Supplementary Material for ZePHyR: Zero-shot Pose Hypothesis Rating

Brian Okorn*, Qiao Gu*, Martial Hebert and David Held

APPENDIX

A. *Ground Truth Translation Results*

The Multipath Augmented Autoencoders [1] baseline assumes that the object is cropped from the scene prior to input. In contrast, the focus of ZePHyR is to perform zero-shot pose estimation in cluttered scenes which contain multiple objects. In such cluttered scenes, finding the correct object crop of a novel object is non-trivial.

In BOP leaderboard[1], Multipath Autoencoders [1] reports their performance with the assistance of a dataset-wise trained MaskRCNNs as a segmentation networks. Consider that the main contribution of [1] is learning rotation encoding that generalizes over objects, we resolve the scale ambiguity and isolate the orientation error by providing this network with the ground truth translation for each object at test time. As shown in Table I, our method still outperforms [1], especially on the YCB-V dataset where most objects have rotational symmetry.

Method	Multipath AutoEncoder		Ours
	w/o GT trans	w/ GT trans	w/o GT trans
YCB-V	0.289	0.355	0.516
LM-O	0.217	0.560	0.598

TABLE I: AR scores for different method with and without ground truth translation ("GT trans").

B. *Pose Hypothesis Ablations Results*

We test our scoring method on different subsets of pose hypotheses to explore our sensitivity to the hypothesis generation method. In Table. II, we report the AR scores of the Point Pair Features baseline ("PPF") [2], our scoring method using pose hypotheses generated only from PPF ("PPF+Scoring"), our scoring method using pose hypotheses generated only from SIFT feature matching ("SIFT+Scoring") and our scoring method using pose hypotheses generated from both PPF and SIFT ("Both+Scoring"). The results indicates that on the YCB-V dataset, where most objects have high-quality mesh models and rich textures, the SIFT feature matching method provides valuable pose hypotheses. When combining PPF and SIFT hypotheses with our scoring method, the results improve over using our scoring method with PPF hypotheses alone. LineMOD (LM-O), however, contains mostly low texture or textureless objects. For this dataset, SIFT hypotheses are less useful and adding them mildly reduces the accuracy of our method but needs more processing time.

Method	PPF	PPF+ZePHyR	SIFT+ZePHyR	Both+ZePHyR
YCB-V	0.344	0.458	0.390	0.516
LM-O	0.527	0.598	0.011	0.595

TABLE II: BOP AR scores for ZePHyR based on different hypothesis generation methods.

C. *Network Details*

1) PointNet++: As mentioned in Section III-D.3, we reduce the sizes of MLP and adjust parameters of original PointNet++ design, to enable the training of the whole network with 1100 pose hypotheses in 11 GB GPU memory. We uniformly downsample the object mesh models so that the leaf size for the voxel grid is 7 millimeter and each object has 1000 points on average, and further randomly subsampled the input points down to 2000 when the number of points in the downsampled object model still exceeds this number. The detailed network architecture is described as follows.

We use the single scale grouping (SSG) version of PointNet++. Following architecture protocol in [3], we denote SA(K,r,[$l_1, ..., l_d$]) as a set abstraction (SA) level with K local regions of ball radius r using PointNet of d fully connected layers with width l_i ($i = 1, ..., d$). SA([$l_1, ...l_d$]) represents a global set abstraction level that converts set to a single vector. FC(l,dp) represents a fully connected layer with width l and dropout ratio dp. All fully connected layers are followed by batch normalization [4] and ReLU activation functions, except for the last score prediction layer. The resulting PointNet++ architecture is as follows:

$$SA(128, 0.2, [16, 32]) \rightarrow SA(16, 0.5, [32.64]) \rightarrow$$
$$SA([64, 128]) \rightarrow FC(64, 0.4) \rightarrow FC(16, 0.4) \rightarrow FC(1)$$

2) PointNet: For the ablation experiment on PointNet in Section V-C, we also use a reduced version of Classification Network described in [5]. We remove the input transform and feature transform layers. We use a three-layer MLP, with the size of the hidden layer to be 16, pre-bottleneck, a bottleneck max pooling layer of dimension 16, and a 3-layer MLP with the hidden layer size 64 post-bottleneck. All except the last MLP layers are followed by a batch normalization layer [4] and a ReLU activation. The final output of the last layer estimates a single score for each input point cloud.

* indicates equal contribution.

B. Okorn, Q. Gu, M. Hebert and D. Held are with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213. (bokorn@andrew.cmu.edu, qiaog@andrew.cmu.edu, hebert@cs.cmu.edu, dheld@andrew.cmu.edu)

[1] https://bop.felk.cvut.cz/method_info/96/

3) Convolutional Network: For the CNN mentioned in Section V-C, we use a vanilla ResNet-18 [6] with no pretrained-weight. The the number of input channels of the first layer is expanded to match the number of error features, and the last layer is changed to a 2-layer MLP with the hidden layer size 64. The final output is a single score for each pose hypothesis.

D. Training Details

For computational efficiency, we subsample the training data points in the YCB-V and LM-O datasets and pre-process them for fast training. Specifically, from the YCB-V training split, we evenly sampled 4716 observations, containing 2346 observations of objects with even IDs and 2370 of objects with odd IDs. From the synthetic training set of LineMOD dataset [7], we evenly sampled 1749 observations of objects that are not in LM-O dataset as the training set. The observations of the training objects are then split, with 90% used for training and 10% used for validation. After training, the model weights at the epoch with lowest error on validation set of the "seen" objects are selected for evaluation, and the observations of "unseen" objects are not used during training or validation.

To train the PointNet and PointNet++ archetectures, we use an Adam optimizer [8] with an initial learning rate 3×10^{-4}. For the CNN training, the initial learning rate is 1×10^{-5}. We trained each network for 100 epochs and the learning rate reduces to $1/10$ after epoch 30 and 80.

We augment the training data by randomly jittering the brightness, contrast, saturation and hue of the observation images by factor of 0.2, 0.2, 0.2 and 0.05 respectively. To prevent overfitting to the training objects, we also jointly perturb the color of the model and the observation color, changing the color of both the real and rendered data in the same way. The factors for brightness, contrast, saturation and hue in this process are all 0.5.

E. Qualitative Results

Figure 1 shows the qualitative results of both our method and the baseline over the YCB-V and LM-O datasets. The left column shows the full scene; the second column shows the ground-truth pose for the target object. The third column shows the highest-scoring pose according to our method, and the last column shows the highest-scoring pose according to the PPF baseline [2]. In the 3rd and 4th columns, the selected pose hypothesis for each method is rendered into the frame.

Overall, Our method demonstrates a better performance than the PPF baseline. As PPF only considers geometry, it cannot determine the correct orientation on some objects that are symmetrical in shape but have distinguishing texture, like the "Master Chef" can and tomato can in row (5), (7) and (8) in Figure 1. But our method considers both shape and color information, and thus can make correct estimations in such cases. PPF also tends to match the flat side of an object to the flat top of a table, such shown in row (3), (6), (7) and (9) in Figure 1; our method fixes such errors.

Figure 1 also shows some cases where our method fails. In row (8), due to the over exposure on the surface of the sugar box, our method mixes the back side of the box with the front side. In row (7), our method fails to detect the "Soft Scrub" bottle probably because only its side is facing towards the camera, where almost no texture or color information is present. The toy cat in row (3) and the egg box in row (2) are two failure cases where the occlusion is so strong that the whole object is almost invisible.

F. Failure Case Analysis

Figure 2 further elaborates the failure case of the sugar box in the row (8) of Figure 1. As we can see, due to the reflection, the upper surface of the sugar box in the observation is overly lightened, which makes the saturation and value errors of the wrongly-picked hypothesis smaller than those of the correct one. However, our method correct recover the geometry and still presents a reasonable result.

G. Time-Accuracy Trade-off on LM-O dataset

In Table III, we report the detailed data for the time-accuracy trade-off curve in Figure 4 in the main paper. We here only vary the PPF parameters and thus its inference time. The speed of our scoring network (ZePHyR) is unchanged. In the table, "Model SD" and "Scene SD" are the sampling distance on the model point cloud and the scene point cloud respectively, relative to the model diameter. Higher numbers lead to smaller point clouds and faster processing times. "Ref Pt Rate" is the ratio of the points on the scene point cloud that are used as reference points when sampling point pairs [2]. "Dense Object PC" means the input object model to PPF is directly converted from the mesh model without downsampling. "Sparse Object PC" means PPF uses the downsampled object point cloud that is used in the scoring network, as described in Section -C.1. "Sparse" and "Dense" in "Refinement" column indicates the spacial density of the point cloud used for ICP step in PPF. We refer readers to [2] and [9] for more details.

Note that ZePHyR is a scoring network on the provided pose hypotheses, and in the table, our PPF+ZePHyR demonstrate a constant improvement over the PPF baseline by a large margin with only little time overhead. This means our method is able robustly pick better hypothesis from the PPF's output. Comparing the first and the third row in the table, we can find that PPF+ZePHyR achieves comparable results with PPF but is sped up by more than 3 times.

References

[1] M. Sundermeyer, M. Durner, E. Y. Puang, Z.-C. Marton, N. Vaskevicius, K. O. Arras, and R. Triebel, "Multi-path learning for object pose estimation across domains," in *CVPR*, June 2020.

[2] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3d object recognition," in *CVPR*, 2010, pp. 998–1005.

[3] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NeurIPS*, 2017, pp. 5099–5108.

[4] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

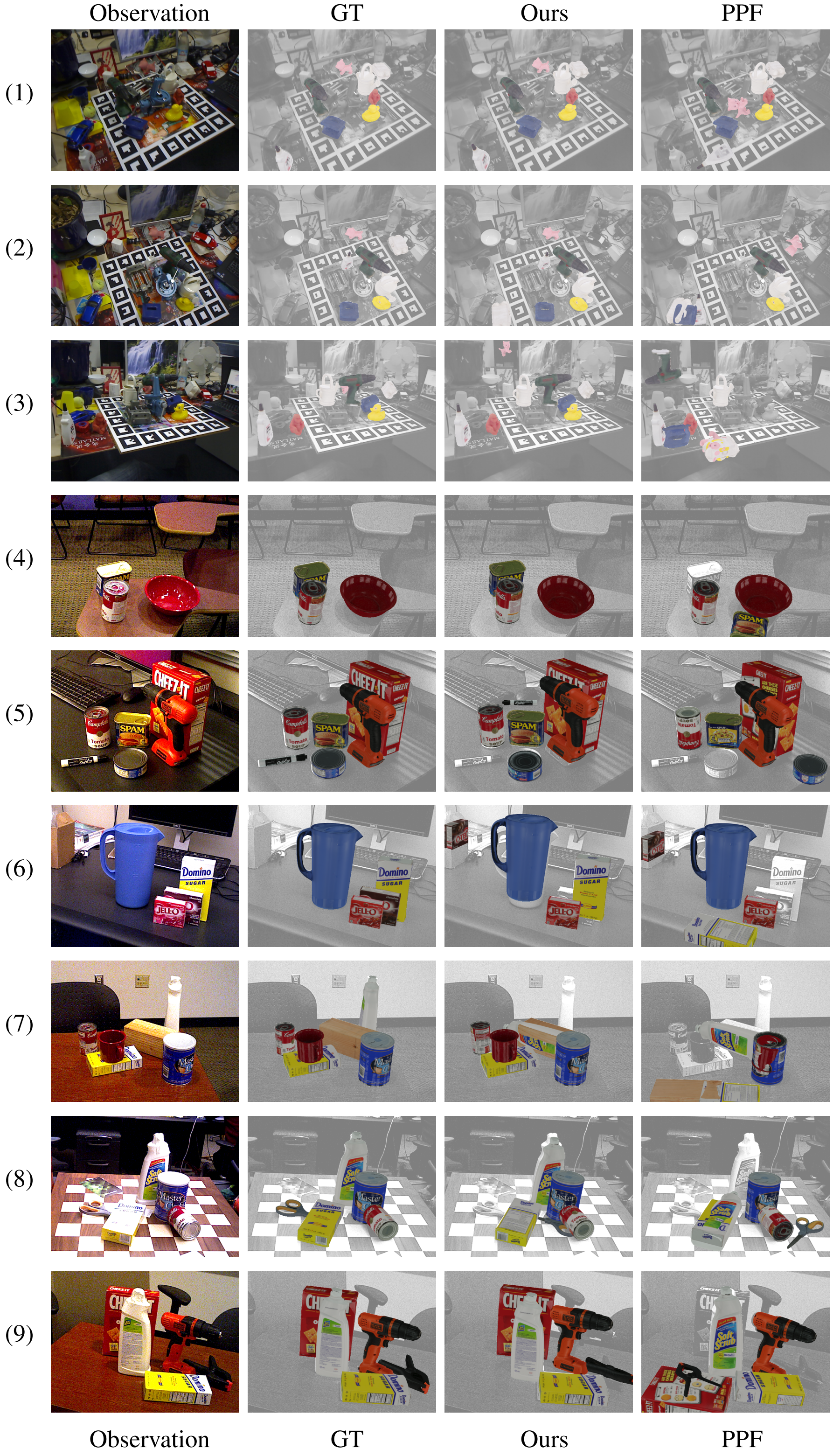
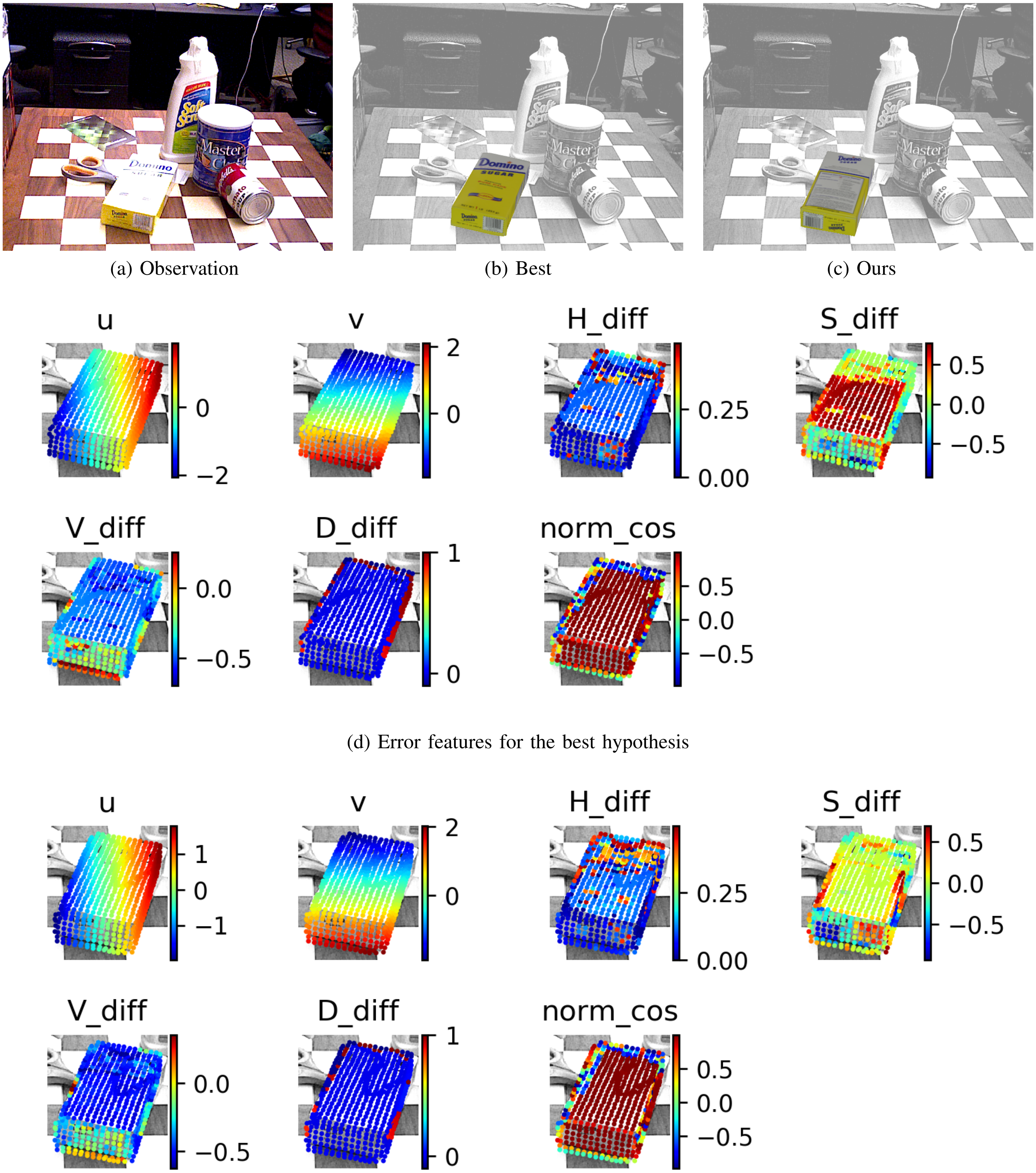


Fig. 1: Qualitative results on LM-O (first 3 rows) and YCB-V (last 6 rows) dataset. Raw input image and ground truth renders shown in the first and second column, respectively. The third and fourth column compare the top results using our scoring pipeline ("Ours") and the original PPF ("PPF") hypothesis algorithm [2], respectively.

[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, July 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[7] T. Hodaň, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. Sinha, and B. Guenter, "Photorealistic image synthesis



(a) Observation

(b) Best

(c) Ours

(d) Error features for the best hypothesis

(e) Error features for our result

Fig. 2: Failures case of our method. “Best” means the pose that has the lowest ADD error in the pose hypothesis set. “Ours” means the highest scoring hypothesis returned by our method. In plot (d) and (e), “u” and “v” are the normalized projection coordinates. “H_diff”, “S_diff”, “V_diff” and “D_diff” represent the signed difference of the hue, value, saturation and depth between projected model points and the observation respectively. “norm_cos” is the cosine of the angle between transformed model normal vectors and observed normal vectors.

Model SD	Scene SD	Ref Pt Rate	Object PC	Refinement	Time (PPF)	BOP score (PPF)	Time (PPF+ZePHyR)	BOP score (PPF+ZePHyR)
0.03	0.03	1	Dense	Dense	2.900	0.527	2.948	**0.598**
0.03	0.05	1	Dense	Sparse	1.626	0.502	1.674	0.571
0.05	0.05	1	Dense	Sparse	1.388	0.480	1.436	0.550
0.05	0.05	0.5	Dense	Sparse	0.794	0.463	0.842	0.524
0.05	0.07	0.5	Dense	Sparse	0.530	0.349	0.578	0.456
0.03	0.04	0.5	Sparse	Sparse	0.524	0.319	0.572	0.504
0.05	0.07	0.25	Dense	Sparse	0.315	0.303	0.363	0.408
0.03	0.04	0.2	Sparse	Sparse	0.257	0.297	0.305	0.484
0.03	0.05	0.2	Sparse	Sparse	0.219	0.253	0.267	0.441
0.05	0.05	0.2	Sparse	Sparse	0.200	0.213	0.248	0.379

TABLE III: Inference time and performance on the LM-O dataset of PPF and PPF+ZePHyR using different PPF settings.

for object instance detection," *ICIP*, 2019.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[9] M. S. GmbH, *find_surface_model [HALCON Operator Reference / Version 13.0.4]*, 2019 (accessed Nov 2nd, 2020), https://www.mvtec.com/doc/halcon/13/en/find_surface_model.html.