Robust 6DoF Pose Estimation Against Depth Noise and a Comprehensive Evaluation on a Mobile Dataset

Supplementary Material

1. Related Work

6DoF Pose Estimation Algorithms. The majority of data-driven approaches for object pose estimation revolve around utilizing either RGB images [18, 27, 34, 35] or RGBD images [11, 12, 16, 25, 31] as their input source. RGBD remains mainstream in industrial environments requiring higher precision. However, due to the high cost of accurate depth sensors, finding more robust solutions compatible with inexpensive and widely used sensors is a problem we aim to address.

Methods [11, 12, 16, 25, 31] that relied on depth maps advocated for the modality fusion of depth and RGB data to enhance inference capabilities. To effectively fuse multi-modalities, Wang et al. [31] introduced a network architecture capable of extracting and integrating dense feature embedding from both RGB and depth sources. Due to its simplicity, this method achieved high efficiency in predicting object poses. In more recent works [11–13], performance improvements were achieved through more sophisticated network architectures. For instance, He et al. [12] proposed an enhanced bidirectional fusion network for key-point matching, resulting in high accuracy on benchmarks such as YCB-Video [34] and LINEMOD [14]. However, these methods exhibited reduced efficiency due to the complex hybrid network structures and processing stages. Addressing symmetric objects, Mo et al. [25] proposed a symmetry-invariant pose distance metric to mitigate issues related to local minima. On the other hand, Jiang et al. [16] proposed an L1-regularization loss named abc loss, which enhanced pose estimation accuracy for non-symmetric objects.

Besides the RGBD approach, studies following the RGB-only approach often rely on incorporating additional prior information and inductive biases during the inference process. These requirements impose additional constraints on the application of 3D object tracking on mobile devices. Their inference process can involve utilizing more viewpoints for similarity matching [19, 23] or geometry reconstruction [28, 33], employing rendering techniques [3, 19, 26] based on precise 3D model or leveraging an additional database for viewpoint encoding retrieval [3]. During the training phase, these approaches typically draw upon more extensive datasets, such as synthetic datasets, to facilitate effective generalization within open-set scenarios. However, when confronted with a limited set of data samples, their performance does not surpass that of closed-set algorithms in cases where there is a surplus of prior information available and depth map loss.

3D Object Tracking Datasets Existing object pose estimation algorithms are predominantly tested on a limited set of real-world 3D object tracking datasets [4–7, 14, 15, 21, 22, 24, 34], which often employ depth-from-stereo sensors or time-of-flight (ToF) sensors for data collection. Datasets like YCB-Video [34], LINEMOD [14], StereoOBJ-1M [22], and TOD [21] utilize depth-from-stereo sensors, while TLess [15] and DTTD [7] deploy ToF sensors, specifically the Microsoft Azure Kinect, to capture meter-scale RGBD data. However, the use of cameras with depth-from-stereo sensors may not be an optimal platform for deploying AR software, because stereo sensors may degrade rapidly at longer distances [9] and may encounter issues with holes in the depth map when stereo matching fails. In our pursuit of addressing the limitations of existing datasets and ensuring a more realistic dataset captured with mobile devices, we opt to collect RGBD data using the iPhone 14 Pro.

iPhone-based Datasets for 3D Applications. Several datasets utilize the iPhone as their data collection device for 3D applications, such as ARKitScenes [1], MobileBrick [20], ARKitTrack [36], and RGBD Dataset [10]. These datasets were constructed to target applications from 3D indoor scene reconstruction, 3D ground-truth annotation, depth-map pairing from different sensors, to RGBD tracking in both static and dynamic scenes. However, most of these datasets did not specifically target the task of 6DoF object pose estimation. Our dataset provides a distinct focus on this task, offering per-pixel segmentation and pose labels. This enables researchers to delve into the 3D localization tasks of objects with a dataset specifically designed for this purpose. The most relevant work is from OnePose [28], which is an RGBD 3D dataset collected by iPhone. However, their dataset did not provide 3D models for close-set settings, and they utilized automatic localization provided by ARKit for pose annotation, which involved non-trivial error for high-accuracy 6DoF pose estimation. On the other hand, we achieve higher localization accuracy with the OptiTrack professional motion capture system to track the iPhone camera's real-time positions as it moves in 3D.

Figure 1. **Sample visualizations of our dataset.** *First row:* Annotations for 3D bounding boxes. *Second row:* Corresponding semantic segmentation labels. *Third row:* Zoomed-in LiDAR depth visualizations.

2. More Dataset Description

2.1. Data Acquisition

Apple's ARKit framework[1] enables us to capture RGB images from the iPhone camera and scene depth information from the LiDAR scanner synchronously. We leverage ARKit APIs to retrieve 1920×1440 RGB images and 256×192 depth maps at a capturing rate of 30 frames per second. Despite the resolution difference, both captured RGB images and depth maps match up in the aspect ratio and describe the same scene. Alongside each captured frame, DTTD-Mobile stores the camera intrinsic matrix and lens distortion coefficients, and also stores a 2D confidence map describing how the iPhone depth sensor is confident about the captured depth at the pixel level. In practice, we disabled the auto-focus functionality of the iPhone camera during data collection to avoid drastic changes in the camera's intrinsics between frames, and we resized the depth map to the RGB resolution using nearest neighbor interpolation to avoid depth map artifacts.

To track the iPhone's 6DoF movement, we did not use the iPhone's own world tracking SDK. Instead, we follow the same procedure as in [7] and use the professional OptiTrack motion capture system for higher accuracy. For label generation, we also use the open-sourced data annotation pipeline provided by [7] to annotate and refine ground-truth poses for objects in the scenes along with per-pixel semantic segmentation. Some visualizations of data samples are illustrated in Fig. 1. Notice that the scenes cover various real-world occlusion and lighting conditions with high-quality annotations. Following previous dataset protocols [7, 34], we also provide synthetic data for scene augmentations used for training.

[1]https://developer.apple.com/documentation/arkit/

The dataset also provides 3D models of the 18 objects as illustrated in the main paper. These models are reconstructed using the iOS Polycam app via access to the iPhone camera and LiDAR sensors. To enhance the models, Blender [2] is employed to repair surface holes and correct inaccurately scanned texture pixels.

2.2. Train/Test Split

DTTD-Mobile offers a suggested train/test partition as follows. The training set contains 8622 keyframes extracted from 88 video sequences, while the testing set contains 1239 keyframes from 12 video sequences. To ensure a representative distribution of scenes with occluded objects and varying lighting conditions, we randomly allocate them across both the training and testing sets. Furthermore, for training purposes of scene augmentations, we provide 20,000 synthetic images by randomly placing objects in scenes using the data synthesizer provided in [7].

3. More Implementation Details

3.1. Details on RGBD Feature Fusion

Attention Mechanism. For both modality fusion and point-wise fusion stage, the scaled dot-product attention is utilized in the self-attention layers:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})_i = \sum_j \frac{\exp(\mathbf{q}_i^T \mathbf{k}_j / \sqrt{d_{\text{head}}})}{\sum_k \exp(\mathbf{q}_i^T \mathbf{k}_k / \sqrt{d_{\text{head}}})} \mathbf{v}_j, \tag{1}$$

where query, key, value, and similarity score are denoted as q, k, v, and s. The distinction between two fusion stages lies in the token preparation prior to the linear projection layer. It results in varying information contained within the query, key, and value.

[2]https://www.blender.org/

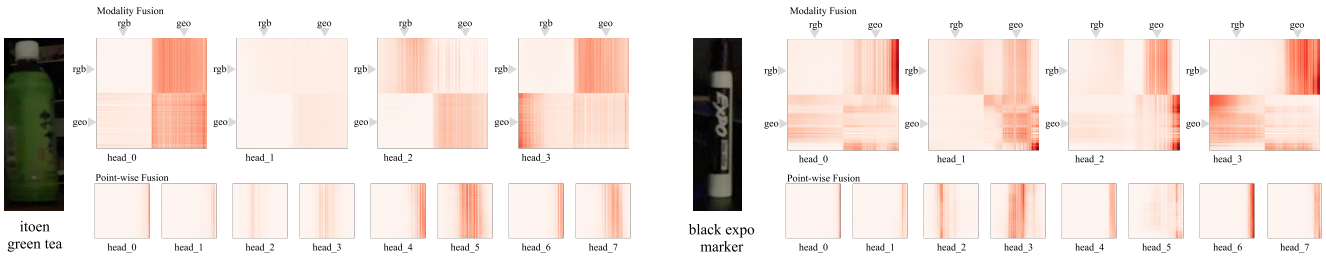

Figure 2. Examples of attention map output visualize of both modality fusion stage (the larger maps in the first row) and point-wise fusion stage (the smaller ones in the second row) on two objects (*itogen_green_tea* and *black_marker*). Due to the different ways we concentrate features in the two fusion stages, the token sequence length in modality fusion is twice that in the point-wise fusion process. For the attention maps produced in the final layer of modality fusion and point-wise fusion, they are of sizes 2000×2000 and 1000×1000, respectively.

The key idea in the first fusion stage is to perform local per-point fusion in a cross-modality manner so that we can make predictions based on each fused feature. Each key or query carries only one type of modal information before fusion, allowing different modalities to equally interact with each other through dot-product operations. It exerts a stronger influence when the RGB and geometric representations produce higher similarity.

In the second stage, where we integrate two original single-modal features with the first-stage feature into each point, we calculate similarities solely among different points. The key idea is to enforce attention layers to further capture potential relationships among multiple local features. A skip connection is employed in a concentrating manner between two fusion outputs so that we can make predictions based on per-point features generated in both the first and second stages.

Modality Fusion. The objective of this module is to combine geometric embedding g and RGB embedding c produced by single-modal encoders in a cross-modal fashion. Drawing inspiration from ViLP [17], both types of embedding are linearly transformed into a token sequence ($\in \mathbb{R}^{N \times d_{emb}}$). Before entering the modality fusion module E_1, these features are combined along the sequence length direction, i.e., all feature embedding is concentrated into a single combined sequence, where the dimension remains d_{emb}, and the sequence length becomes twice the original length.

$$f_1 = E_1 \left[c \oplus g \right] \in \mathbb{R}^{d_{f_1} \times 2N} \tag{2}$$

where the operation symbol "$\oplus$" denotes concentrating along the row direction. It is then reshaped into the sequence f_1' with the length of N and dimension of $2d_{f_1}$ in order to adapt the point-wise transformer encoder in the next fusion stage. This step enables the model's attention mechanism to effectively perform cross-modal fusion tasks.

Point-Wise Fusion. The goal of this stage is to enhance the integration of information among various points. The primary advantage of our method over the previous work [12] is that our model can calculate similarity scores not only with the nearest point but also with all other points, allowing for more comprehensive interactions. In order to enable the point-wise fusion to effectively capture the similarities between different points, we merge the original RGB token sequence c and the geometric token sequence g together with the output embedding sequence m' from the modality fusion module along the feature dimension direction. The combined sequence input $\left[c^T \oplus g^T \oplus (f_1')^T\right]^T \in \mathbb{R}^{(2d_{emb}+2d_{f_1}) \times N}$ is then fed into the point-wise transformer encoder E_2 to acquire the final fusion:

$$f_2 = E_2 \left[c^T \oplus g^T \oplus (f_1')^T\right]^T \in \mathbb{R}^{d_{f_2} \times N} \tag{3}$$

Attention Map Visualization. To visualize what our fusion module learns during the training process, we draw on previous studies [8, 29] and represent our attention map as $a_{i,j}$ described in section 3.1. Taking two objects (*itoen_green_tea* and *black_marker*) as examples, Fig. 2 displays the attention maps produced by different attention heads in the two fusion stages. We showcase the attention maps generated by the modality fusion and point-wise fusion at their respective final layers. The modality fusion part reveals distinct quadrant-like patterns, reflecting differences in how the two modalities fuse. The lower-left and upper-right quadrants offer insights into the degree of RGB and geometric feature fusion. The point-wise fusion part exhibits a striped pattern and shows that it attends to the significance of specific tokens during training.

3.2. Hyperparameters

Details on Fusion Stages' Hyperparameters. We extracted 1000 of pixels from the decoded RGB representation corresponding to the same number of points in the LiDAR point set. Both extracted RGB and geometric features are linear projected to 256-D before fused together. In the final experiment results, we utilized an 8-layer transformer encoder with 4 attention heads for the modality fusion stage

Table 1. **Comparison with diverse 6DoF pose estimation baselines on YCB video dataset.** We evaluate the results as the prior works [31] using ADD-S AUC and ADD-S (2cm) on all 21 objects, higher is better. Note that the left-most column indicates the per-object *depth-ADD* error. Objects with names in bold are symmetric.

	depth-ADD	DenseFusion [31]		MegaPose-RGBD [19]		ES6D [25]		BundleSDF [32]		DTTDNet (Ours)	
Object	Average	ADD-S AUC	ADD-S (2cm)	ADD-S AUC	ADD-S (2cm)	ADD-S AUC	ADD-S (2cm)	ADD-S AUC	ADD-S (2cm)	ADD-S AUC	ADD-S (2cm)
master_chef_can	0.005	95.20	**100.00**	79.11	69.88	82.47	73.56	**97.05**	**100.00**	96.32	**100.00**
cracker_box	0.005	92.50	**99.30**	74.98	80.65	81.09	84.68	90.69	87.67	**92.92**	98.04
sugar_box	0.009	95.10	**100.00**	81.42	90.95	95.97	97.80	**97.79**	**100.00**	96.76	**100.00**
tomato_can	0.011	93.70	96.90	86.11	94.79	89.02	92.71	68.27	70.00	**96.69**	**99.17**
mustard_bottle	0.005	95.90	**100.00**	87.41	99.72	93.13	87.11	**98.21**	**100.00**	97.39	**100.00**
tuna_can	0.013	94.90	**100.00**	91.03	**100.00**	74.86	74.22	91.11	**100.00**	**95.78**	**100.00**
pudding_box	0.005	94.70	**100.00**	89.65	**100.00**	90.13	98.60	**97.67**	**100.00**	93.24	95.33
gelatin_box	0.011	95.80	**100.00**	87.17	99.07	97.39	**100.00**	**98.46**	**100.00**	97.97	**100.00**
potted_meat	0.008	90.10	**93.10**	77.88	80.81	78.56	75.46	62.00	58.22	**93.56**	92.04
banana	0.007	91.50	93.90	76.18	71.77	92.83	84.70	**97.72**	**100.00**	94.52	**100.00**
pitcher_base	0.007	94.60	**100.00**	91.26	**100.00**	93.67	90.18	**96.53**	**100.00**	95.76	**100.00**
bleach_cleanser	0.007	**94.30**	**99.80**	82.97	74.05	88.12	87.76	69.67	71.72	93.30	99.61
bowl	0.021	86.60	69.50	83.80	59.61	2.68	0.00	**97.57**	**100.00**	84.33	51.23
mug	0.010	95.50	**100.00**	86.63	97.64	88.58	89.15	**97.09**	**100.00**	97.00	99.53
power_drill	0.010	92.40	97.10	88.86	97.73	85.43	78.62	**97.17**	**99.81**	95.57	**99.81**
wood_block	0.007	85.50	**93.40**	35.55	0.41	29.56	1.24	19.57	0.00	**87.63**	90.50
scissors	0.010	**96.40**	**100.00**	26.52	7.18	33.38	27.07	93.25	97.24	71.68	20.99
large_marker	0.011	94.70	**99.20**	83.02	67.13	90.08	87.96	95.08	93.83	**95.66**	97.22
large_clamp	0.011	71.60	78.50	85.93	90.03	43.74	17.84	**96.77**	99.16	90.99	**99.44**
extra_large_clamp	0.012	69.00	69.50	76.49	88.32	66.77	69.35	**95.15**	**100.00**	89.70	93.11
foam_brick	0.007	92.40	**100.00**	84.29	92.36	26.11	27.08	0.00	0.00	**95.63**	**100.00**
Average	0.009	91.20	95.30	82.64	84.81	78.14	75.35	86.31	87.64	**94.19**	**96.14**

and a 4-layer transformer encoder with 8 attention heads for the point-wise fusion stage.

Training Strategies. For our DTTDNet, learning rate warm-up schedule is used to ensure that our transformer-based model can overcome local minima in early stage and be more effectively trained. By empirical evaluation, in the first epoch, the learning rate lr linearly increases from 0 to $1e{-}5$. In the subsequent epochs, it is decreased using a cosine scheduler to the end learning rate $min\_lr = 1e{-}6$. Additionally, following the approach of DenseFusion [31], we also decay our learning rate by a certain ratio when the average error is below a certain threshold during the training process. Detailed code and parameters will be publicly available in our code repository. Moreover, we set the importance factor λ of Chamfer distance loss to 0.3 and the initial balancing weight w to 0.015 by empirical testing.

4. More Experimental Details

4.1. Baseline Implementation Details

For all the baseline methods [19, 25, 31, 32] that we adopted, we did not integrate any additional iterative refinement processes (e.g., ICP [2]) for fair comparison.

ES6D [25]. We preprocessed the training datasets including both DTTD-Mobile and YCB video according to the original paper and official codebase of ES6D [25], including removing some high-noise data, normalizing 3D translation, averaging the xyz map, and filtering out outliers in the point cloud. For the test set, to ensure fair comparison with other baselines, we did not adopt certain noise reduction methods that require prior knowledge based on ground truth data, nor did we exclude some high-noise data samples. We ensured that all test samples were retained and had errors calculated as those in other baselines.

BundleSDF [32]. The object-centric camera pose coordinates outputted by BundleSDF [32] are based on its own embedded geometric reconstruction. In order to compute metrics in the same coordinate system and with the same CAD models as other baselines, we aligned the camera trajectory computed for each scene-object combination with the ground truth camera pose through trajectory-wise alignment based on the Umeyama algorithm [30].

4.2. More Results on YCB video Dataset

Due to the lower depth noise in the YCB video dataset (YCB's average *depth-ADD* is 0.009, while DTTD-Mobile's average *depth-ADD* is 0.239), we adopted a simplified DTTDNet model structure, omitting the GFF module, to accelerate training convergence. Additionally, the reference point set used for computing Chamfer distance loss was directly extracted from the depth map. Furthermore, regarding hyper-parameters, we chose 0 layers for modality fusion and 6 self-attention layers for point-wise fusion.

When examining the performance of each baseline on the YCB video dataset, as shown in **??**. DTTDNet achieves the highest average performance, with an ADD-S AUC of 94.19 and an ADD-S (2cm) of 96.14. DenseFusion [31] is the second best, with an ADD-S AUC of 91.20 and an ADD-S (2cm) of 95.30. Although BundleSDF [32] shows strong performance across many object classes, it struggles with pose estimation for some objects, primarily due to its inability to reconstruct 3D models in the presence of occlusions. Its ADD-S AUC is 86.31, and its ADD-S (2cm) is

87.64. In contrast, MegaPose-RGBD [19] and ES6D [25] lag behind in performance, particularly in the number of object classes in which they perform the best, with ADD-S AUC scores of 82.64 and 78.14, and ADD-S (2cm) scores of 84.81 and 75.35, respectively.

References

[1] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, et al. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. *arXiv preprint arXiv:2111.08897*, 2021. 1

[2] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, pages 586–606. Spie, 1992. 4

[3] Dingding Cai, Janne Heikkilä, and Esa Rahtu. Ove6d: Object viewpoint encoding for depth-based 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6803–6813, 2022. 1

[4] 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, 2015. 1

[5] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. *IEEE Robotics & Automation Magazine*, 22 (3):36–52, 2015.

[6] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3):261–268, 2017.

[7] Weiyu Feng, Seth Z. Zhao, Chuanyu Pan, Adam Chang, Yichen Chen, Zekun Wang, and Allen Y. Yang. Digital twin tracking dataset (dttd): A new rgb+depth 3d dataset for longer-range object tracking applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3288–3297, 2023. 1, 2

[8] Yong Guo, David Stutz, and Bernt Schiele. Robustifying token attention for vision transformers, 2023. 3

[9] Hussein Haggag, Mohammed Hossny, D. Filippidis, Douglas C. Creighton, Saeid Nahavandi, and Vinod Puri. Measuring depth accuracy in rgbd cameras. *2013, 7th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–7, 2013. 1

[10] Lingzhi He, Hongguang Zhu, Feng Li, Huihui Bai, Runmin Cong, Chunjie Zhang, Chunyu Lin, Meiqin Liu, and Yao Zhao. Towards fast and accurate real-world depth super-resolution: Benchmark dataset and baseline. *arXiv preprint arXiv:2104.06174*, 2021. 1

[11] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[12] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 3

[13] Yisheng He, Yao Wang, Haoqiang Fan, Jian Sun, and Qifeng Chen. Fs6d: Few-shot 6d pose estimation of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6814–6824, 2022. 1

[14] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Computer Vision – ACCV 2012*, pages 548–562, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 1

[15] Tomáš Hodaň, Pavel Haluza, Štěpán Obdržálek, Jiří Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017. 1

[16] Xiaoke Jiang, Donghai Li, Hao Chen, Ye Zheng, Rui Zhao, and Liwei Wu. Uni6d: A unified cnn framework without projection breakdown for 6d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11174–11184, 2022. 1

[17] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision, 2021. 3

[18] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1

[19] 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, 2022. 1, 4, 5

[20] Kejie Li, Jia-Wang Bian, Robert Castle, Philip HS Torr, and Victor Adrian Prisacariu. Mobilebrick: Building lego for 3d reconstruction on mobile devices. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4892–4901, 2023. 1

[21] Xingyu Liu, Rico Jonschkowski, Anelia Angelova, and Kurt Konolige. Keypose: Multi-view 3d labeling and keypoint estimation for transparent objects. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020)*, 2020. 1

[22] Xingyu Liu, Shun Iwase, and Kris M. Kitani. Stereobj-1m: Large-scale stereo image dataset for 6d object pose estimation. In *ICCV*, 2021. 1

[23] Yuan Liu, Yilin Wen, Sida Peng, Cheng Lin, Xiaoxiao Long, Taku Komura, and Wenping Wang. Gen6d: Generalizable model-free 6-dof object pose estimation from rgb images, 2023. 1

[24] Pat Marion, Peter R Florence, Lucas Manuelli, and Russ Tedrake. Label fusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3325–3242. IEEE, 2018. 1

[25] Ningkai Mo, Wanshui Gan, Naoto Yokoya, and Shifeng Chen. Es6d: A computation efficient and symmetry-aware 6d pose regression framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6718–6727, 2022. 1, 4, 5

[26] Van Nguyen Nguyen, Yinlin Hu, Yang Xiao, Mathieu Salzmann, and Vincent Lepetit. Templates for 3d object pose estimation revisited: Generalization to new objects and robustness to occlusions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6771–6780, 2022. 1

[27] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019. 1

[28] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. OnePose: One-shot object pose estimation without CAD models. *CVPR*, 2022. 1

[29] Asher Trockman and J. Zico Kolter. Mimetic initialization of self-attention layers, 2023. 3

[30] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380, 1991. 4

[31] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. 2019. 1, 4

[32] Bowen Wen, Jonathan Tremblay, Valts Blukis, Stephen Tyree, Thomas Muller, Alex Evans, Dieter Fox, Jan Kautz, and Stan Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. *CVPR*, 2023. 4

[33] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. *arXiv preprint arXiv:2312.08344*, 2023. 1

[34] 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

[35] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. pages 1941–1950, 2019. 1

[36] Haojie Zhao, Junsong Chen, Lijun Wang, and Huchuan Lu. Arkittrack: A new diverse dataset for tracking using mobile rgb-d data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5126–5135, 2023. 1