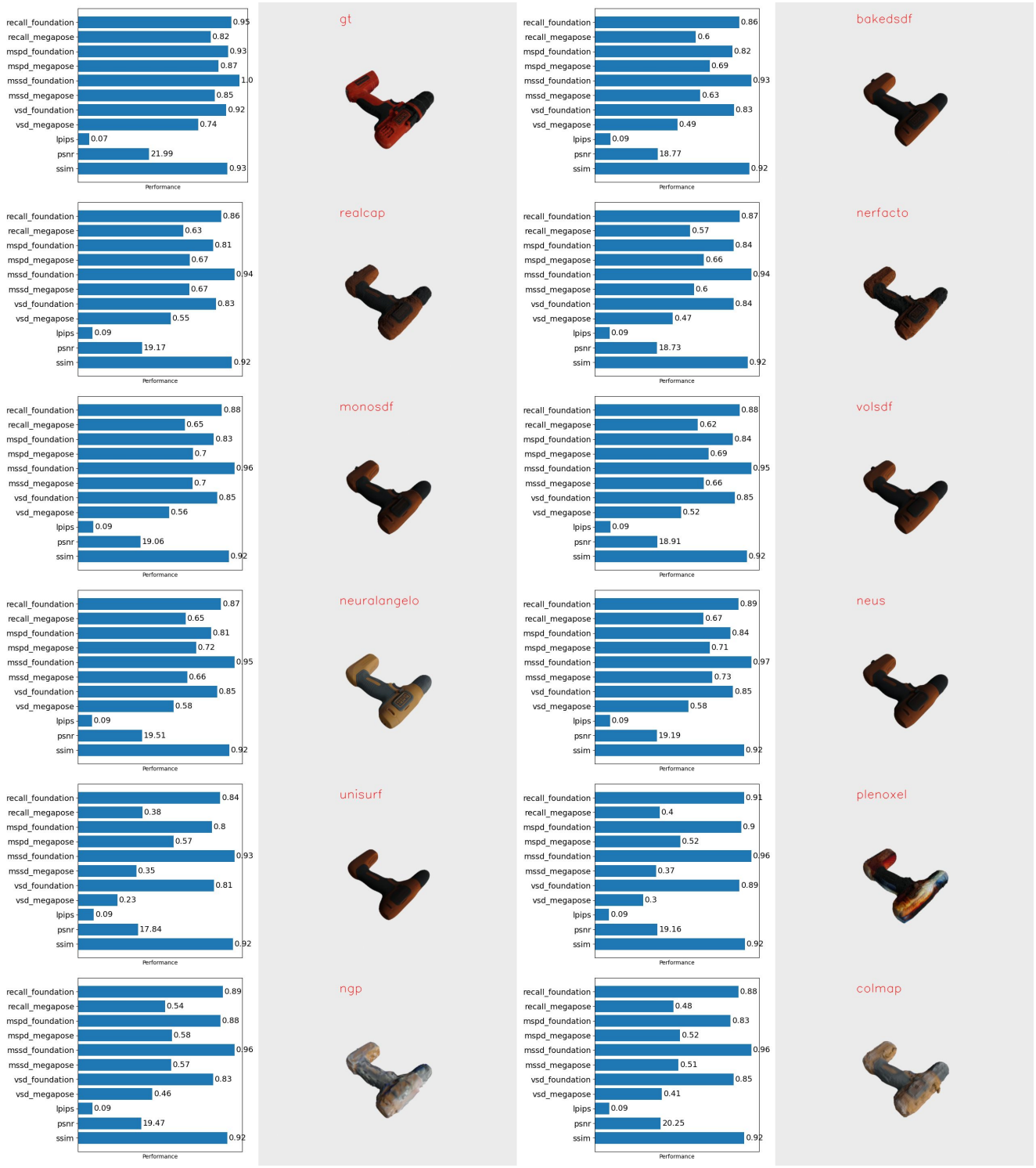

Figure 26. Pose performance and texture scores (left) and object renderings (right) of 15-power-drill reconstructed with various reconstructed methods.

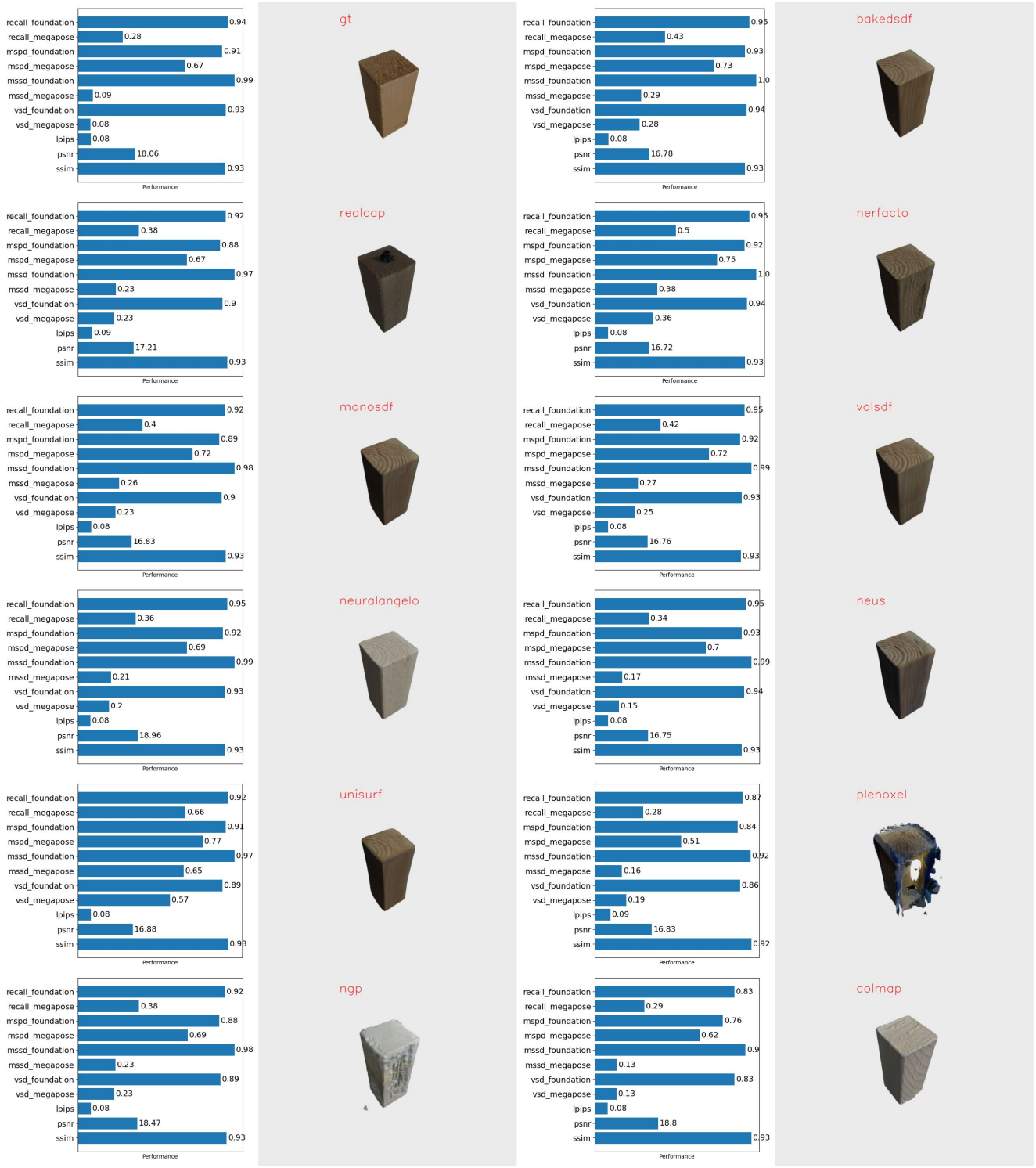

Figure 27. Pose performance and texture scores (left) and object renderings (right) of 16-wood-block reconstructed with various reconstructed methods.

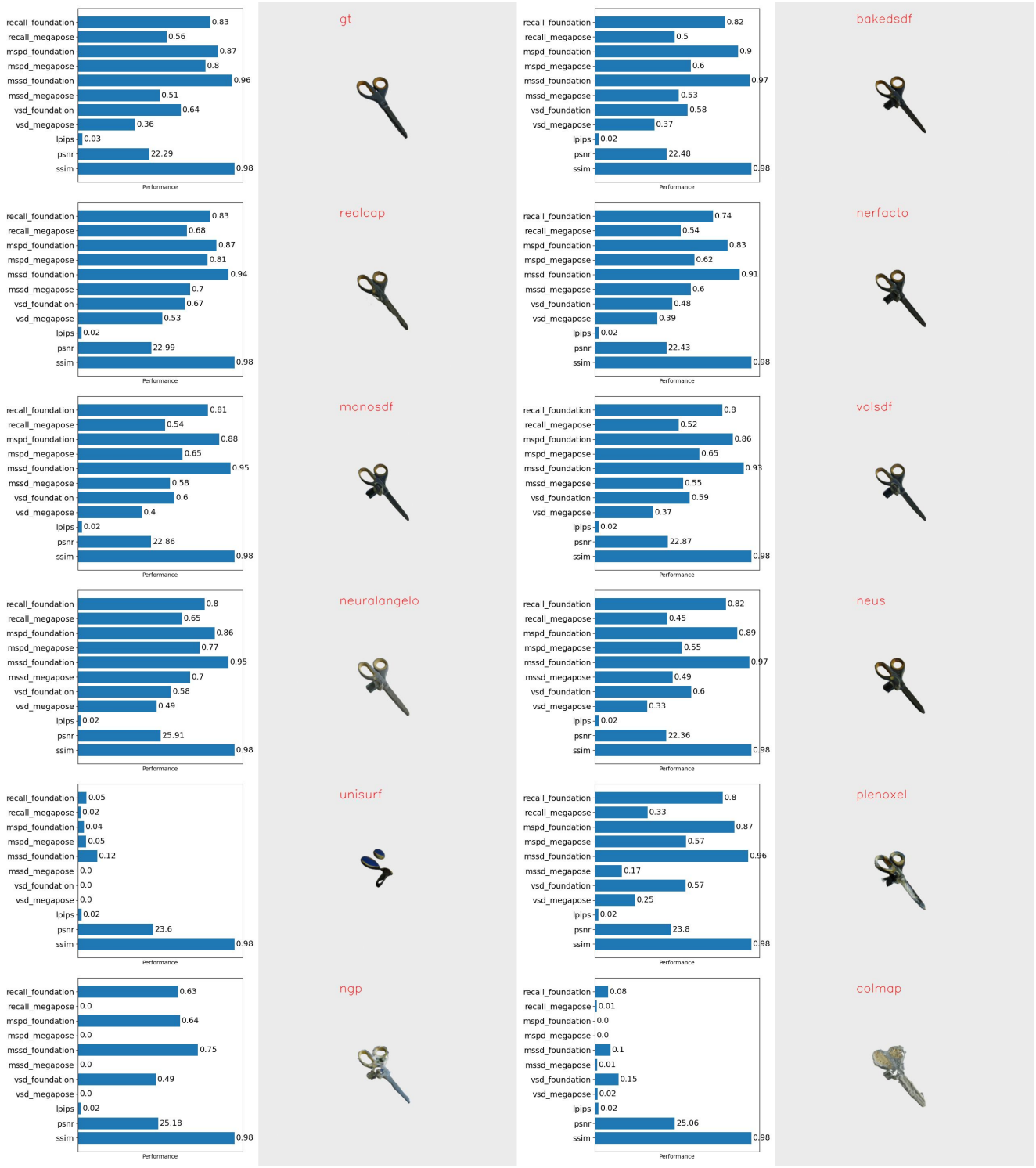

Figure 28. Pose performance and texture scores (left) and object renderings (right) of 17-scissors reconstructed with various reconstructed methods.

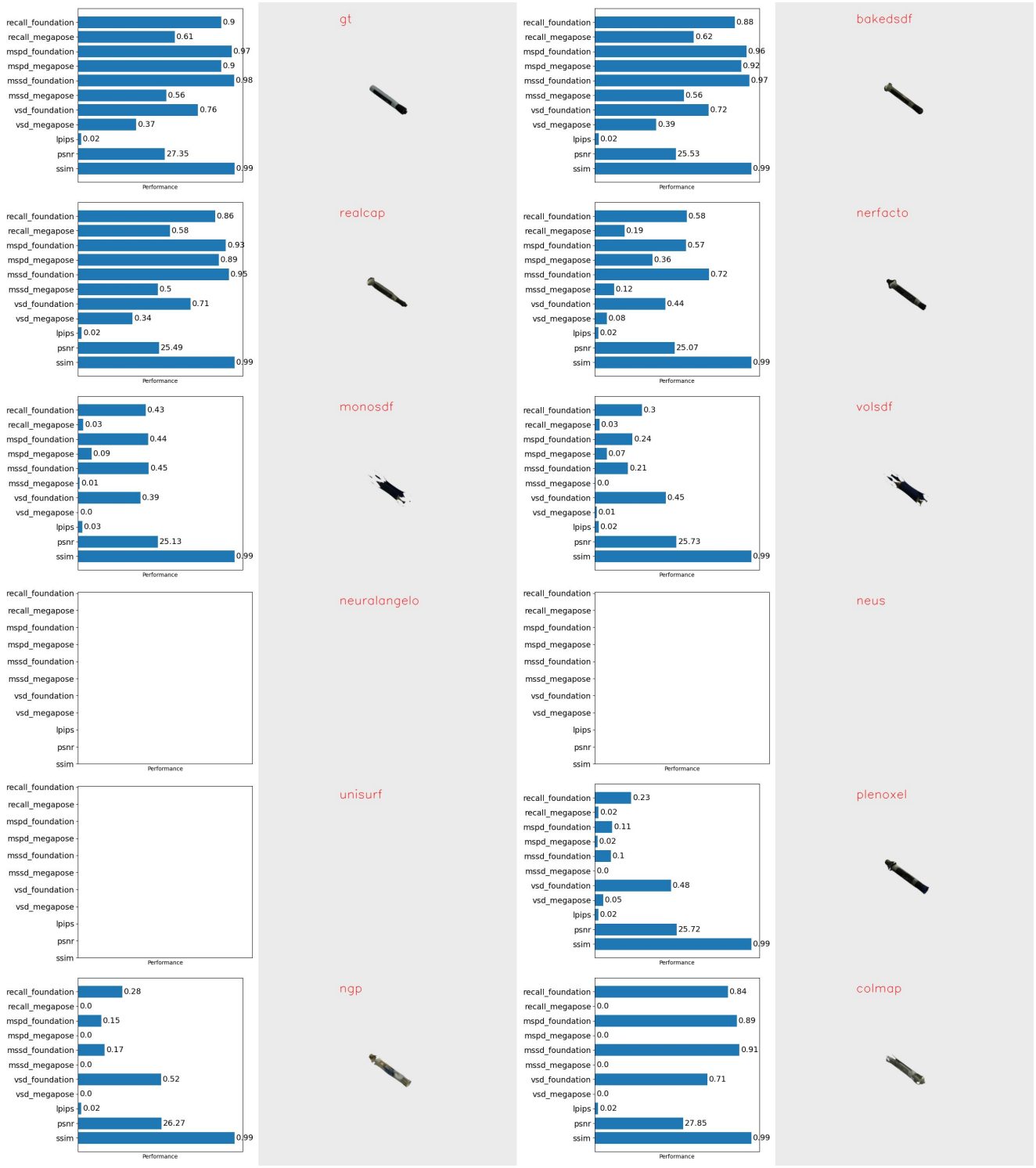

Figure 29. Pose performance and texture scores (left) and object renderings (right) of 18-large-marker reconstructed with various reconstructed methods.

Figure 30. Pose performance and texture scores (left) and object renderings (right) of 19-large-clamp reconstructed with various reconstructed methods.

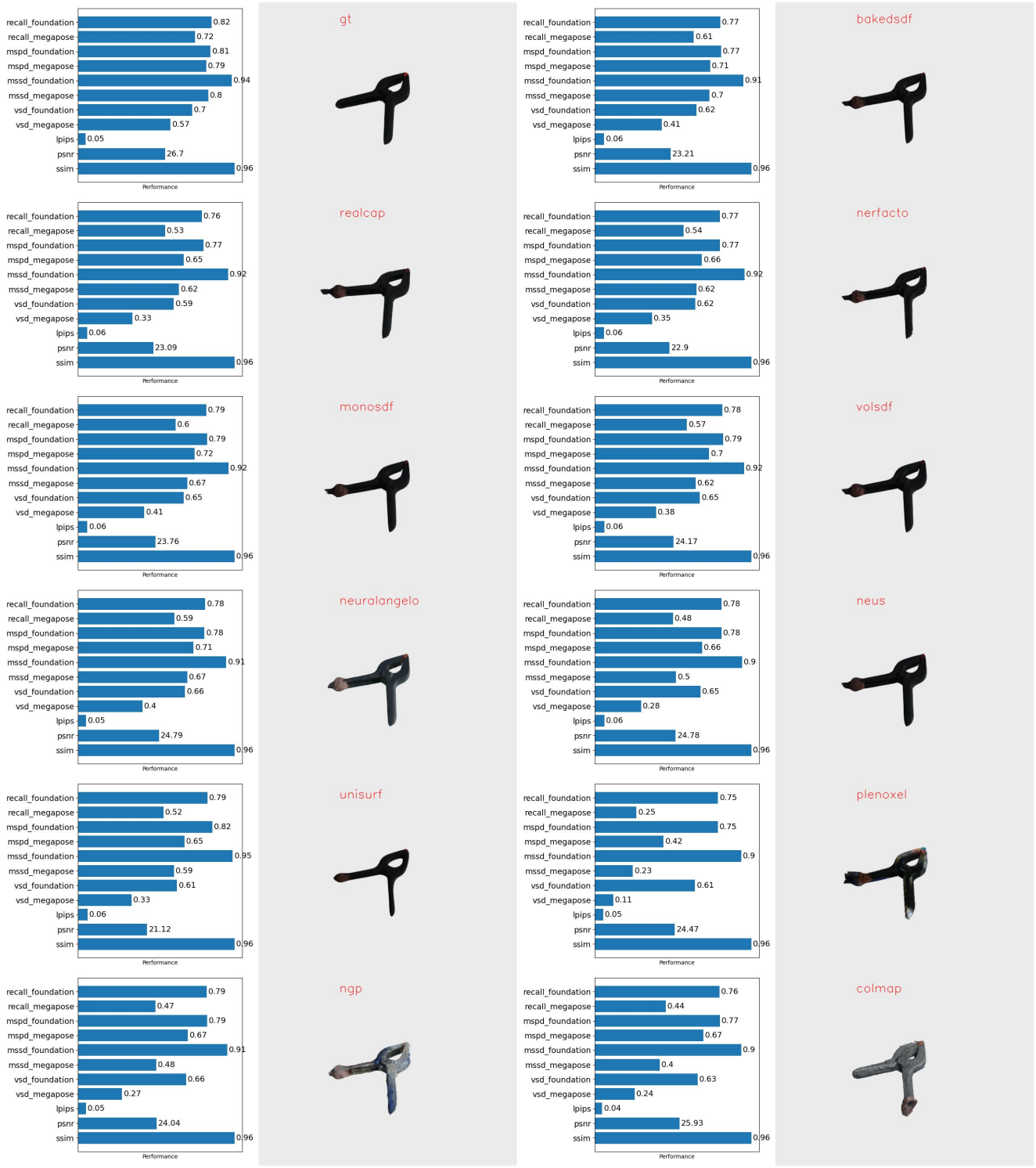

Figure 31. Pose performance and texture scores (left) and object renderings (right) of 20-extra-large-clamp reconstructed with various reconstructed methods.

Figure 32. Pose performance and texture scores (left) and object renderings (right) of 21-foam-brick reconstructed with various reconstructed methods.

References

[1] Bop: Benchmark for 6d object pose estimation. https://bop.felk.cvut.cz/home/. 7

[2] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2911–2918. IEEE, 2012. 3

[3] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. Spie, 1992. 2

[4] Gary Bradski, Adrian Kaehler, et al. Opencv. *Dr. Dobb's journal of software tools*, 3(2), 2000. 1

[5] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015. 1, 3, 5, 6, 7, 10

[6] Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit![ros topics]. *IEEE Robotics & Automation Magazine*, 19(1):18–19, 2012. 2

[7] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. 3

[8] Dawson-Haggerty et al. trimesh. 3

[9] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 4

[10] Álvaro González. Measurement of areas on a sphere using fibonacci and latitude–longitude lattices. *Mathematical Geosciences*, 42:49–64, 2010. 2, 8

[11] Christoph Hennersperger, Bernhard Fuerst, Salvatore Virga, Oliver Zettinig, Benjamin Frisch, Thomas Neff, and Nassir Navab. Towards mri-based autonomous robotic us acquisitions: a first feasibility study. *IEEE transactions on medical imaging*, 36(2):538–548, 2017. 2

[12] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. Bop: Benchmark for 6d object pose estimation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018. 1, 5, 7

[13] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas. Bop challenge 2020 on 6d object localization. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 577–594. Springer, 2020. 1, 5, 7

[14] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 3, 4

[15] Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. MegaPose: 6D Pose Estimation of Novel Objects via Render & Compare. In *CoRL*. 1, 3, 4, 7, 8, 9, 10, 11, 12, 14, 15

[16] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023. 4, 7, 9

[17] 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. 4

[18] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004. 3

[19] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2, 4

[20] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021. 4, 10

[21] Nathan Ratliff, Matt Zucker, J Andrew Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *2009 IEEE international conference on robotics and automation*, pages 489–494. IEEE, 2009. 2

[22] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 3

[23] RealityCapture2023. RealityCapture, 4 2023. 3, 4, 6, 7, 8, 9

[24] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 2, 3, 10

[25] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 501–518. Springer, 2016. 3, 10

[26] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, 2023. 2, 4, 6, 7, 8, 10

[27] Michael Waechter, Nils Moehrle, and Michael Goesele. Let there be color! — Large-scale texturing of 3D reconstructions. In *Proceedings of the European Conference on Computer Vision*. Springer, 2014. 10, 13

[28] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 34:27171–27183, 2021. 4, 9, 10

[29] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. FoundationPose: Unified 6d pose estimation and tracking of novel objects. In *CVPR*, 2024. 1, 3, 4, 7, 8, 9, 10, 11, 14, 15

[30] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018. 1, 2, 3, 5, 6, 7, 8, 9, 10, 14, 15

[31] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021. 4, 7, 10

[32] Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P Srinivasan, Richard Szeliski, Jonathan T Barron, and Ben Mildenhall. Bakedsdf: Meshing neural sdfs for real-time view synthesis. *ACM Transactions on Graphics*, 2023. 4, 7, 8, 10

[33] Zehao Yu, Anpei Chen, Bozidar Antic, Songyou Peng, Apratim Bhattacharyya, Michael Niemeyer, Siyu Tang, Torsten Sattler, and Andreas Geiger. Sdfstudio: A unified framework for surface reconstruction, 2022. 4, 10

[34] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in neural information processing systems*, 35:25018–25032, 2022. 4, 7, 10

[35] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 11

[36] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 2, 3