A. Supplementary

A.1. Alignment Ablation

In Table 5, we study the effect for different distance calculations between two vertices containing many features from many viewpoints. We observe, that averaging over the features in a single vertex before calculating the distance to another vertex reduces the performance drastically. Further, calculating the distance by averaging over the bi-directional nearest neighbor distances, slightly improves the performance compared to taking the minimum distance over the bi-directional nearest neighbor distances.

Besides that, we show that refining the initial alignment using few gradient-based optimization steps improves the results, especially with respect to the more fine-grained $10°$ and $15°$ accuracies.

In the same table, we observe the significant effect for using different feature extractors.

A.2. In-the-Wild 3D Pose Estimation Ablation

In Table 6, we ablate our 3D pose estimation method for various amount of training data. We show the results for maximum 5, 10, 20, or 50 videos per category.

A.3. Mesh Reconstruction from Videos

Using structure-from-motion [20] we obtain a point cloud $P = \{\mathrm{v}_i \in \mathbb{R}^3\}$ for each video. Further, we reconstruct a coarse mesh using three steps. First, we randomly downsample the point cloud to 20000 points and clean it using the object segmentation provided by CO3D. Therefore, we compute an average ratio of visibility for each point v_i by projecting it in all N frames, using the respective projection π_j for frame j, and averaging over the respective visibilities $\delta_j(\pi_j(\mathrm{v}_i))$ as follows

$$p(\mathrm{v}_i) = \frac{1}{N}\sum_{j=1}^{N}\delta_j(\pi_j(\mathrm{v}_i)). \tag{14}$$

Hereby we set $\delta_j(\pi_j(\mathrm{v}_i)) = 0$, if the projected vertex is not inside the frame. We filter out all points which ratio of visibility lies below 60%. Second, we use alpha shapes [11] to estimate a coarse shape from the clean point cloud. Figuratively speaking, this algorithm starts off with a convex volume and then iteratively carves out spheres while preserving all original points. We set the size of the sphere to 10 times the particle size, where the particle size is the average distance of each point to its 5th closest point. Third, we use quadratic mesh decimation [4] to end up with a maximum of 500 faces. This method iteratively contracts a pair of vertices, minimizing the projective error with the faces normals. All steps are visualized in Figure 6.

A.4. Videos Filtering

We filter out three types of videos. Type a), object is too far away from the camera. Type b), object is too close to the camera. Type c), the variance of viewpoints is too small. Type a) is not ideal because the point cloud and the images yield only few details of the object. Type b) is problematic because the close-ups prevent us from robustly cleaning the noisy point cloud as there is less information accumulated from the object segmentations. Type c) results in a very noisy or even broken structure-from-motion. For the identification of type a), object is too far away from the camera, we use the average object visibility $\bar{\delta}$ over all N frames width U and height V formally defined as

$$\bar{\delta} = \frac{1}{NUV}\sum_{j=1}^{N}\sum_{u=1}^{U}\sum_{v=1}^{V}\delta_j(u,v). \tag{15}$$

We require an average object visibility of at least 10%. A filtered out video is illustrated in Figure 7. For the identification of type b), the object is too close to the camera, we use the projection of the 3D center into all frames, expecting it to be in the center of the frames. We compute the 3D center c using the camera rays with position r_j and direction n_j by minimizing its projected distances to the rays

$$\min_c \sum_j \langle (c - r_j)^T n_j, (c - r_j)^T n_j \rangle. \tag{16}$$

It can be shown that this resolves to the following system of linear equations

$$\left(\sum_j n_j n_j^T\right) x = \sum_j n_j n_j^T r_j. \tag{17}$$

With the outer product $n_j n_j^T \in \mathbb{R}^{3\times 3}$. For a correct camera focus, we expect the projected 3D center to lie within the centered rectangle spanning 60% of the image width and height. In total, we require 80% of the frames to be focused on the 3D center. A negative example is provided in Figure 7. A filtered out video is illustrated in Figure 8.

For the identification of type c), the variance of viewpoints is too small, we subtract the center c of all camera positions r_j and normalize them to lie on the unit sphere. Further, we divide the unit sphere into 38 bins and calculate the viewpoint coverage as percentage of viewpoint bins covered. We require a viewpoint coverage for each video of 15%. A rejected video is shown in Figure 9.

A.5. ObjectNet3D

We report more qualitative results are visualized in Figure 10.

Feat. Encoder	Avg. Feat.	Vertex Feat. Dist.	Refine	Acc. 30°	Acc. 15°	Acc. 10°
DINOv2 ViT-S/14	✓	min-min		26.0	22.6	21.2
DINOv2 ViT-S/14		min-min		77.0	61.6	46.1
DINOv2 ViT-S/14		mean-min		79.1	62.6	47.7
DINOv2 ViT-S/14		mean-min	✓	79.9	69.7	56.9
DINOv2 ViT-B/14		mean-min	✓	84.3	73.4	60.6
DINOv1 ViT-S/8		mean-min	✓	74.4	63.0	51.8

Table 5. Ablation for 7D alignment on the CO3D dataset, averaged across 20 categories.

	# Videos	Acc. 30°	Acc. 15°	Acc. 10°
PASCAL3D+	5	56.3	33.2	21.6
	10	58.2	33.4	21.2
	20	65.2	38.9	25.0
	50	69.2	41.3	25.5
ObjectNet3D	5	45.3	21.0	11.7
	10	45.8	20.9	11.4
	20	49.1	24.1	13.8
	50	52.4	25.5	14.1

Table 6. Ablation for 3D pose estimation on PASCAL3D+ and ObjectNet3D.

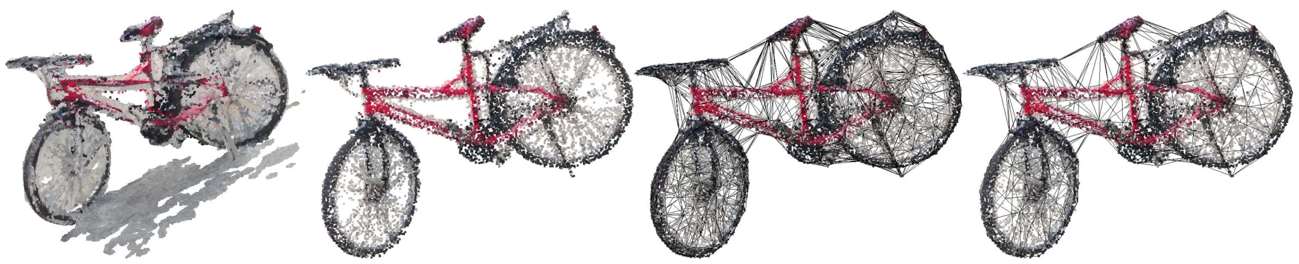
Figure 6. Steps of estimating a coarse mesh given a noisy point cloud (left). First, we clean the point cloud using the object segmentation for each frame. Second, we estimate a coarse mesh using alpha shapes [11]. Third, we remove faces using quadratic decimation [4].

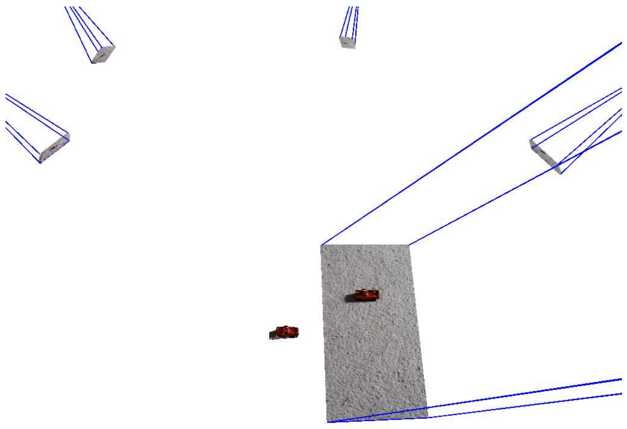

Figure 7. Rejected video type a), object is too far away from the camera.

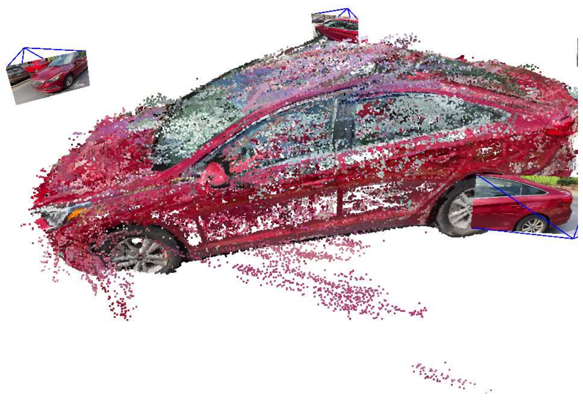

Figure 8. Rejected video type b), object is too close to the camera.

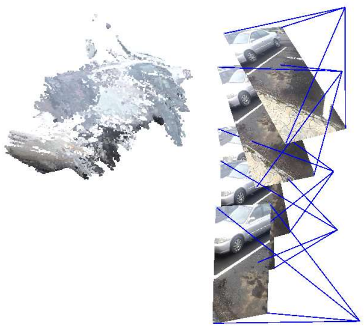

Figure 9. Rejected video type c), the variance of viewpoints is too small.

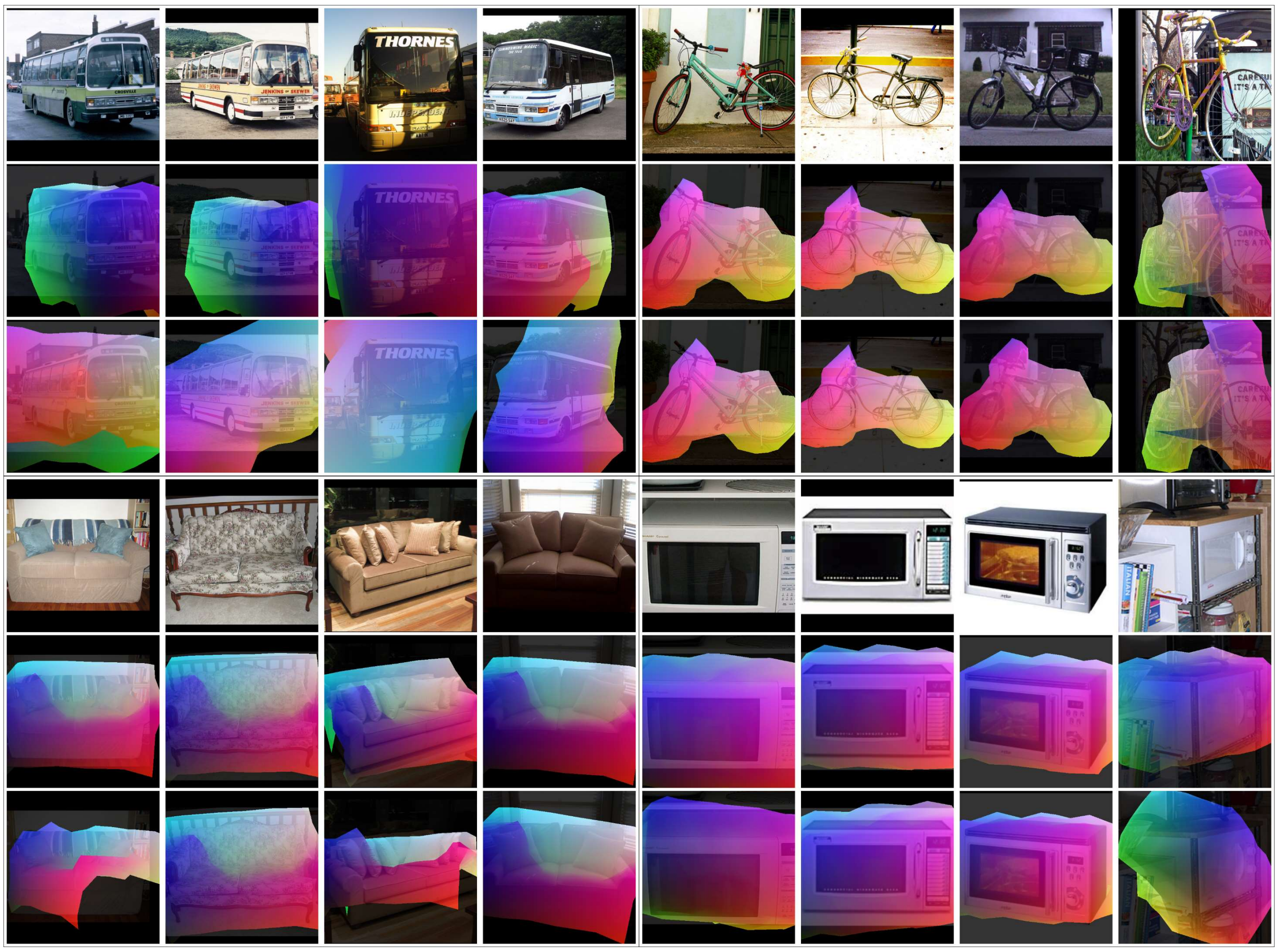

Figure 10. Qualitative comparison of our method (top) and ZSP (bottom) at category-level 3D pose prediction in the wild on samples from ObjectNet3D (we randomly selected the samples to demonstrate the diversity of the results). For both methods, we overlay our coarse mesh reconstruction in the predicted 3D pose.

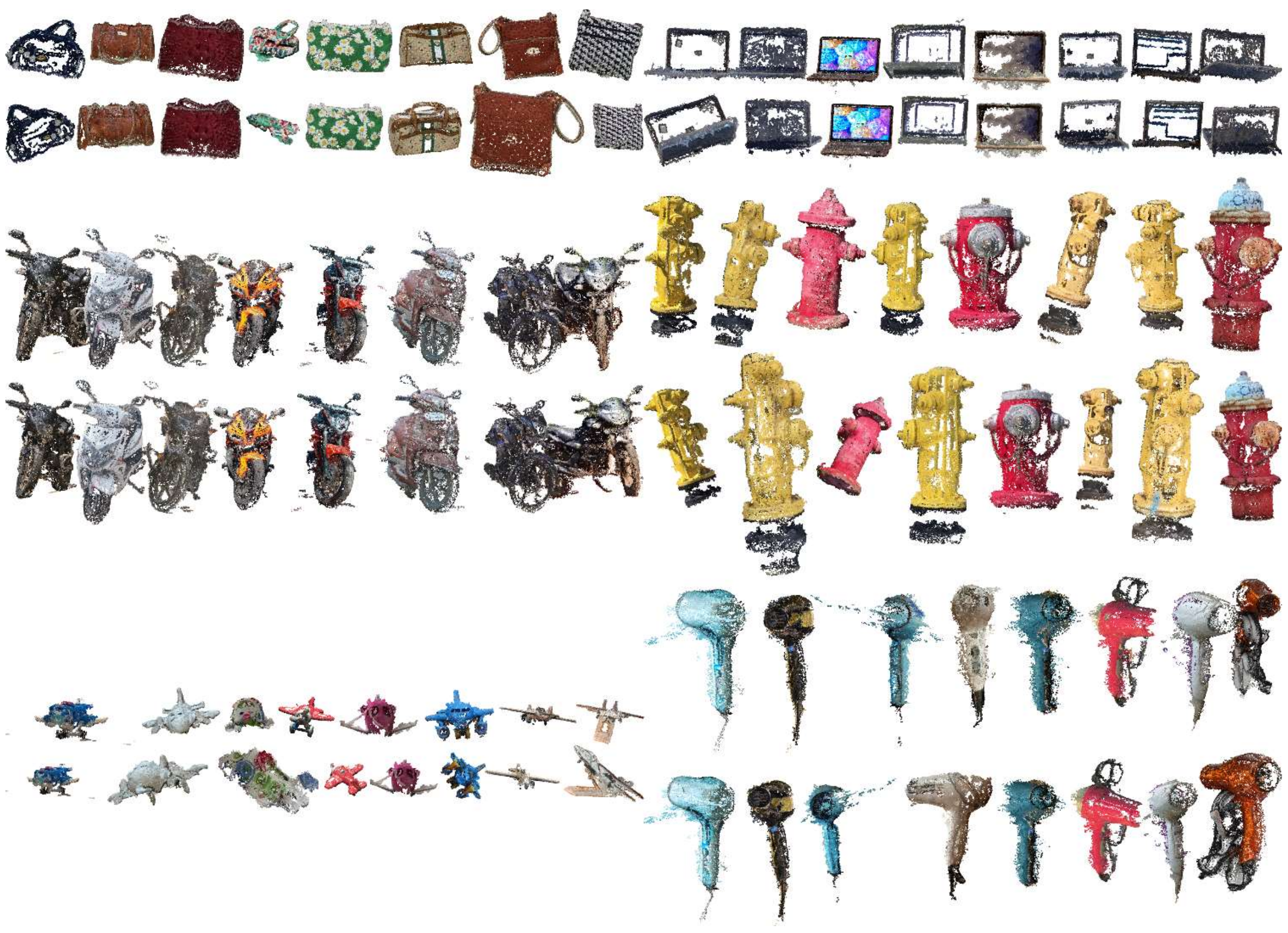

Figure 11. Qualitative comparison of two unsupervised alignment methods. The first row illustrates the alignment of our proposed method. The second row shows the alignment using ZSP [5]. For both methods, we utilize the 5th object instance from the left as a reference point. Our proposed method proves to be more precise than ZSP.