Object Wake-up: 3D Object Rigging from a Single Image Supplementary

1 Network Implementation Details

In this section, we describe missing details of our proposed network for 3D object wake-up. For the reconstruction stage, as shown in Fig. 1, taking a single RGB image as input, and going through the Transformer based image encoder, the network predicts the occupancy probability for the sampled points from the deep implicit function. In addition, following the common encoder-decoder design for occupancy prediction, we also incorporate an auxiliary voxel prediction branch with a 3D CNN decoder to further improve the performance.

1.1 Transformer Encoder

A Vision Transformer takes image I_n as input by splitting the image into P^2 patches. At each time step, the corresponding patch is embedded by converting it into a fixed-size vector then summed with positional embedding. The Transformer model takes the embedded feature as input and outputs a dense image feature vector for each of the input patch as well as a global dense image feature vector extracted from all the patches. For single-view reconstruction, we only use the global dense image vector as the input. In our implementation, we use the Data-efficient image Transformer (DeiT) [10] as the backbone encoder.

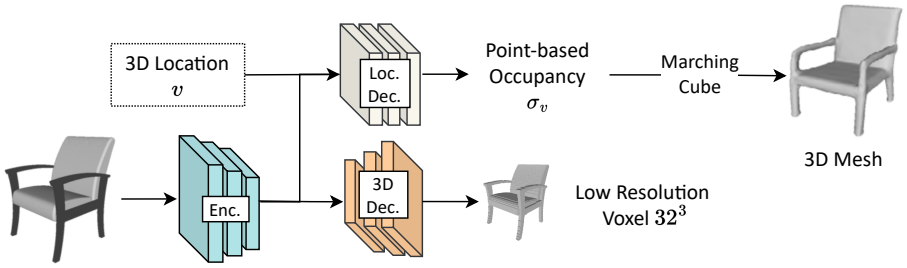


Figure 1: An overview of our reconstruction method from single RGB image.

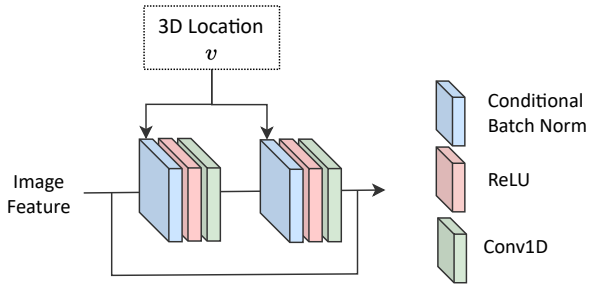


Figure 2: The residual block in occupancy decoder.

1.2 3D CNN Voxel Decoder

The input of the 3D CNN decoder is the image embedding outputted by the Transformer encoder. To make it feasible for a 3D convolutional network, we transform the 1D feature vector of length N to a $1^3 \times N$ volume. Here, we use a relatively simple architecture for the CNN Decoder in order to motivate the encoder to encode more information, where we follow the design of 3D-UNet [3] by applying 3D convolutional layers along with upsampling layers for the 3D feature volume iteratively until the final desired resolution is obtained. In our work, a voxel grid with a resolution of 64^3 is in use.

1.3 Occupancy Decoder

To reconstruct the 3D mesh, 3D points are uniformly sampled from the object space, and the proposed framework predicts the occupancy value for each of the sampled points conditioned on the extracted image feature from the encoder. The occupancy decoder consists of 5 residual blocks and each of them has two conditional batch normalization layers followed by a 1D convolutional layer. The conditional latent feature comes from the Transformer encoder, (i.e., the image feature) where the input latent feature is the embedded positional feature for each of the sampled points.

A simple illustration of the residual block can be found in Fig. 2.

1.4 3D Channel Activation Module for 3D-UNet

As described in our main manuscript, we adopt the popular SE block in 2D image classification to 3D-UNet. A 3D adaptive channel activation module here is developed as a plug-in module, to be attached after each of the encoder and decoder blocks of the 3D UNet. A clear visual illustration of the 3D adaptive channel activation module is presented in Fig. 3.

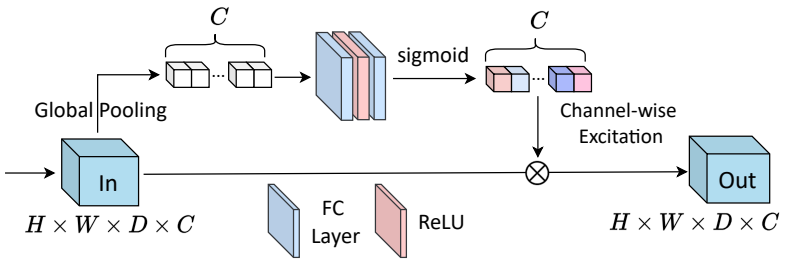


Figure 3: The 3D adaptive channel activation module.

2 Datasets

2.1 Our ShapeRR dataset

ShapeNet [1] is a large-scale 3D object dataset consisting of 55 object categories with over 50K 3D models. In this work, we focus on 4 categories that can be applied for animation, namely chair, table, airplane and lamp.

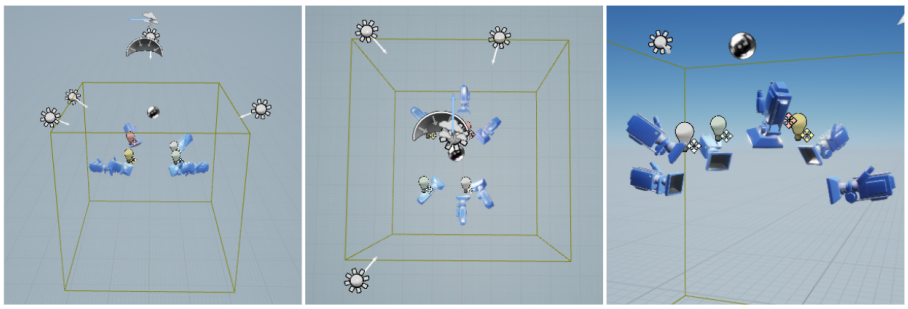
Our rendering system is designed using Unreal Engine 4 where we have multiple cameras placed in the scene with a group of spotlights and skylights. The setup of our ShapeRR image capturing system is shown in Fig. 4. Basically, the object is placed at the origin in the scene, and a randomly generated transformation is applied to the object to enhance the diversity of object orientation and position. Specifically, we have used 6 cameras to capture the image and 8 different lighting sources where they are randomly turned on and off in the rendering process. Our rendering system is able to generate realistic images with customized setting in terms of adaptive output image resolution, multiple materials for objects, varied lighting sources and conditions.

For each object, we generate 4 random transformations. The rendered images have the resolution of 256×256 with 24 random views. For a better comparison with previous work, R2N2 [2], here we provide sample rendered images in Fig. reffig:compare. The top row contains the original rendered image where the bottom row provides a zoom-in view. It is worth noting that our rendering results provide fine-grained details and realistic shadow created by adaptive lighting sources. In particular, in the original R2N2 dataset, the material of 3D objects cannot be correctly loaded and rendered (in the table example) where we have successfully fixed this issue.

2.2 Our SSkel dataset

As there is no existing dataset of general 3D objects with ground-truth skeletons, we collect such a dataset (named SSkel for *ShapeNet skeleton*) by designing an annotation tool to place joints and build kinematic trees for the 3D shapes. To ensure consistency, a predefined protocol is used for each object category.

Annotation Tool. In order to place joints, bones and eventually build a kinematic tree annotation, we need to be able to interact and manipulate the 3D object in a given



(a) Front-view (b) Top-view (c) Camera Placement

Figure 4: An overview of our realistic rendering image capturing system. In (a) and (b) we provide a glance on our system configuration. In (c), different camera positions that simulate common photo capture viewpoint are shown in a closer view.

Figure 5: A comparison of rendered images between our developed system and the R2N2 dataset. Our rendering provides significantly better visual results.

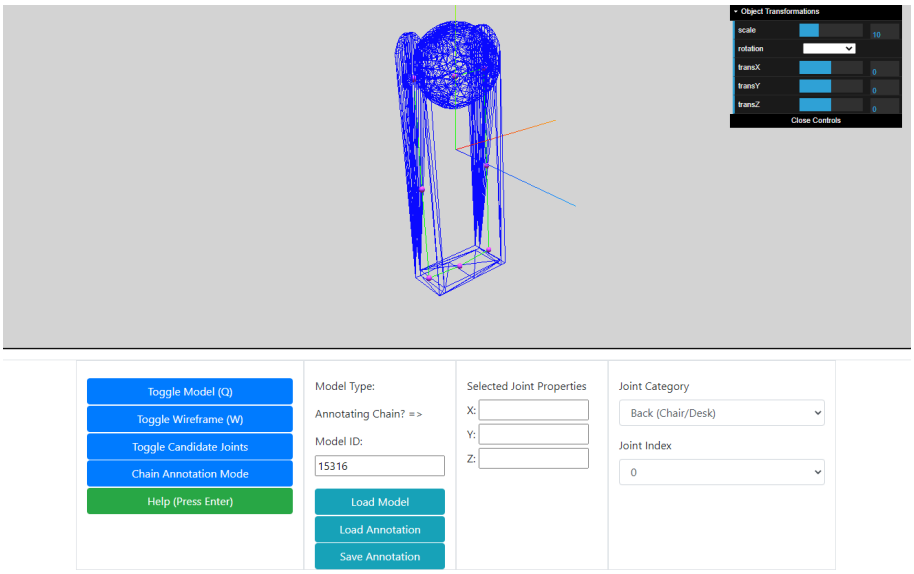


Figure 6: The developed annotation tool for skeleton annotation to build our SSkel dataset.

scene. We used modern web graphic libraries including three.js and WebGL to develop a web-based annotation interface. The interface can process commonly used 3D object format, such as OBJ, FBX and PLY.

Once the object is loaded to the scene, we can view the object in different modes, namely meshed, wire frame, and point cloud. We will automatically generate candidate joints by using the part segmentation information [6, 9]. Furthermore, the interface allows the user to manually place joints. Basically, the annotator can adjust the spatial position of the joint, label the type of the joint (root, or other types). A screenshot of the developed annotation tool is shown in Fig. 6.

Visualization of Sample Data. We show some sampled annotation results from different categories in Fig. 7.

3 Experiments

In this section, we demonstrate supplementary experiments for our work.

3.1 Training

The 3D reconstruction network is trained in a 2-stage fashion: we first train the Transformer encoder and voxel 3D CNN decoder for 10000 iteration with a batch size of 32, learning rate 0.00001 using Adam optimizer. Then, we freeze the 3D voxel branch and fine-tune the warmed-up Transformer encoder with the occupancy decoder, using the same batch size but a learning rate of 0.0001 and train it with an end-to-end fashion

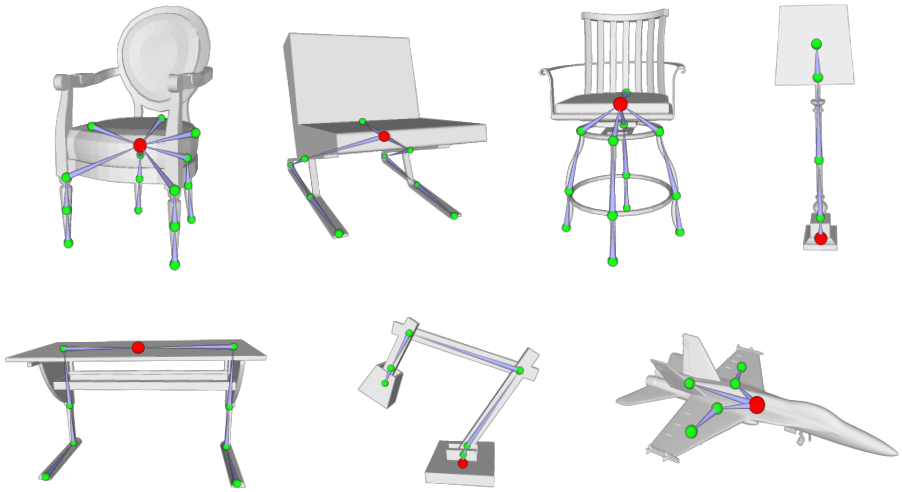


Figure 7: Sampled annotation models in our SSkel dataset. The root joint is marked in red.

ResNet	DeiT	Vox.	Dice	CD (↓)	Vol. IoU (↑)
✓				2.1504	0.5014
	✓			1.9801	0.5217
	✓	✓		1.9723	0.5268
	✓	✓	✓	1.9301	0.5339

Table 1: Ablation study on the ShapeRR dataset to validate the effectiveness of each component in our image-based reconstruction step. Chamfer Distance (CD) and Volumetric IoU (Vol. IoU) are used as metrics.

for another 10000 iteration. We use the exact same training scheme for experiments on both the ShapeRR dataset and R2N2 dataset.

3.2 Ablation study

For the sake of completeness, we conduct a group of ablative studies on the components we have in the reconstruction model. First, we test different backbone architectures (i.e., ResNet and DeiT-Tiny) used in our method. We then perform a warm-up training by including the voxel prediction branch, after the warm-up training, the voxel branch is discarded. Lastly, we compare training with Dice loss and the ordinary binary cross-entropy loss. Brief quantitative results are demonstrated in Table 1.

We progressively incorporate different module and loss function

	Volumetric IoU (↑)				
Method	Chair	Table	Lamp	Airplane	Avg.
OccNet [5]	0.501	0.506	0.371	0.571	0.487
DVR [7]	-	-	-	-	-
D^2IM-Net [4]	**0.561**	0.536	**0.421**	0.558	0.519
Ours	0.555	**0.539**	0.407	**0.626**	**0.531**
	Chamfer Distance (↓)				
Method	Chair	Table	Lamp	Airplane	Avg.
OccNet [5]	0.228	0.189	0.479	0.147	0.260
DVR [7]	0.264	0.280	0.413	0.190	0.286
D^2IM-Net [4]	0.329	0.356	0.557	0.358	0.401
Ours	**0.224**	**0.176**	**0.356**	**0.112**	**0.217**

Table 2: Image-based 3D mesh reconstruction on R2N2. Metrics are Chamfer-L1 Distance (the smaller the better) and Volumetric IoU (the larger the better). Best results are in **bold face**.

3.3 3D reconstruction on R2N2

For the sake of completeness, we also performed quantitative and qualitative evaluation of our proposed 3D reconstruction method with the originally proposed R2N2 dataset [2]. The R2N2 dataset is consisted of rendered images based on ShapeNet dataset. It is widely used by the literature therefore used by methods [5, 7, 4] where our proposed framework compared with. To make sure the quantitative comparison is fair and valid, we always follow the data split used by OccNet [5] and we also borrow the evaluation code from it.

The evaluation follows the previous works [5], we use volumetric IoU and Chamfer-L1 distance. State-of-the-art methods are compared, namely, OccNet [5], DVR [7], and D^2IM-Net [4]. Where D^2IM-Net is the most recent and best performing methods for 3D reconstruction task on R2N2, DVR and OccNet are also recent methods that perform well with real images.

The quantitative result is demonstrated in Table 2. Our proposed method outperform all existing methods according to the Chamfer-L1 metric, where we are slightly behind D^2IM-Net on two categories (Lamp and Chair) but still have advantage on overall performance in terms of volumetric IoU.

3.4 Failure Cases

To achieve the goal of object wake-up and manipulate the object in the image with articulated motions, it is critical to have a well reconstructed and rigged 3D model from the input image. In Fig. 8 we show some failure cases where the quality of the

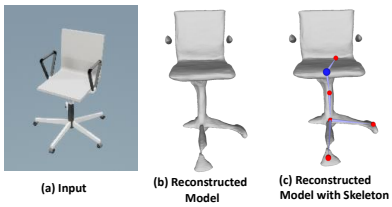


Figure 8: A failure case.

rigged 3D model cannot meet the requirements for animation purposes.

4 More visual results

4.1 Visual comparison on reconstruction

In Fig. 9, we provide extra visualization results of the reconstructed models when evaluated on our ShapeRR dataset. We have compared with several existing approaches, namely OccNet [5], DVR [8], and D2IM-Net [4].

References

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

[2] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3D object reconstruction. In *European Conference on Computer Vision (ECCV)*, pages 628–644. Springer, 2016.

[3] Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3D models. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

[4] Manyi Li and Hao Zhang. D2im-net: Learning detail disentangled implicit fields from single images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10246–10255, 2021.

[5] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[6] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[7] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

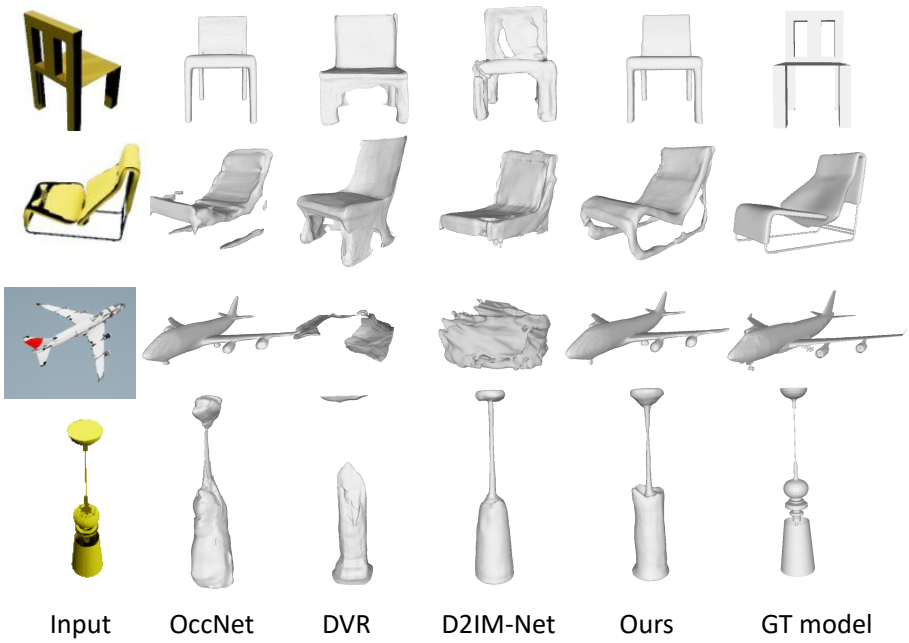


Figure 9: Visualizations of image-based 3D reconstruction on our ShapeRR dataset.

[8] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[9] Manolis Savva, Angel X. Chang, and Pat Hanrahan. Semantically-enriched 3d models for common-sense knowledge. 2015.

[10] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers and distillation through attention. In *International Conference on Machine Learning*, volume 139, pages 10347–10357, 2021.