RIC: Rotate-Inpaint-Complete for Generalizable Scene Reconstruction

Isaac Kasahara, Shubham Agrawal, Selim Engin, Nikhil Chavan-Dafle, Shuran Song, and Volkan Isler

***Abstract*— General scene reconstruction refers to the task of estimating the full 3D geometry and texture of a scene containing previously unseen objects. In many practical applications such as AR/VR, autonomous navigation, and robotics, only a single view of the scene may be available, making the scene reconstruction task challenging. In this paper, we present a method for scene reconstruction by structurally breaking the problem into two steps: rendering novel views via inpainting and 2D to 3D scene lifting. Specifically, we leverage the generalization capability of large visual language models (DALL·E 2) to inpaint the missing areas of scene color images rendered from different views. Next, we lift these inpainted images to 3D by predicting normals of the inpainted image and solving for the missing depth values. By predicting for normals instead of depth directly, our method allows for robustness to changes in depth distributions and scale. With rigorous quantitative evaluation, we show that our method outperforms multiple baselines while providing generalization to novel objects and scenes. Code and data links can be found at https://samsunglabs.github.io/RIC-project-page/.**

I. INTRODUCTION

The understanding of 3D scene geometry is essential for many robotics and augmented reality applications. With smartphones and robots that are equipped with high quality depth sensors, the task of 3D scene reconstruction is becoming feasible in such domains.

These depth sensors allow for accurate reconstruction of the observed parts of the scene. However, to reconstruct the unseen parts, we must use prior information conditioned on the observed information. The missing information in the input image combined with the diversity in shapes, sizes, and depth distribution of the household objects presents a major challenge for scene reconstruction in-the-wild. In this paper, we study this problem in a general setting, where the goal is to reconstruct a complex scene with multiple novel objects, given only one RGB-D image of the scene.

We present our method Rotate-Inpaint-Complete (RIC), which predicts both the 3D geometry and the texture of the unseen parts of the scene in the input image by leveraging the inpainting capabilities of large visual-language models. Given an RGB-D image of a scene, first we generate novel views (RGB and depth images) by rotating and then projecting the input scene. Then we use a surface-aware masking method to select regions in the image to allow us to inpaint utilizing the powerful 2D inpainting capabilities of DALL·E 2 [1] for exposing the potential object geometry not visible in the input image. Finally, we optimize the depth images using the input depth values, as well as the occlusion boundaries and normals estimated from the inpainted images. These inpainted and completed novel RGB-D views provide the reconstructed scene geometry as a fused point cloud with associated textures. To mitigate the object hallucination and spatial inconsistency of predictions from DALL·E 2, we use a consistency filtering method to enforce consistency across viewpoints which plays a crucial role for generalizable, yet accurate and robust scene reconstruction.

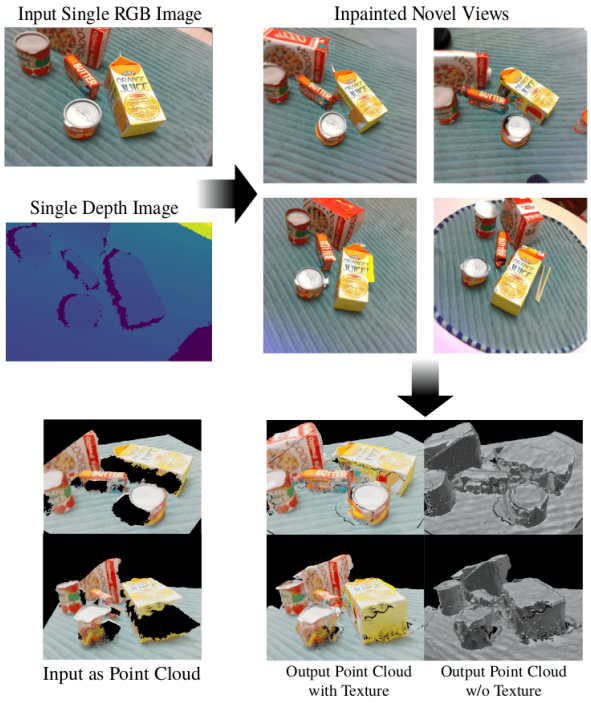

Fig. 1: RIC inputs a single RGB-D image and generates complete 3D scene reconstruction with texture (bottom-left). RIC first **R**otates the input RGB-D image to a novel viewpoint, **I**npaints the missing regions using generalizable visual language models, and finally **C**ompletes the depth via normal prediction and optimization. For comparison, input as a point cloud is also shown on the bottom-left.

In short, the contributions of this paper can be summarized as follows. *i)* We present an integrated approach for scene completion of unseen objects under occlusion and clutter, by solving the problem through novel view inpainting and 2D to 3D scene lifting. *ii)* We develop a method for selectively inpainting regions in the novel views of the input scene that enables synthesis of consistent 2D geometry. *iii)* We train a 2D to 3D lifting method on the YCB-V [2] dataset and demonstrate the generalization capability to cluttered scenes containing novel household objects and categories.

Work done while authors were at Samsung AI Center NY, 837 Washington St, New York, NY 10014

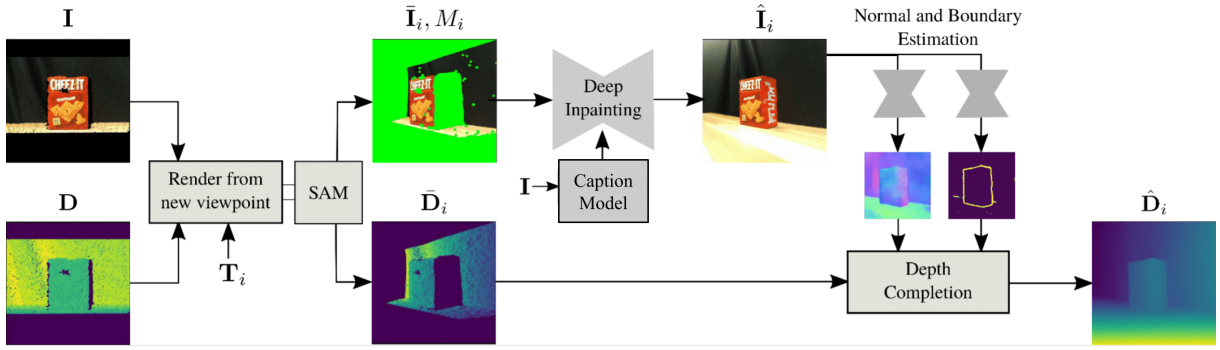

Fig. 2: **Method Overview:** RIC takes as input an RGB-D image and starts by rendering incomplete RGB-D images $\bar{\mathbf{I}}_i$ and $\bar{\mathbf{D}}_i$ from a new viewpoint $\mathbf{T}_i$. The missing RGB values of $\bar{\mathbf{I}}_i$ are inpainted using a diffusion-based VLM given a generated prompt, such as "a photo of household objects on a table", where the pixels to be inpainted are determined by our Surface-Aware Masking (SAM) technique. The inpainted image is used to predict surface normals and occlusion boundaries at the new viewpoint $\mathbf{T}_i$, which are then used for completing the missing depth values along with the incomplete depth image $\bar{\mathbf{D}}_i$. After repeating this process for V viewpoints, the final output of RIC is a merge of deprojected depth predictions.

II. METHOD

RIC takes in as input an RGB-D image $\mathcal{I} = (\mathbf{I}, \mathbf{D}) \in \mathbb{R}^{H \times W \times 4}$ and outputs a color point cloud $S \in \mathbb{R}^{N \times (3+3)}$, where N is the number of predicted points in the scene. Our method consists of three main components: 1) An inpainting step that takes in an RGB-D image $\mathcal{I}$ and outputs an inpainted RGB image $\hat{\mathbf{I}}_i$ from a novel viewpoint $\mathbf{T}_i \in SE(3)$. 2) A depth completion component that takes in the inpainted RGB image $\hat{\mathbf{I}}_i$ as well as an incomplete depth image $\bar{\mathbf{D}}_i$ rendered from the viewpoint $\mathbf{T}_i$, and outputs a completed depth $\hat{\mathbf{D}}_i$ at that viewpoint. 3) A viewpoint selection and consistency filtering method that utilizes the above two components to generate completed RGB-D images at rotated novel views and uses them to reconstruct the scene.

A. Inpainting

1) Rotate and Project RGB-D Image: Given a scene RGB-D image and the camera intrinsics, we deproject the image into a point cloud in the camera frame. This point cloud is then projected onto a novel viewpoint $\mathbf{T}_i$ and the resulting image is masked using our Surface-Aware Masking method (SAM), which we describe in detail in the following section. The projection from this new viewpoint creates a new RGB-D image $\bar{\mathbf{I}}_i$ with missing RGB and depth information as seen in Figure 2. Small holes of the missing RGB values are filled with a naive inpainting algorithm [3] by inpainting pixels that are covered after a morphological closing operation of kernel size 5 is applied to the mask. The larger missing areas are left for the deep inpainting module.

2) Surface-Aware Masking: In order for inpainting to work properly, a mask covering the areas to inpaint needs to be generated. After projecting to the new camera frame, any 3D space possible to be reconstructed needs to be represented as an inpainting mask in the 2D image. This issue can be seen in Figure 3 as the table takes up pixels we may want to fill in with the bottle. To solve this problem, a 3D frustum is generated from the original camera and depth image. For every pixel in the original camera frame, a ray is cast from the camera through each point in the projected point cloud from $\mathcal{I}$. Once the ray has passed through its respective point, it is used to generate a list of points along the ray from that depth onward with m points of equal spacing c. This is done for every ray, and from this process results a point cloud covering the potential space that the 3D scene could possibly fill. This point cloud is then converted to a mesh, and when the point cloud from the RGB-D image $\mathcal{I}$ is rotated to novel views, the mesh is rotated with it. Finally, when projecting back to the camera frame after rotation, points that are occluded by the mesh are discarded. Any blank pixels are then used as the 2D inpainting mask to be filled when passed to the inpainting step. This procedure of generating the final image and mask is detailed in the appendix and its outputs are shown in Figure 2, with the green pixels representing the inpainting mask.

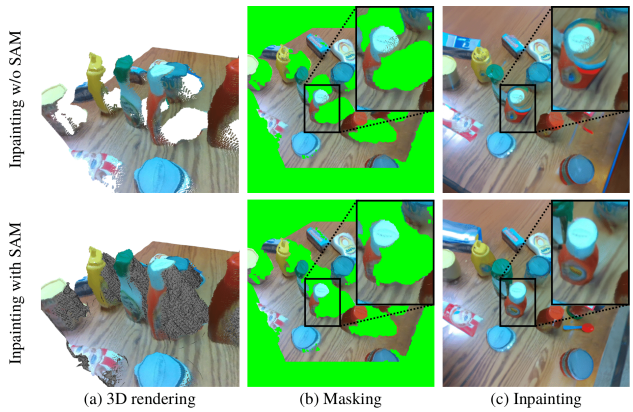

Fig. 3: Surface-Aware Masking (SAM) is a necessary step to obtain realistic inpaintings. Naively rotating the input point cloud moves the background pixels next to the foreground object pixels (b-top) which results in poor inpainting (c-top). Using SAM, we correctly mask out the background pixels which results in good inpainting results (b-bottom).

3) Diffusion-based Inpainting: We use DALL·E 2 [1] for image inpainting as it produces most realistic results. This model takes in the incomplete image $\bar{\mathbf{I}}_i$, the mask generated in the previous step M, and an input prompt P that describes the context of the image in words. For prompt, we pass the

RGB image $\mathbf{I}$ to a deep captioning model [4] and prefix the generated caption with *"A photo of"*. We also explore using a more specific and generic prompt in our ablation experiments (Table IV). The output from this inpainting method is an image $\hat{\mathbf{I}}_i$ that now contains estimated areas from the diffusion model. Figure 2 shows an example before and after inpainting with DALL·E 2.

B. Depth Completion

We use a method proposed in [5] for generating a complete depth image $\hat{\mathbf{D}}_i$ from an incomplete depth image $\bar{\mathbf{D}}_i$ and its corresponding RGB image. This method estimates the normals and occlusion boundaries from the RGB image, and optimizes for the complete depth by utilizing the estimated normals, occlusion boundaries, and incomplete depth. In order to obtain estimations for the normals and occlusion boundaries, we train Deeplabv3+ with DRN-D-54 in the same manner as in [6] using the the YCB-V training dataset [2], the YCB-V synthetic dataset [7], [8], and the HomebrewedDB synthetic dataset [9]. Given the incomplete depth, the estimated normals from the image, and estimated occlusion boundaries, we solve for the completed depth using the method described in [5]. Please see appendix for details.

C. Scene Completion

This section describes the complete process we follow to reconstruct a 3D scene from a single RGB-D image.

1) Viewpoint Selection: For diffusion-based inpainting, "known" pixels, i.e., the non-masked areas, guide the prediction of the unknown masked areas. We refer to the known pixels as context pixels and define the context ratio C for any given image as $C = (\#context\ pixels) / (\#all\ pixels)$. This ratio gives us some indication about how accurately the inpainting model will be able to fill in the missing areas. With a low C, many areas are unknown and inpainting will struggle, and with a high C inpainting will do well but only fill in minimal information. An example of different context ratio values can be seen in the appendix Figure 7.

We then design our viewpoint selection process to search for a context ratio that will allow for accurate inpainting. To do this, we define a sphere with a center as the mean of the input point cloud, and the radius as the distance between the center and the initial camera location. Then from the starting viewing angle, we rotate in various directions along this sphere away from the starting position. At each step in this rotation, we compute the context C of the projected input image. If the C is closest to our chosen context threshold C^*, we use that viewpoint $\mathbf{T}_i$ as next to inpaint. We repeat this process for V evenly spaced directions we traverse along the sphere as visualized in Figure 7, where both C^* and V are chosen using the experiment described in Section III.

2) Enforcing Consistency Across Viewpoints: The final step in our method involves combining these generated viewpoints while enforcing consistency across them. One drawback of utilizing DALL·E 2 for inpainting real objects, is its inconsistent completion of objects as well as the hallucination of objects that are not originally in the scene.

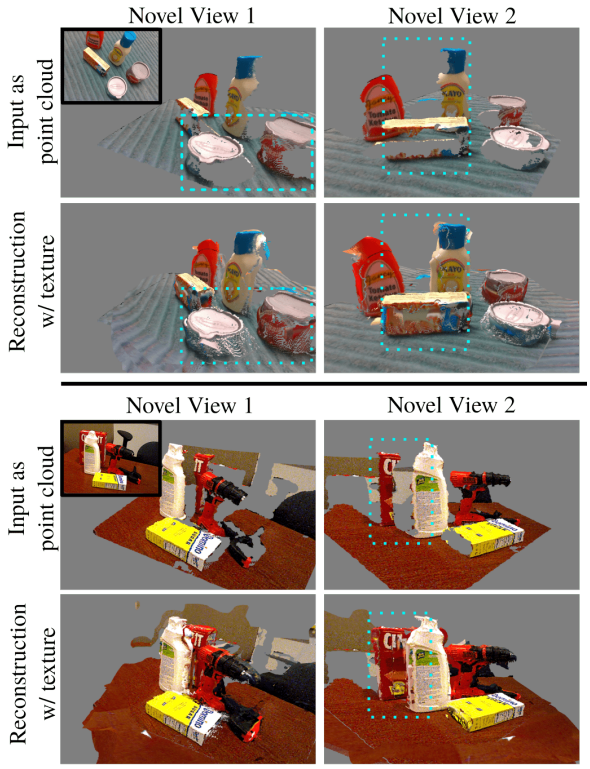

Fig. 4: **Qualitative Results:** We show our scene completion results given a single RGB-D image, as color point clouds from two viewpoints. Top two rows are from the HOPE dataset [10], and the bottom two are from YCB-V [2].

To combat this issue, we filter for consistent predictions across viewpoints. The final prediction is achieved by first deprojecting the RGB-D images from each viewpoint $\mathbf{T}_i$ back into the original camera frame as point clouds. We then apply the following *consistency rule* across all the generated points: If a predicted point from one viewpoint has a predicted point within a 1cm radius from at least two other viewpoints we keep that point, otherwise we remove that point from our final prediction. This rule allows us to keep points that only multiple viewpoints predict. We then combine all filtered points to obtain our final output point cloud of the completed scene S, which contains more accurate geometry and color than without filtering as seen in Table III and Figure 6.

III. EXPERIMENTS

In this section, we evaluate the performance of RIC for single view RGB-D scene reconstruction task. Please see appendix for ablation studies.

Implementation Details: The inpainting step of our algorithm is based on OpenAI's DALL·E 2 API. For our implementation of SAM, we choose a spacing value c of 0.01 meters with m = 100 points for generating our rays. Via grid-search on 4 held out validation scenes from the YCB-V test set, we chose 10 views as the number of viewpoints / viewpoint directions V and 0.4 as the context threshold C^* (see appendix Table II). For our module that enforces consistency between synthesized views, we choose

Method	IoU ↑	F-Score ↑	CD(S^*, S) ↓	CD(S, S^*) ↓	CD ↓
YCB-V [2]					
CON [11]	0.087	0.354	0.036	0.014	0.050
ShellNet [12]	0.224	0.607	0.019	0.012	0.031
CenterSnap [13]	0.225	0.622	0.019	**0.009**	**0.028**
RIC (Ours)	**0.294**	**0.661**	**0.018**	0.010	**0.028**
HOPE [10]					
CON [11]	0.086	0.279	0.094	0.035	0.128
ShellNet [12]	0.185	0.523	0.035	0.013	0.047
CenterSnap [13]	0.180	0.526	0.037	0.006	0.042
RIC (Ours)	**0.290**	**0.649**	**0.031**	**0.005**	**0.036**

TABLE I: Comparison of methods for the task of 3D scene completion on YCB-V [2] and HOPE [10]. Higher numbers for the IoU and F-score metrics, and lower numbers for the Chamfer Distances (CD) indicate better performance.

a threshold of 0.01 meters when computing the intersection between points in the viewpoints point clouds.

Datasets: We trained our depth completion model using the YCB-V training dataset [2]. For testing, we test on 8 unseen scenes from the YCB-V test set, and select 5 RGB-D images from each of the scenes. For ground truth point clouds, we concatenate point-clouds from (a) deprojected RGB-D frames of the scene, and (b) point-clouds of the ground-truth object meshes. Finally, we crop this point cloud around the ground truth meshes with a 10cm buffer as the RGB-D frames may contain floors and walls far away that we are not interested in reconstructing. This creates our final ground truth point cloud covering the majority of the scene with full geometry of the objects in the scene.

To demonstrate our model's capabilities of generalizing to unseen objects and to entirely new datasets, we also compare our method on the HOPE dataset [14]. HOPE test set only contains individual RGB-D images and is unusable for generating full scene point cloud. Instead, we use HOPE training dataset for evaluation, as the train set contains RGB-D video and cluttered tabletop scenes with novel objects. The dataset has 10 scenes, and we again sample 5 frames per scene. Ground truth point clouds are obtained similarly to the YCB-V dataset.

Baselines: We compare RIC against four baselines: Convolutional Occupancy Networks (CON) [11], CoReNet [15], ShellNet [12], and CenterSnap [13] (see Appendix for baseline implementation details).

Table I shows quantitative evaluations on within-training-distribution YCB-V dataset [2] and out-of-training-distribution HOPE dataset [14]. On YCB-V dataset, RIC is able to outperform CON and ShellNet on all 3D scene reconstruction metrics. CON takes as input a sparse point-cloud of the scene. When major parts of the input point clouds are missing, as the common case for single-view RGB-D point clouds, CON fails to infer those regions. ShellNet is trained to predict back-side depth image for the detected object. We notice that with varying viewing directions, ShellNet backside depths are either too thin or too thick resulting in low performance. MaskRCNN's failure to detect objects also directly contributed to lower performance for ShellNet. CenterSnap inputs RGB-D image and predicts object shapes via a multi-step procedure allowing CenterSnap to learn strong shape and pose priors for objects within training distribution. This allowed CenterSnap to perform strongly on YCB-V objects as it was trained on them, but we noticed it struggles with cases of occluded objects. RIC which is trained without ground truth object pose or shape supervision is able to match or outperform the baselines in all metrics. Fig. 5 shows a qualitative comparison with baselines.

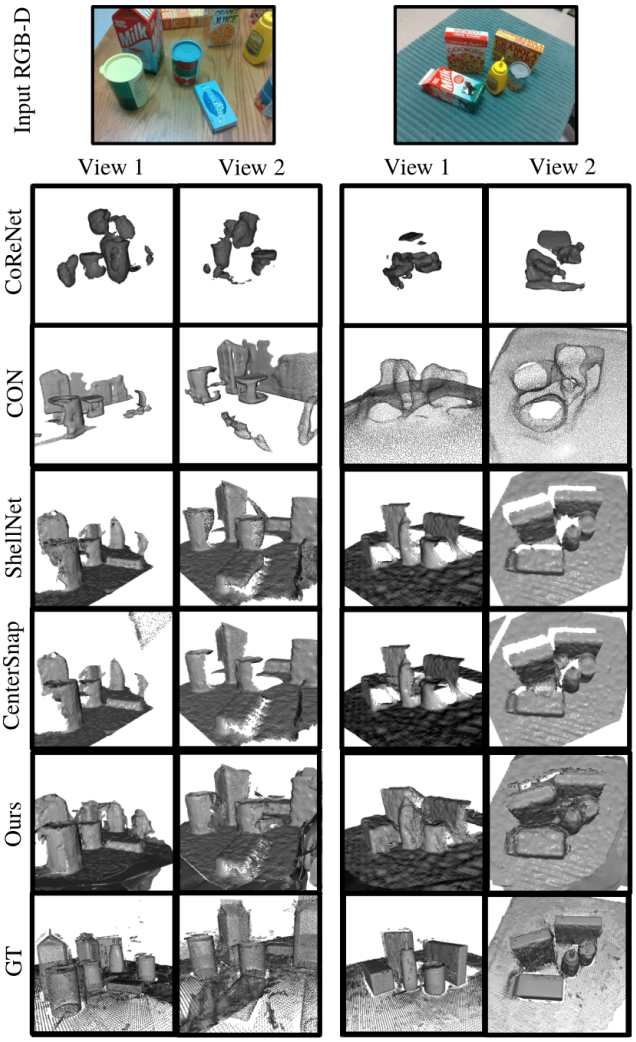

Fig. 5: **Qualitative Comparison:** We compare our method against baselines for completing scene geometry given a single RGB-D image. View 1 and 2 show novel viewpoints of the predicted point clouds from each method. Our method provides a denser and more complete reconstruction.

On the out-of-distribution HOPE dataset, RIC is able to outperform all baselines by an even larger margin. This shows that our normal and occlusion boundary-based depth completion method generalizes well to unseen novel scenes. Figure 4 shows qualitative results on these datasets.

IV. DISCUSSION

We presented RIC, a novel method for 3D scene reconstruction. RIC solves the problem of 3D reconstruction of a cluttered scene of novel objects by leveraging the generalization capabilities of large visual language models.

REFERENCES

[1] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[2] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[3] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 9(1):23–34, 2004.

[4] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

[5] Yinda Zhang and Thomas Funkhouser. Deep depth completion of a single rgb-d image. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] Shreeyak Sajjan, Matthew Moore, Mike Pan, Ganesh Nagaraja, Johnny Lee, Andy Zeng, and Shuran Song. Clear grasp: 3d shape estimation of transparent objects for manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3634–3642. IEEE, 2020.

[7] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Dmitry Olefir, Tomas Hodan, Youssef Zidan, Mohamad Elbadrawy, Markus Knauer, Harinandan Katam, and Ahsan Lodhi. BlenderProc: reducing the reality gap with photorealistic rendering. *Robotics: Science and Systems (RSS) Workshops*, 2020.

[8] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas. BOP challenge 2020 on 6D object localization. *European Conference on Computer Vision Workshops (ECCVW)*, 2020.

[9] Roman Kaskman, Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[10] Yunzhi Lin, Jonathan Tremblay, Stephen Tyree, Patricio A. Vela, and Stan Birchfield. Multi-view fusion for multi-level robotic scene understanding. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6817–6824, 2021.

[11] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 523–540. Springer, 2020.

[12] Nikhil Chavan-Dafle, Sergiy Popovych, Shubham Agrawal, Daniel D Lee, and Volkan Isler. Simultaneous object reconstruction and grasp prediction using a camera-centric object shell representation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1396–1403. IEEE, 2022.

[13] Muhammad Zubair Irshad, Thomas Kollar, Michael Laskey, Kevin Stone, and Zsolt Kira. Centersnap: Single-shot multi-object 3d shape reconstruction and categorical 6d pose and size estimation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10632–10640. IEEE, 2022.

[14] Stephen Tyree, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Jeffrey Smith, and Stan Birchfield. 6-dof pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark. In *International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[15] Stefan Popov, Pablo Bauszat, and Vittorio Ferrari. Corenet: Coherent 3d scene reconstruction from a single rgb image. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 366–383. Springer, 2020.

[16] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019.

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

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

APPENDIX

Here we include further details about implementation and experiments.

A. Surface-Aware Masking Pseudocode

We include pseudocode to help explain how our Surface-Aware Masking module (SAM) is implemented.

Algorithm 1 SURFACE-AWARE MASKING (SAM)

Require: Input RGB-D image $\mathcal{I} = (\mathbf{I}, \mathbf{D})$, intrinsics $\mathbf{K}$, new viewpoint $\mathbf{T}_i$

$U \leftarrow$ Subsample pixels from a uniform grid in $\mathcal{I}$

$X \leftarrow \{\}$ ▷ initialize an empty point set.

for all $\mathbf{u} \in U$ **do**

$\mathbf{x} \leftarrow \mathbf{D}(\mathbf{u})\mathbf{K}^{-1}\mathbf{u}$ ▷ deprojection of $\mathbf{u}$ to 3D point $\mathbf{x}$.

for $i \leftarrow 1$ to m **do**

$\mathbf{p} \leftarrow \mathbf{x} + i \cdot c \cdot \mathbf{K}^{-1}\mathbf{u}$

$X \leftarrow X \cup \{\mathbf{p}\}$ ▷ set of points with equal spacing.

$\mathcal{M} \leftarrow$ Mesh(X) ▷ surface triangulation to create a mesh.

$\bar{\mathbf{I}}_i, \bar{\mathbf{D}}_i \leftarrow$ Reprojection of $\mathbf{I}, \mathbf{D}$ in camera viewpoint $\mathbf{T}_i$, where missing values are set to 0.

$\widetilde{\mathbf{D}}_i \leftarrow$ Depth map rendering of $\mathcal{M}$ in camera $\mathbf{T}_i$

$M \leftarrow \mathbf{0}_{H \times W}$ ▷ initialize the mask image as zeros.

for all $\mathbf{u} \in M$ **do**

$M(\mathbf{u}) \leftarrow 1$ if $\bar{\mathbf{D}}_i(\mathbf{u}) = 0 \vee \bar{\mathbf{D}}_i(\mathbf{u}) > \widetilde{\mathbf{D}}_i(\mathbf{u})$

return $M, \bar{\mathbf{D}}_i$

B. Depth Completion

We use a method proposed in [5] for generating a complete depth image $\hat{\mathbf{D}}_i$ from an incomplete depth image $\bar{\mathbf{D}}_i$ and its corresponding RGB image. This method estimates the normals and occlusion boundaries from the RGB image, and optimizes for the complete depth by utilizing the estimated normals, occlusion boundaries, and incomplete depth.

Normals and Occlusion Boundaries Prediction In order to obtain estimations for the normals and occlusion boundaries, we train Deeplabv3+ with DRN-D-54 in the same manner as in [6]. The ground truth normals and occlusion boundaries are obtained using the depth images from the YCB-V training dataset [2], the YCB-V synthetic dataset [7], [8], and the HomebrewedDB synthetic dataset [9].

Optimize for Depth Given the incomplete depth, the estimated normals from the image, and estimated occlusion boundaries, we solve for the completed depth. The main idea behind this method in [5] is that the areas with missing depth can be computed by tracing along the estimated normals from areas of known depth with the occlusion boundaries acting as barriers where normals should not be traced across. Formally we solve a system of equations to minimize an error E, where E is defined as $E = \lambda_D E_D + \lambda_S E_S + \lambda_N E_N B$. Here, E_D is the distance between the ground truth and estimated depth, E_S influences nearby pixels to have similar depths, E_N measures the consistency of estimated depth and estimated normal values, and B weights the normal values based on the probability that it is a boundary. We use the same $\lambda_D, \lambda_S, \lambda_N$ values as in [6].

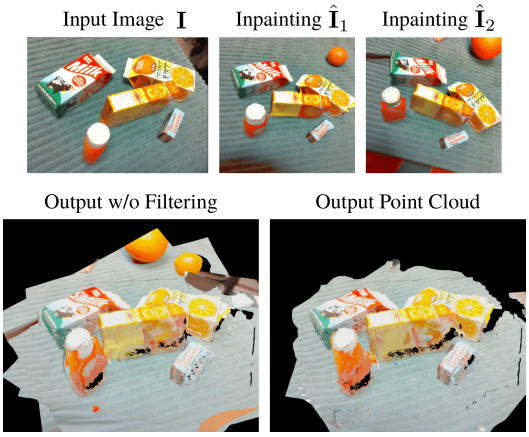

Fig. 6: The consistency filtering step is used to remove the hallucinated objects in 3D, e.g., the oranges in the top right of the images get filtered out. For simplicity we visualize our consistency filtering step for only two viewpoints, while we filter using all viewpoints in our main method.

C. Metrics

We also include additional information about the metrics we use for quantitative results in our paper:

Intersection-over-Union (IoU): We voxelize the ground truth and predicted point clouds at a fixed resolution and compute the IoU score by dividing the number of voxels that intersect to that of their union. In our experiments, we evaluate all the methods at the same grid resolution of 100^3 after rescaling the predictions and ground truth to fit into the unit cube. **Chamfer Distance (CD)**: Chamfer distance is commonly used to measure the similarity between two point sets and is defined as:

$$CD(X, Y) = \frac{1}{|X|} \sum_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y} ||\mathbf{x} - \mathbf{y}||_2 \quad (1)$$

We separately report $CD(S, S^*)$ and $CD(S^*, S)$, as well as their their sum. $CD(S, S^*)$ measures how close the reconstructed points from S are to the ground truth points S^*, whereas $CD(S^*, S)$ computes how well the ground truth shape is covered. **F-Score**: Following [16], we also report F-Score@1% which is a measure for the percentage of the surface points that were reconstructed correctly.

D. Parameter Grid Search Experiment

For choosing the number of viewpoints/viewpoint directions V as well as the context ratio C^* described in our method section, we perform a parameter search using 4 held out validation scenes from the YCB-V test set. We test using 6, 8, 10, and 12 viewpoints as well as a value of 0.3, 0.4, 0.5, 0.6, and 0.7 for our context threshold. We found that 10 views and 0.4 as a context threshold gave us the best accuracy on the validation set. 12 views and 0.4 as a context threshold performed similarly, but in the interest of runtime we use 10 for the final method. Table II shows the full results from this experiment.

V | C^*	0.3	0.4	0.5	0.6	0.7
6	0.064	0.053	0.051	0.057	0.064
8	0.057	0.048	0.050	0.059	0.066
10	0.053	**0.047**	0.052	0.059	0.070
12	0.052	**0.047**	0.052	0.063	0.071

TABLE II: Experiment using different values for context C^* and number of viewpoints V for our method on 4 validation scenes of the YCB-V [2] dataset using Chamfer Distance to indicate better performance.

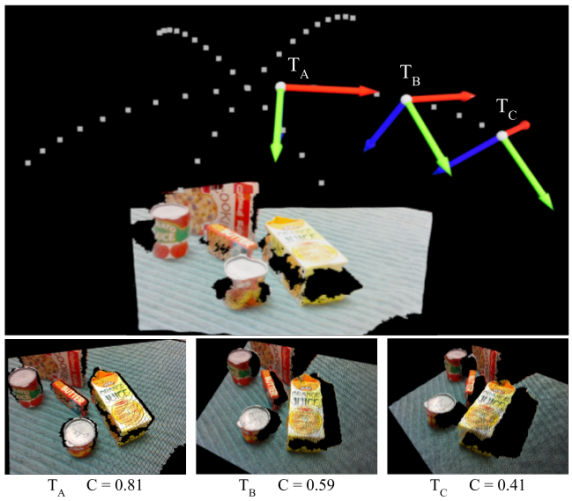

Fig. 7: RIC samples viewpoints on evenly spaced directions along the viewing sphere (white dots). Several viewpoints (top) as well as their corresponding rendered images are shown together with the context ratio of each image (bottom).

E. Baseline Implementation Details

We compare RIC against four baselines: Convolutional Occupancy Networks (CON) [11] is a 3D scene reconstruction method that inputs a sparse point cloud. We use their pre-trained model for *Synthetic Indoor Scene dataset* where similar to our YCB-V and HOPE datasets, they place multiple ShapeNet [17] objects in indoor scenes. CoReNet [15] is a multi-object shape estimator that inputs an RGB image and estimates a mesh. We compare against CoReNet's pre-trained model qualitatively since its predictions lack scale information. ShellNet [12] is trained for single object reconstruction. Given a scene depth image and object instance mask, ShellNet produces reconstruction for the object instance. We re-implemented ShellNet's architecture and trained it with Mask R-CNN [18] as the segmentation network on YCB-V dataset.

Finally, we compare against CenterSnap [13], a multi-object point cloud prediction method. CenterSnap inputs an RGB-D image and predicts point clouds for each object in the scene. Similar to CenterSnap's original training, we first train it on YCB-V synthetic dataset [7], [8], then fine-tune it on the YCB-V real training dataset [2]. Since CenterSnap and ShellNet only predict the point clouds for objects and not the rest of the scene, for a fair evaluation, we concatenate their outputs with deprojected point cloud from the input RGB-D image.

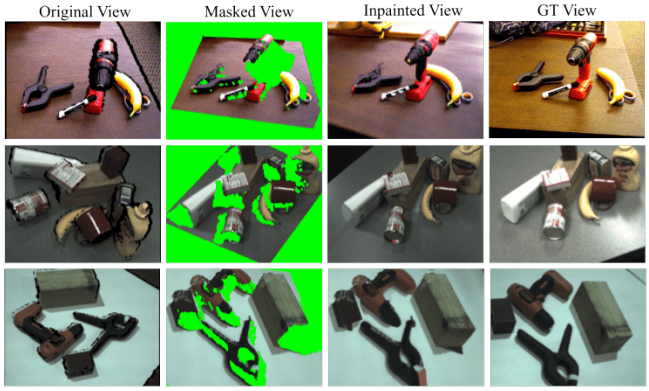

Fig. 8: **Qualitative Novel View Results:** As a byproduct of our method we also show qualitative results for generating novel views of scenes from the the YCB-V dataset.

Method	IoU ↑	F-Score ↑	CD(S^*, S) ↓	CD(S, S^*) ↓	CD ↓
RIC (SD-2)	0.265	0.620	0.033	**0.005**	0.037
RIC (No Filter)	0.271	0.574	**0.017**	0.030	0.047
RIC (Ours)	**0.290**	**0.649**	0.031	**0.005**	**0.036**

TABLE III: Result of swapping out various parts of our method shown on the HOPE [10] dataset. (Ours) utilizes OpenAI's DALL·E 2 model and our consistency filtering method, (SD-2) uses Stable Diffusion 2's inpainting model, and (No Filter) refers to our method without filtering.

F. Ablation Studies

Method	IoU ↑	F-Score ↑	CD ↓
RIC (S)	0.262	0.613	0.038
RIC (G)	0.261	0.613	0.038
RIC (Ours)	**0.290**	**0.649**	**0.036**

TABLE IV: Prompt specificity results on HOPE dataset [14]: RIC (S) denotes our model with scene specific prompt, RIC (G) uses "household objects on a table" as the prompt for all scenes, and RIC (Ours) uses an image caption generator [4].

Prompt Reliance: Image diffusion models tend to heavily rely on the input prompt. To test our methods robustness, we performed an experiment using a general prompt (G), "a photo of household objects on a table", for every scene to see how much performance degrades. We also use a specific prompt (S) where using the ground truth list of objects we list out every object on the table as the prompt. Table IV shows that our method does not largely depend on the type of prompt. We hypothesize that our view selection method retains enough surrounding context information in the input RGB image required for the inpainting model to inpaint successfully.

Inpainting Model: We substitute in Stable Diffusion 2's [19] inpainting model as an open-source alternative to DALL·E 2 in Table III. We find that while accuracy decreases, it is still a viable inpainting substitute for our method.

Consistency Filtering: We test our method without applying the consistency filtering step by just combining all predicted viewpoints in Table III. This caused a substantial decrease in accuracy as any hallucinated object is kept in.

G. Texture as byproduct

As a byproduct, our method also produces novel views of unseen multi-object scenes from a single RGB-D image. Appendix Figure 8 shows our method compared to the ground truth. We show that by combining our masking method with DALL·E 2's inpainting capability, realistic novels views can be generated for multiple unseen objects.