Appendix

In this supplementary, we provide detailed information about the data preparation, method implementation, and experimental results.

1 Data Preparation

1.1 Data Preprocessing

Normalization Matrix The normalization transformation is constructed by first moving the origin to the center of the object, denoted as $\mathbf{c} = [c_x, c_y, c_z]^{\mathrm{T}}$, and then applying a scale normalization according to the diameter of the object. The normalization matrix is calculated as:

$$\mathbf{S} = \begin{bmatrix} 1/d & 0 & 0 & 0 \\ 0 & 1/d & 0 & 0 \\ 0 & 0 & 1/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \text{(A1)}$$

where d is the diameter of the object. For simplicity, our normalization uses the diameter as a single size parameter, which is different from that of One2Any (Liu et al. 2025), where a size parameter is calculated for each axis. Precisely estimating the size parameters of an object is challenging from a single reference depth image, particularly when the object is occluded by itself or other objects. Moreover, the estimation is vulnerable to depth noise, which can introduce outliers. To overcome this, we estimate the diameter from the 2D query mask and the median depth. Specifically, the diameter is approximated by:

$$d = \frac{D_{median} \cdot \sqrt{w^2 + h^2}}{(f_x + f_y)/2}, \quad \text{(A2)}$$

where $\{w, h\}$ (in pixels) are the width and height of the visible area, respectively; D_{median} is the median of the depth in the visible area, and $\{f_x, f_y\}$ are the camera intrinsic focal lengths (in pixels) along the x- and y-axes, respectively.

1.2 Training Data

We apply data augmentations during training. The RGB images are augmented with random backgrounds using images from the PASCAL VOC dataset (Everingham et al. 2010). The reference ROC maps are randomly corrupted, with parts masked out.

2 Implementation Details

2.1 The Image Encoders

For RGB image encoding, we employ a DINOv2 (Oquab et al. 2023) backbone with trainable parameters, leveraging its powerful pretrained encoder while allowing for fine-tuning to adapt to pose estimation requirements. To ensure dimensional consistency between modalities, we encode ROC maps using a ViT-B encoder (Dosovitskiy et al. 2020).

2.2 The Tokenizer

To ensure compatibility with our RGB feature encoder's output scale, the tokenizer generates compact token maps at $1/f$ the input resolution ($f = 16$ for our tokenizer). Formally, given an input ROC map $\mathbf{X}^Q \in \mathbb{R}^{H \times W \times 3}$, the encoder of the VQ-VAE $\mathcal{T}_e(\cdot)$ encodes $\mathbf{X}^Q$ to a continuous latent vector map:

$$\mathbf{z}_e = \mathcal{T}_e(\mathbf{X}^Q) \in \mathbb{R}^{h \times w \times d}, \quad \text{where} \begin{cases} h = \lfloor H/f \rfloor, \\ w = \lfloor W/f \rfloor \end{cases}. \quad \text{(A3)}$$

Each latent vector is then quantized using a learned codebook $\mathcal{B} = \{\mathbf{e}_k\}_{k=1}^K \subset \mathbb{R}^d$:

$$\mathbf{z}_q^i = \underset{\mathbf{e}_k \in \mathcal{B}}{\operatorname{argmin}} \|\mathbf{z}_e^i - \mathbf{e}_k\|_2 \quad \text{(A4)}$$

where $\mathbf{z}_e^i, \mathbf{z}_q^i$ denotes the latent vectors at position $i \in \{1, \ldots, h \cdot w\}$. The discrete tokens are the indices of the quantized vectors in the codebook: $\{s_1, \ldots, s_{h \cdot w}\}$.

We use the ground-truth $\mathbf{X}^Q$ sampled from the training set to train the tokenizer. The training objective of the tokenizer is as follows:

$$\begin{aligned} \mathcal{L}_{\text{VQ-VAE}} = &\left\|\mathbf{X}^Q - \mathcal{T}_d(\mathbf{z}_q)\right\|^2 + \|\text{sg}(\mathbf{z}_e) - \mathbf{z}_q\|^2 \\ &+ \beta \left\|\mathbf{z}_e - \text{sg}(\mathbf{z}_q)\right\|^2, \end{aligned} \quad \text{(A5)}$$

where $\mathbf{z}_q$ is the latent vector returned by the encoder of the tokenizer, $\mathbf{z}_e$ is the embedding vector found by nearest lookup in the codebook, and $\text{sg}(\cdot)$ is the stop gradient operator to prevent gradients back-propagating through its argument during training.

2.3 The Fusion Blocks

As illustrated in Fig. A1, we employ N (N=8) fusion blocks to integrate information from both the query and reference. Each fusion block primarily consists of a self-attention layer, a cross-attention layer, a feed-forward network (FFN), and layer normalizations. To incorporate positional information, the features are added with positional encoding before being fed into the blocks. The final output is a set of conditional features used for decoding.

2.4 The ROC Decoder

As depicted in Fig. A2, the ROC decoder operates autoregressively at each generation step by processing both the sequence of previously generated tokens and the conditional features through multiple stacked decoding layers to predict the subsequent tokens. The embeddings for these generated tokens are retrieved by querying the tokenizer's codebook, while the conditional features aligned with the target token position are selectively incorporated into the final layer of each decoding block.

2.5 Training Details

Training an ROC-map tokenizer is a prerequisite for our main network. The tokenizer is trained using ROC maps of the query-reference pairs sampled from the train set. The training of the tokenizer lasts for 200k iterations. After being

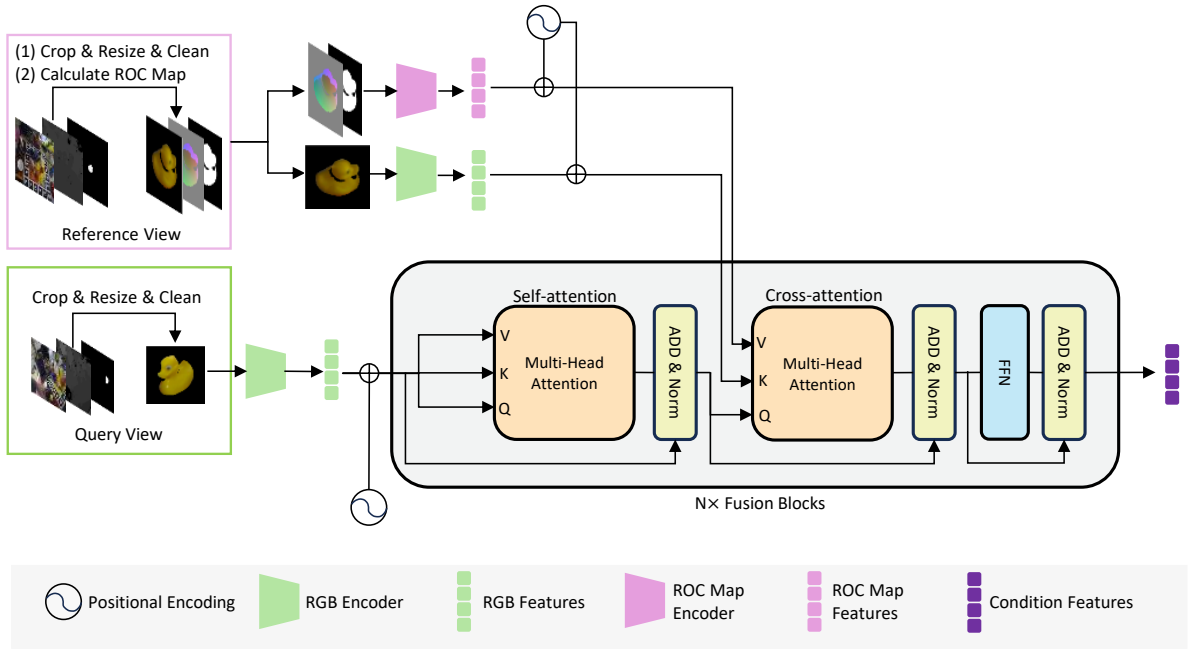

Figure A1: Architecture of the Fusion Blocks. The reference features and query features are integrated by the fusion blocks. The output condition feature is further used for token decoding.

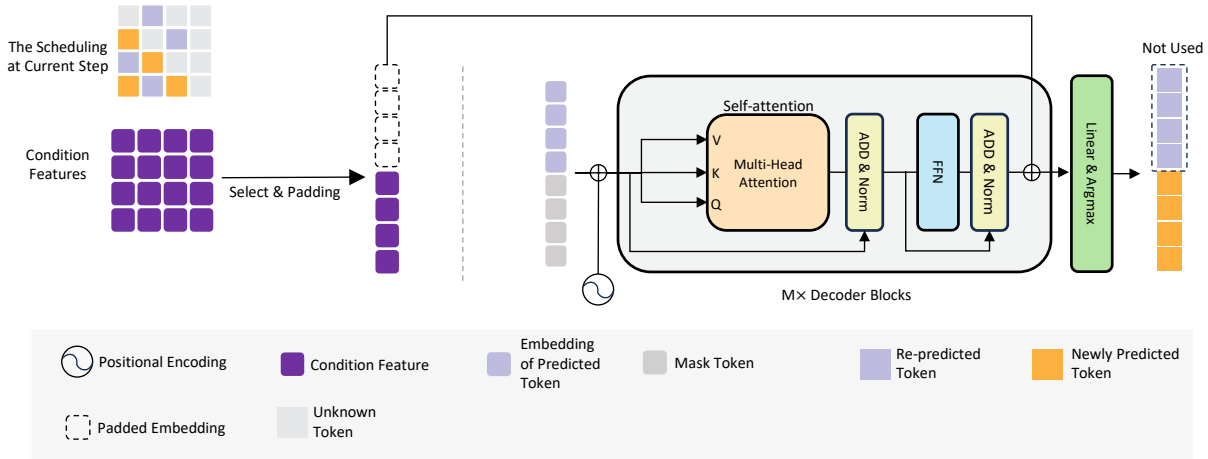

Figure A2: Architecture of the ROC Decoder. This component autoregressively decode tokens conditioned on the condition features. Condition features are padded for a consistent length with the input of the decoder.

trained, the tokenizer is frozen in the following steps. Subsequently, the coordinate map estimation network is trained for 400k iterations. We employ the Ranger (Wright 2019) optimizer with a OneCycle (Smith and Topin 2019) learning rate scheduler. We conduct all training on a server equipped with 4 NVIDIA RTX 4090 GPUs. The tokenizer training completes within several hours, while the main network requires approximately 5 days to converge. For both the tokenizer and the main network, the batch size is set to 64. Due to computational limitations, the number of training iterations for the ablation study is set to 200K, with a batch size of 32 for all models.

3 Experimental Results

3.1 Full Results on the YCB-Video Dataset

In Table A3, we present the performance of each object in the YCB-Video dataset.

3.2 Inference with Multi-view References

Our method can be further extended to a multi-view setup in which multi-view references and their poses are available. We generate a pose hypothesis from each reference and then adopt the pose selection introduced by One2Any (Liu et al. 2025) to select the best pose, where the reference masks are re-projected by relative pose and the best pose is selected based on the mIoU between the re-projected reference mask and the query mask. As demonstrated in Tab. A1, our method surpasses One2Any and existing multi-view based methods except FS6D (He et al. 2022). Since the mIoU based selection does not guarantee the optimal selection (Liu et al. 2025), we further present the performance with optimal selection where the pose of the best view is selected. Our method demonstrates superior performance compared to other approaches with optimal view selection, suggesting strong potential for further enhancement in the multi-view setting.

3.3 Ablation on Inference Configurations

The autoregressive framework provides multiple prediction choices during inference. To investigate how these design choices affect pose accuracy, we conducted experiments on the LINEMOD dataset (Hinterstoisser et al. 2011).

Generation Steps Our initial evaluation focused on model performance under varying generation steps. As shown in Tab. A2, our model achieves optimal pose accuracy at 64 steps, with gradually degrading performance as the step count decreases. This demonstrates that predicting new tokens conditioned on preceding tokens is a critical architectural consideration. Notably, even when reduced to a single step (step=1), our model maintains reasonable accuracy without catastrophic failure. This property enables deployment in latency-sensitive applications such as pose tracking, which is discussed in the following sections.

Token Scheduler In our implementation, the number of tokens to be predicted in each generation step is controlled by a token scheduler. To investigate the sensitivity to the type of scheduler, we apply a linear scheduler and a cosine scheduler individually at evaluation time. As displayed in Tab. A2, the cosine scheduler, which progressively increases generation speed after an initial cautious phase, results in measurable performance gains, empirically validating the need for deliberate token generation during high-ambiguity initial stages.

Generation Order Generating image by random order or raster scan order are common in autoregressive image-generation models (Xiong et al. 2024). To investigate whether the performance can be improved by a better global setting of generation order, we evaluate our method with random order and raster scan order. As presented in Tab. A2, random order or raster-scan order shows comparable performance, suggesting that the overall performance is invariant to a global setting of order.

Randomness in Inference. In our method, we use argmax to select tokens from the predicted distribution deterministically. However, probabilistic token sampling from the distribution with a temperature parameter τ can also be an option. As displayed in Tab. A2, we observe a performance degradation after adding randomness to the inference procedure, indicating that deterministic token selection is preferable for maintaining pose estimation accuracy.

3.4 Results on Pose Tracking

We compare the pose tracking performance on the full test sequence of the YCB-Video dataset. Following existing methods (Liu et al. 2025; Wen et al. 2024), we use the first frame as the reference for the entire video sequence. As shown in Tab. A4, our method achieves a higher AUC than existing one-reference methods and shows competitive results against pose tracking approaches based on CAD models.

3.5 Visualization of the Generation

For a better understanding of our method, we present a visualization of the token generation process. As demonstrated in Fig. A3, we visualize tokens cumulatively predicted at each step by decoding them using the tokenizer.

3.6 Qualitative Results on Toyota-Light

In Fig. A4, we present qualitative results on the Toyota-Light dataset (Hodaň et al. 2018).

3.7 Qualitative Results on Real-275

In Fig. A5, we display qualitative results on the Real-275 dataset (Wang et al. 2019).

3.8 Qualitative Results on Self-collected Data

To further demonstrate the performance of our method on real-world novel objects, we collected RGB-D videos of several common household objects using a RealSense D435i camera. In the first frame, which serves as the reference frame, each object is placed on a platform facing the camera, and its orientation is defined as the canonical rotation. The mask of the object is obtained by Track Anything (Yang

Methods	Modality	Ref. Images	ape	benchvise	cam	can	cat	driller	duck	eggbox	glue	holepuncher	iron	lamp	phone	avg.
OnePose	RGB	200	11.8	92.6	88.1	77.2	47.9	74.5	34.2	71.3	37.5	54.9	89.2	87.6	60.6	63.6
OnePose++	RGB	200	31.2	97.3	88.0	89.8	70.4	92.5	42.3	99.7	48.0	69.7	97.4	97.8	76.0	76.9
LatentFusion	RGBD	16	88.0	92.4	74.4	88.8	94.5	91.7	68.1	96.3	49.4	82.1	74.6	94.7	91.5	83.6
FS6D + ICP	RGBD	16	78.0	88.5	91.0	89.5	97.5	92.0	75.5	99.5	99.5	96.0	87.5	97.0	97.5	91.5
One2Any	RGBD	16-mIoU	82.1	85.5	92.8	75.9	94.1	80.4	65.9	**100.0**	99.9	70.7	61.7	91.5	84.1	83.7
One2Any	RGBD	16-best view	84.8	98.3	98.8	95.2	95.9	93.3	76.2	**100.0**	99.9	92.9	95.1	94.4	93.9	93.8
CoordAR	RGBD	16-mIoU	85.0	99.9	79.4	95.1	93.3	96.4	66.3	98.2	100.0	93.2	79.9	94.1	88.0	89.9
CoordAR	RGBD	16-best view	**95.1**	**100.0**	**99.7**	**100.0**	**99.7**	**99.4**	**96.0**	99.9	**100.0**	**99.6**	**99.3**	**99.8**	**99.4**	**99.1**

Table A1: Multiview performance on LINEMOD (Hinterstoisser et al. 2011). Baseline results of taken from One2Any (Liu et al. 2025).

.

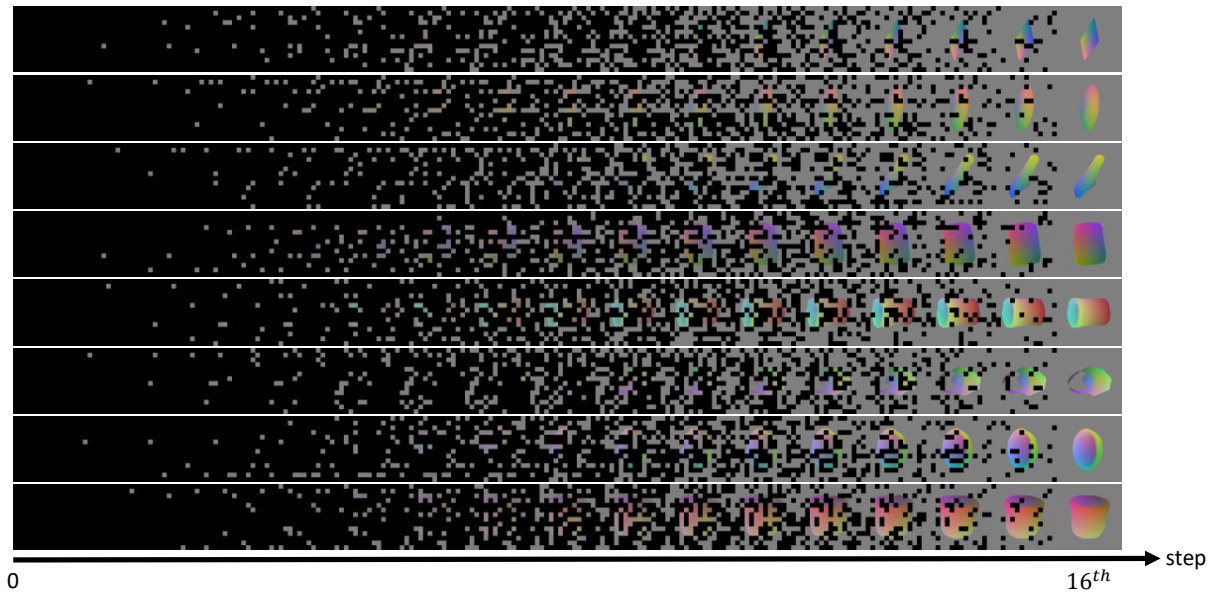

Figure A3: Visualization of the generation. We visualize the generation with 16 steps. Patches corresponding to unpredicted tokens are masked in black.

Figure A4: Qualitative results on the Toyota-Light dataset (Hodaň et al. 2018).

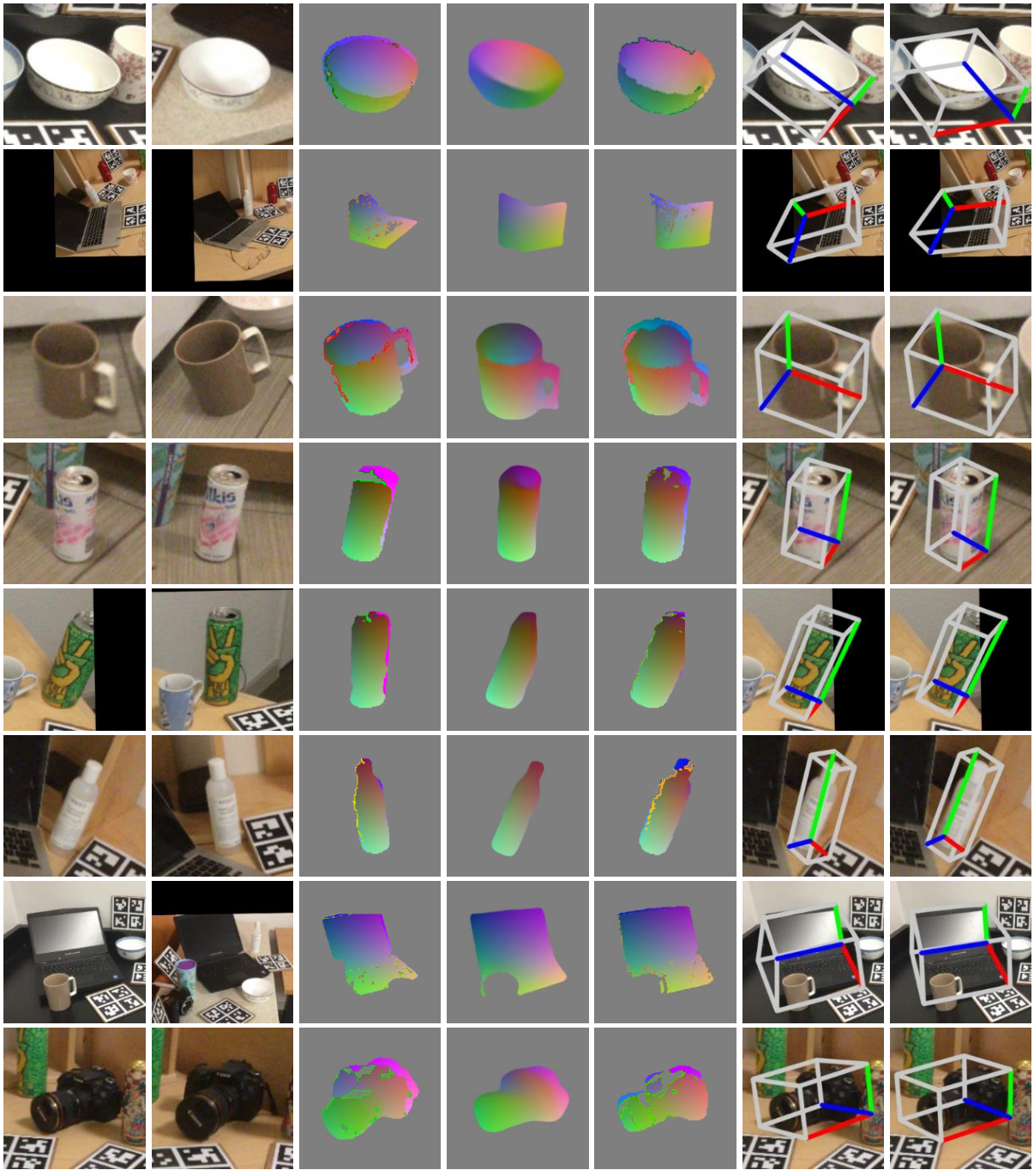

Figure A5: Qualitative results on the Real275 dataset (Wang et al. 2019).

Component	Variations	AUC of ADD(-S)	ADD(-S)	AR
Generation steps	1	73.5	68.0	56.3
	4	74.2	69.5	58.4
	16	**74.4**	73.2	61.8
	64	74.3	**73.6**	**61.9**
Token scheduler	linear	74.2	72.4	60.4
	cosine	**74.3**	**73.6**	**61.9**
Generation order	raster scan	**74.3**	**73.6**	**62.1**
	random	**74.3**	**73.6**	61.9
Randomness	$\tau = 1.0$	72.1	69.4	57.6
	$\tau = 0.5$	72.6	70.6	58.2
	argmax	**74.3**	**73.6**	**61.9**

Table A2: Performance with different inference configurations. The last row of each configuration is our default setting.

et al. 2023). As shown in Fig. A6, our method accurately estimates the relative poses even when the viewpoints exhibit significant variations.

4 Limitations

While demonstrating promising performance, our method is bothered by token generation order. Following previous work (Li et al. 2024) for image generation, we adopt random order by default and other orders are tried in the supplementary materials. As demonstrated in Fig. A7, we find that generation with an improper token order may result in an erroneous ROC map, particularly when initiating generation from high-uncertainty regions. This limitation suggests future improvements could be achieved by optimized token ordering strategies, or adoption of next-scale prediction paradigm (Tian et al. 2024) to mitigate order dependence.

Methods	**PREDATOR**		**FS6D**		**FoundationPose**		**FoundationPose**		**NOPE**		**One2Any**		**CoordAR**	
Ref. Images	16		16		16 - CAD		1 - CAD		1 + GT trans		1		1	
metrics of AUC	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD
002_master_chef_can	73.0	17.4	92.6	36.8	**96.9**	**91.3**	87.3	73.3	96.8	17.8	94.9	84.3	**98.5**	**88.9**
003_cracker_box	41.7	8.3	83.9	24.5	**97.5**	**96.2**	**92.0**	72.2	83.0	2.8	91.1	83.3	**98.9**	**95.6**
004_sugar_box	53.7	15.3	95.1	43.9	**97.5**	**87.2**	88.2	87.1	86.5	22.3	95.3	88.0	**100.0**	**99.3**
005_tomato_soup_can	81.2	44.4	93.0	54.2	**97.6**	**93.3**	95.2	92.3	95.9	48.4	93.6	80.9	**96.7**	**90.8**
006_mustard_bottle	35.5	5.0	97.0	71.1	**98.4**	**97.3**	88.4	76.6	91.3	42.7	93.8	87.6	**100.0**	**94.6**
007_tuna_fish_can	78.2	34.2	94.5	53.9	**97.7**	**73.7**	90.5	76.9	97.0	33.3	95.9	90.0	**99.4**	**85.5**
008_pudding_box	73.5	24.2	94.9	79.6	**98.5**	**97.0**	91.7	77.8	84.4	20.9	96.1	93.3	**100.0**	**100.0**
009_gelatin_box	81.4	37.5	98.3	32.1	**98.5**	**97.3**	92.7	87.7	87.3	35.3	97.7	96.1	**100.0**	**100.0**
010_potted_meat_can	62.0	20.9	87.6	54.9	**96.6**	**82.3**	90.3	**83.5**	**92.8**	31.9	86.3	72.5	86.0	70.4
011_banana	57.7	9.9	94.0	69.1	**98.1**	**95.4**	90.3	76.3	61.3	11.4	95.0	85.7	**97.6**	**89.8**
019_pitcher_base	83.7	18.1	91.1	40.4	**97.9**	**96.6**	92.1	86.9	88.9	6.1	93.6	87.7	**100.0**	**97.7**
021_bleach_cleanser	88.3	48.1	89.4	44.1	**97.4**	**93.3**	90.8	85.5	89.6	32.3	93.0	84.6	**99.3**	**93.8**
024_bowl	73.2	17.4	74.7	0.9	**94.9**	**89.7**	87.5	43.6	93.2	6.7	92.1	**65.1**	**99.9**	**71.1**
025_mug	84.8	29.5	86.5	39.2	**96.2**	**75.8**	91.0	74.1	92.5	31.6	95.5	82.9	**99.8**	**95.3**
035_power_drill	60.6	12.3	73.0	19.8	**98.0**	**96.3**	97.0	**96.8**	56.0	0.0	92.4	84.6	**99.7**	96.3
036_wood_block	70.5	10.0	94.7	27.9	**97.4**	**94.7**	67.1	19.9	77.1	0.0	92.7	**85.0**	**96.4**	74.2
037_scissors	75.5	25.0	74.2	27.7	**97.8**	**95.5**	**97.4**	**94.7**	75.5	0.0	92.6	84.7	85.1	69.8
040_large_marker	81.8	38.9	97.4	74.2	**98.6**	**96.5**	**92.7**	90.4	79.6	39.3	**96.5**	**91.0**	82.0	71.9
051_large_clamp	83.0	34.4	82.7	34.7	**96.9**	**92.7**	87.4	68.9	**100.0**	**100.0**	92.9	84.2	93.4	78.3
052_extra_large_clamp	72.9	24.1	65.7	10.1	**97.6**	**94.1**	**90.5**	43.7	82.6	0.0	89.5	65.8	90.3	**79.5**
061_foam_brick	79.2	35.5	95.7	45.8	**98.1**	**93.4**	98.7	90.9	95.2	43.5	97.6	**95.9**	**98.9**	93.2
mean	71.0	24.3	88.4	42.1	**97.4**	**91.5**	90.4	76.1	86.0	25.1	**93.7**	**84.4**	**96.3**	**87.3**

Table A3: Full results on the YCB-Video dataset.

Method	**Wuthrich**		**RGF**		**ICG**		**FoundationPose**		**FoundationPose**		**One2Any**		**CoordAR**	
Ref.	CAD		CAD		CAD		16 frames-CAD		1^{st} frame-CAD		1^{st} frame		1^{st} frame	
metrics of AUC	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD
002_master_chef_can	90.7	55.6	90.2	46.2	89.7	66.4	**96.9**	**91.2**	83.3	38.1	94.8	83.8	**98.7**	**92.5**
003_cracker_box	97.2	96.4	72.3	57.0	92.1	82.4	**97.5**	**96.2**	94.0	78.3	91.3	83.0	**98.9**	**95.1**
004_sugar_box	97.9	97.1	72.7	50.4	**98.4**	**96.1**	97.4	94.5	78.7	40.0	95.3	88.7	**100.0**	**99.3**
005_tomato_soup_can	89.5	64.7	91.6	72.4	97.3	73.2	**97.9**	**94.3**	49.3	14.0	**95.5**	87.1	95.4	**88.6**
006_mustard_bottle	98.0	97.1	98.2	87.7	98.4	96.2	**98.5**	**97.3**	58.8	24.8	93.8	87.7	**99.8**	**92.1**
007_tuna_fish_can	93.3	69.1	52.9	28.7	95.8	73.2	**97.8**	**84.0**	**97.4**	75.3	95.9	**89.5**	97.2	85.8
008_pudding_box	97.9	96.8	18.0	12.7	88.9	73.8	**98.5**	**96.9**	98.3	96.9	96.3	93.5	**100.0**	**100.0**
009_gelatin_box	98.4	97.5	70.7	49.1	**98.8**	97.2	98.5	**97.6**	98.6	97.2	97.7	96.1	**100.0**	**100.0**
010_potted_meat_can	86.7	83.7	45.6	44.1	97.3	93.3	**97.5**	**94.8**	52.6	5.5	**84.0**	**65.9**	80.9	62.2
011_banana	96.1	86.3	97.7	93.3	**98.4**	**95.6**	98.1	**95.6**	84.7	64.7	95.1	83.6	**98.5**	**91.1**
019_pitcher_base	97.7	97.3	98.2	97.9	**98.8**	**97.0**	98.0	96.8	96.4	94.6	93.4	87.0	**100.0**	**97.8**
021_bleach_cleanser	97.2	95.2	97.3	95.9	**97.5**	92.6	**97.5**	**94.7**	58.6	16.6	93.2	84.8	**98.7**	**93.1**
024_bowl	97.2	30.4	82.4	24.2	**98.4**	74.4	95.3	**90.5**	40.2	12.4	91.8	71.8	**98.2**	**73.1**
025_mug	93.3	83.2	71.2	60.0	**98.5**	**95.6**	96.1	91.5	91.3	54.4	95.5	83.3	**99.8**	**96.1**
035_power_drill	97.8	97.1	98.3	97.9	98.5	96.7	**97.9**	**96.3**	69.2	50.4	92.8	85.5	**99.6**	**96.2**
036_wood_block	96.9	95.5	62.5	45.7	**97.2**	**93.5**	97.0	92.9	**95.9**	**88.4**	92.8	85.5	95.4	74.9
037_scissors	16.2	4.2	38.6	20.9	97.3	93.5	**97.8**	**95.5**	**97.9**	**96.0**	91.7	81.0	85.4	69.4
040_large_marker	53.0	35.6	18.9	12.2	97.8	88.5	**98.6**	**96.6**	90.3	74.0	**96.2**	**90.3**	83.5	73.0
051_large_clamp	72.3	61.2	80.1	62.8	**96.9**	91.8	96.7	**92.5**	81.0	60.1	**93.2**	**84.5**	91.6	75.4
052_extra_large_clamp	96.6	93.7	69.7	67.5	94.3	85.9	**97.3**	**93.4**	85.1	44.4	91.0	71.1	**91.6**	**82.5**
061_foam_brick	98.1	96.8	86.5	70.0	**98.5**	96.2	98.3	**96.8**	**98.2**	89.8	97.7	**96.1**	97.8	91.5
mean	90.2	78.0	74.3	59.2	96.5	86.4	**97.5**	**93.7**	80.9	57.9	93.8	84.8	**95.8**	**87.1**

Table A4: Tracking performance on the YCB-Video full video sequences. Results of Wuthrich (Wüthrich et al. 2013), RGF (Issac et al. 2016), ICG (Stoiber, Sundermeyer, and Triebel 2022), FoundationPose(Wen et al. 2024),One2Any(Liu et al. 2025) are taken from (Liu et al. 2025).

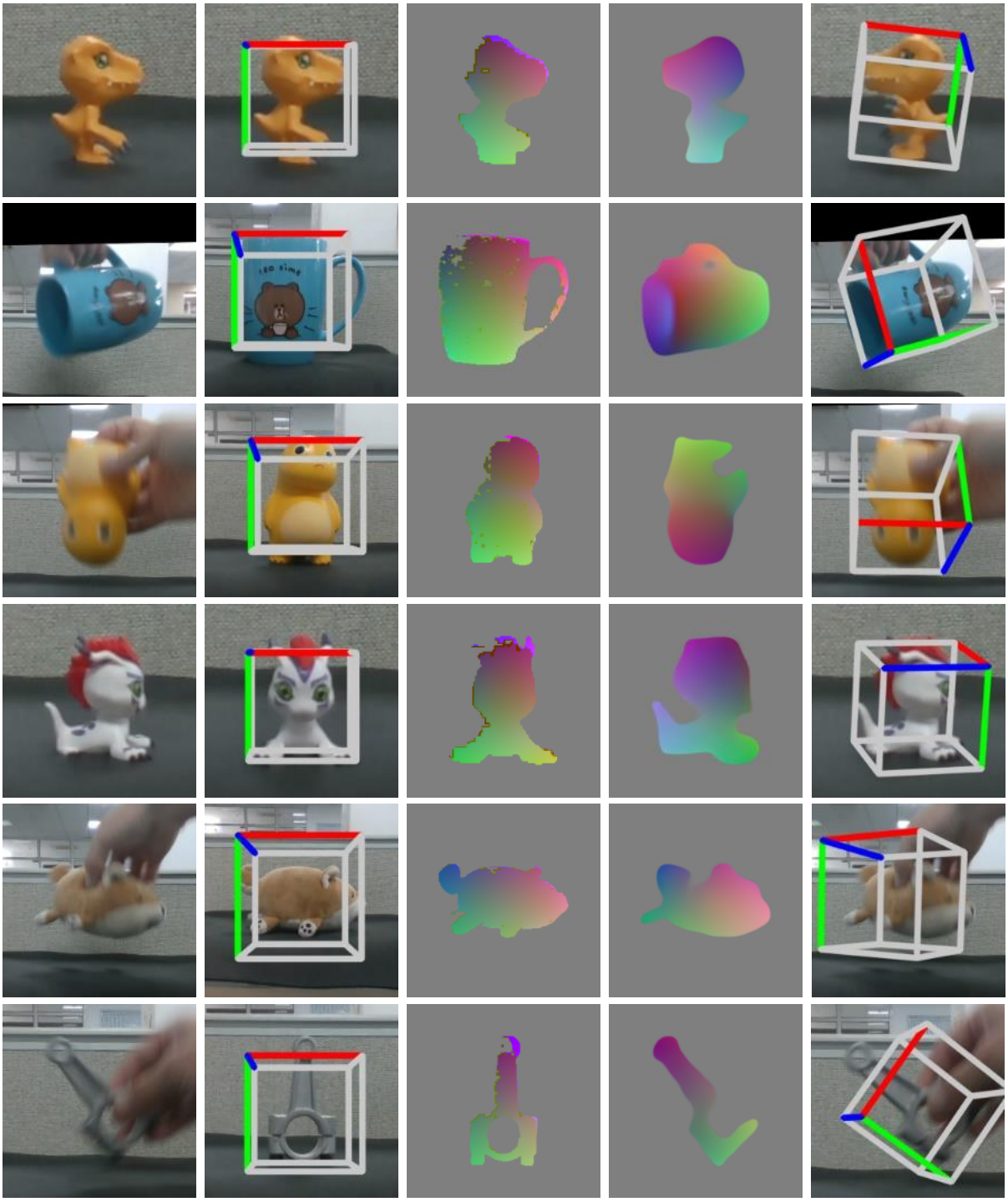

Figure A6: Qualitative results on the self-collected data. Our method demonstrates robust pose estimation capabilities under significant viewpoint variations.

Figure A7: Limitations. The predicted ROC map may varies under different generation orders.

References

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Everingham, M.; Gool, L. V.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2010. The PASCAL Visual Object Classes (VOC) Challenge. arXiv:0909.5206.

He, Y.; Wang, Y.; Fan, H.; Sun, J.; and Chen, Q. 2022. Fs6d: Few-shot 6d pose estimation of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6814–6824.

Hinterstoisser, S.; Holzer, S.; Cagniart, C.; Ilic, S.; Konolige, K.; Navab, N.; and Lepetit, V. 2011. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 international conference on computer vision*, 858–865. IEEE.

Hodaň, T.; Michel, F.; Brachmann, E.; Kehl, W.; Glent Buch, A.; Kraft, D.; Drost, B.; Vidal, J.; Ihrke, S.; Zabulis, X.; Sahin, C.; Manhardt, F.; Tombari, F.; Kim, T.-K.; Matas, J.; and Rother, C. 2018. BOP: Benchmark for 6D Object Pose Estimation. *European Conference on Computer Vision (ECCV)*.

Issac, J.; Wüthrich, M.; Cifuentes, C. G.; Bohg, J.; Trimpe, S.; and Schaal, S. 2016. Depth-based object tracking using a robust gaussian filter. In *2016 IEEE international conference on robotics and automation (ICRA)*, 608–615. IEEE.

Li, T.; Tian, Y.; Li, H.; Deng, M.; and He, K. 2024. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445.

Liu, M.; Li, S.; Chhatkuli, A.; Truong, P.; Van Gool, L.; and Tombari, F. 2025. One2Any: One-Reference 6D Pose Estimation for Any Object. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 6457–6467.

Oquab, M.; Darcet, T.; Moutakanni, T.; Vo, H.; Szafraniec, M.; Khalidov, V.; Fernandez, P.; Haziza, D.; Massa, F.; El-Nouby, A.; et al. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.

Smith, L. N.; and Topin, N. 2019. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, 369–386. SPIE.

Stoiber, M.; Sundermeyer, M.; and Triebel, R. 2022. Iterative corresponding geometry: Fusing region and depth for highly efficient 3d tracking of textureless objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6855–6865.

Tian, K.; Jiang, Y.; Yuan, Z.; Peng, B.; and Wang, L. 2024. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37: 84839–84865.

Wang, H.; Sridhar, S.; Huang, J.; Valentin, J.; Song, S.; and Guibas, L. J. 2019. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2642–2651.

Wen, B.; Yang, W.; Kautz, J.; and Birchfield, S. 2024. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17868–17879.

Wright, L. 2019. Ranger - a synergistic optimizer. `https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer`.

Wüthrich, M.; Pastor, P.; Kalakrishnan, M.; Bohg, J.; and Schaal, S. 2013. Probabilistic object tracking using a range camera. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, 3195–3202. IEEE.

Xiong, J.; Liu, G.; Huang, L.; Wu, C.; Wu, T.; Mu, Y.; Yao, Y.; Shen, H.; Wan, Z.; Huang, J.; et al. 2024. Autoregressive models in vision: A survey. *arXiv preprint arXiv:2411.05902*.

Yang, J.; Gao, M.; Li, Z.; Gao, S.; Wang, F.; and Zheng, F. 2023. Track Anything: Segment Anything Meets Videos. arXiv:2304.11968.