

Prior-free 3D Object Tracking: Supplementary Materials

Xiuqiang Song^{1,3} Li Jin^{1,3} Zhengxian Zhang^{1,3} Jiachen Li⁴

Fan Zhong^{2,3} Guofeng Zhang⁵ Xueying Qin^{1,3*}

¹School of Software, Shandong University

²School of Computer Science and Technology, Shandong University

³Engineering Research Center of Digital Media Technology, Ministry of Education, P.R. China

⁴Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center, Qilu University of Technology

⁵State Key Lab of CAD&CG, Zhejiang University

song.xq, jinli, zhangzhengxian@mail.sdu.edu.cn jcli@qlu.edu.cn

zhongfan@sdu.edu.cn zhangguofeng@zju.edu.cn qxy@sdu.edu.cn

1. Mask Resize

The implementation for resizing masks obtained from SAM in Sec. 3.1 of our paper is shown in Fig. 1 and detailed below.

Assuming the world coordinate system coincides with the model coordinate system of the 3D model G_o , the model can be transformed into the camera coordinate system using the pose T , resulting in G_c :

$$G_c = TG_o. \quad (1)$$

The imaging process of the 3D model G_c onto the 2D image requires the use of the camera intrinsic parameters K , which can be calibrated in advance. Given that $P = [X, Y, Z]^T$ is a point of the 3D model G_c , the 2D projection point u of P in the 2D image is given by:

$$u = \pi(KP) = \pi\left(\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}\right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \end{bmatrix}, \quad (2)$$

where $\pi(\cdot)$ denotes the imaging process. From the equation above, we can observe that for the model G_c , f_x and f_y affect the size of its projection mask, while c_x and c_y determine the position of the projection mask. Therefore, by adjusting f_x , f_y , c_x , and c_y , we can scale and translate the mask without altering the shape of the object's projection or modifying the pose T .

As shown in Fig. 1, assume the origin is at the top-left corner of the mask image, with the horizontal direction as the x -axis and the vertical direction as the y -axis. The mask image \mathcal{M} has a width w and height h , the bounding box

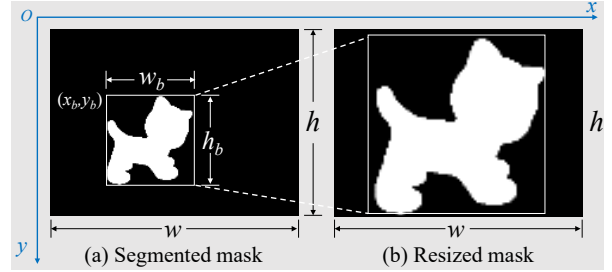


Figure 1. The segmented mask and the resized mask.

b has a width w_b and height h_b , and the coordinate of the top-left corner of b is $(x_b, y_b)^T$.

First, calculate a scaling factor s to enlarge the the imaging mask:

$$s = \max(1.0, \lambda \times \min(\frac{w}{w_b}, \frac{h}{h_b})), \quad (3)$$

where λ is a adjust parameter, here we set it to 0.9. In addition, in practical implementation, if $s < 1.1$, the scale operation can be skipped (i.e., set $s = 1.0$) to avoid the accuracy loss caused by image resizing.

Then, calculate the center c of \mathcal{M} and the center c_b of the bounding box b :

$$c = [\frac{w}{2}, \frac{h}{2}]^T, \quad (4)$$

$$c_b = [x_b + \frac{w_b}{2}, y_b + \frac{h_b}{2}]^T. \quad (5)$$

Calculate an offset Δc that is used to translate the imaging

*Xueying Qin is the corresponding author.

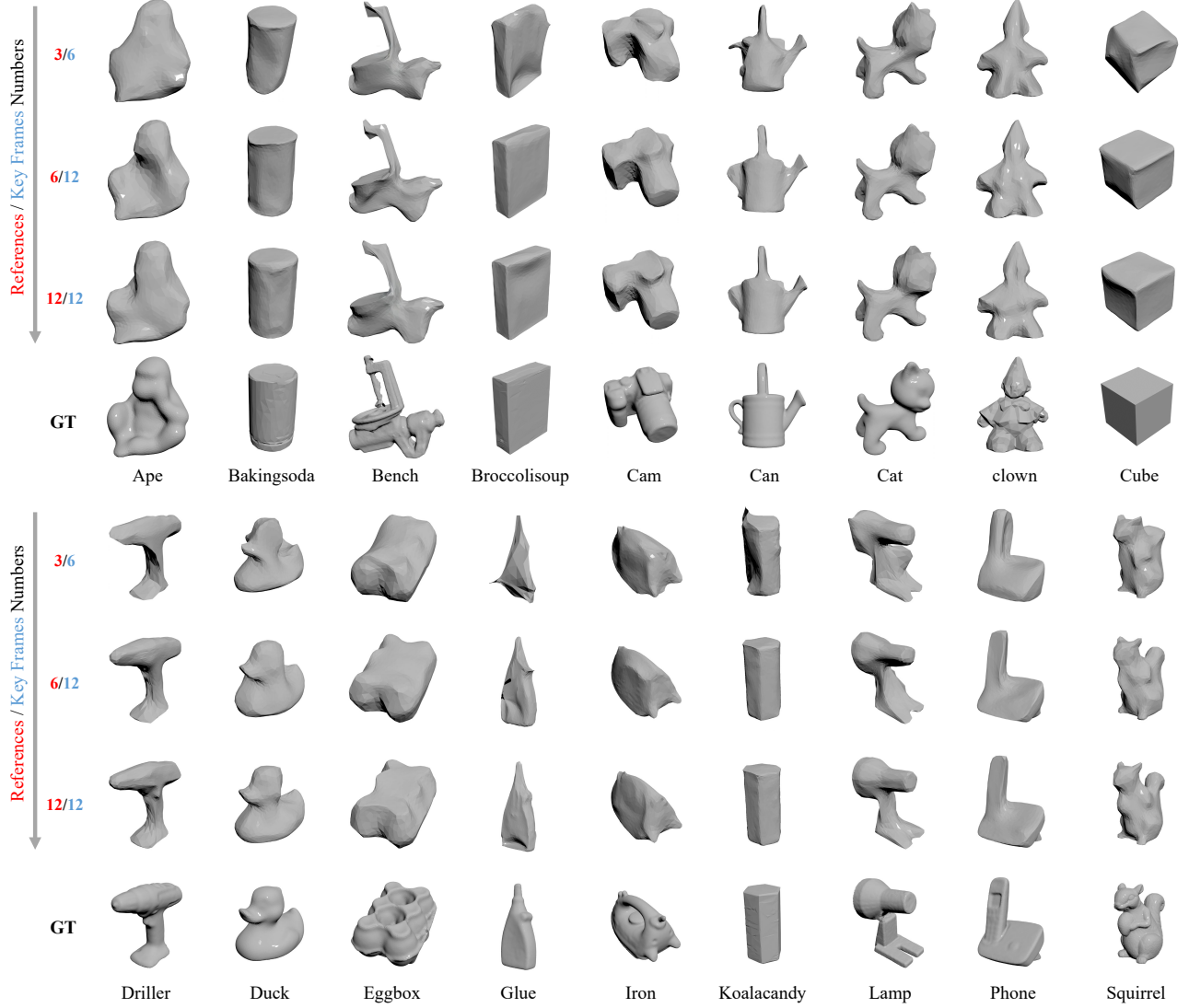


Figure 2. The final generated models (3 iterations) in different reference and key frames on the RBOT dataset.

mask:

$$\begin{aligned}
 \Delta \mathbf{c} &= \mathbf{c} - s \times \mathbf{c}_b \\
 &= \left[\frac{w - s \times w_b - 2s \times x_b}{2}, \frac{h - s \times h_b - 2s \times y_b}{2} \right]^\top \\
 &= [\Delta w, \Delta h]^\top.
 \end{aligned} \tag{6}$$

Then adjust f_x , f_y , c_x and c_y in \mathbf{K} using the scaling factor s and the offset $\Delta \mathbf{c}$:

$$\begin{aligned}
 f_x &\leftarrow s \times f_x, \\
 f_y &\leftarrow s \times f_y, \\
 c_x &\leftarrow s \times c_x + \Delta w, \\
 c_y &\leftarrow s \times c_y + \Delta h.
 \end{aligned} \tag{7}$$

By projecting the model under the updated \mathbf{K} , a new bounding box can be obtained. We only need to map the mask from the original bounding box area to the new bounding box to obtain a new mask, as shown in Fig. 1 (b). SoftRas [2] requires the input mask to be square, so we performed zero-padding on the resized mask. Padding with zeros along the x -axis or y -axis does not require changing the intrinsic parameters and can be done directly.

2. Loss Function

The loss function \mathcal{L} (see Eq. (1) in our paper) in the model generation is:

$$\mathcal{L} = \mathcal{L}_{IoU}(\{\mathbf{G}, \mathbf{T}_i, \mathbf{K}_i\}, \{\mathcal{M}_i\}) + \mu \mathcal{L}_l(\mathbf{G}), \tag{8}$$

where \mathcal{L}_{IoU} represents the 2D IoU loss between the given masks $\{\mathcal{M}_i\}$ and the projected masks of the model \mathbf{G} under poses $\{\mathbf{T}_i\}$ and intrinsics $\{\mathbf{K}_i\}$, \mathbf{G} represents the intermediate 3D model during inverse rendering. \mathcal{L}_l is the Laplacian regularization term used to control the smoothness of the geometry. The parameter μ acts as a balancing factor:

$$\mathcal{L}_{IoU} = 1 - \frac{1}{n} \sum_{i=1}^n \frac{|\mathcal{R}(\mathbf{G}, \mathbf{T}_i, \mathbf{K}_i) \cap \mathcal{M}_i|}{|\mathcal{R}(\mathbf{G}, \mathbf{T}_i, \mathbf{K}_i) \cup \mathcal{M}_i|}, \quad (9)$$

$$\mathcal{L}_l = \sum_{v=1}^m \sum_{d=1}^3 (\mathbf{L} \cdot \mathbf{X}_G)_{v,d}^2, \quad (10)$$

where $\mathcal{R}(\mathbf{G}, \mathbf{T}_i, \mathbf{K}_i)$ represents the rendering of the 3D model \mathbf{G} into a 2D mask at the pose \mathbf{T}_i with the intrinsic matrix \mathbf{K}_i , and n is the number of input poses. $\mathbf{L} \in \mathbb{R}^{m \times m}$ is the Laplacian matrix, where m is the number of vertices in the model \mathbf{G} . $\mathbf{X}_G \in \mathbb{R}^{m \times 3}$ is the matrix composed of all vertices in \mathbf{G} , with the subscript (v, d) indicating the element in the v -th row and d -th column of the matrix. The loss function (8) is intentionally kept concise to facilitate the rapid convergence of the input geometry to a specific shape suitable for tracking, while also reducing the risk of generalization issues.

3. The Baseline Tracker

The baseline tracker SLOT [1] relies on a given mesh model to optimize object poses. Its pose optimization process can be summarized as aligning the implicit contour of the object in the frame with the projected contour of the model under the initial pose. BIT generates 3D models based on the 2D contours of the target object in the frames. Consequently, the generated 3D models have sufficiently accurate projected contours, making them well-suited for the tracking purposes of the SLOT method.

4. The Generated Models on the RBOT Dataset

In this section, we present the detailed results of Fig. 7 in our paper, as shown in Fig. 2.

References

- [1] Hong Huang, Fan Zhong, and Xueying Qin. Pixel-wise weighted region-based 3D object tracking using contour constraints. *TVCG*, 28(12):4319–4331, 2022. 3
- [2] Shichen Liu, Weikai Chen, Tianye Li, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In *ICCV*, pages 7707–7716, 2019. 2