Supplementary Material

A. Rigid Body Transform Notation

Throughout the paper, we have regularly included rigid body transforms in many equations. Here, we briefly explain the notation. A 6DoF rigid body transform ${}^B_A\mathbf{T} \in SE(3)$ will transform a point defined in the reference frame $\{A\}$ into the reference frame $\{B\}$. We write this in two possible ways. For the first, and most common, we separate ${}^B_A\mathbf{T}$ into its rotational and positional components, ${}^B_A\mathbf{R} \in SO(3)$ and ${}^B\boldsymbol{p}_A \in \mathbb{R}^3$ respectively. In this form we write ${}^B\boldsymbol{p}_k = {}^B_A\mathbf{R}\,{}^A\boldsymbol{p}_k + {}^B\boldsymbol{p}_A$ to transform the 3D point ${}^A\boldsymbol{p}_k$ from the $\{A\}$ frame into the $\{B\}$ frame.

In the other form, which shows up in Eq. 4, we leave the transform in its full 4×4 $SE(3)$ form, and use the homogeneous form of translation vectors ${}^A\bar{\boldsymbol{p}}_k = [{}^A\boldsymbol{p}_k^\top \; 1]^\top$. In this way, we write ${}^B\bar{\boldsymbol{p}}_k = {}^B_A\mathbf{T}\,{}^A\bar{\boldsymbol{p}}_k$. This form specifically allows us to chain together multiple transformations with simplified notation, for example: ${}^B\bar{\boldsymbol{p}}_k = {}^B_{A_2}\mathbf{T}\,{}^{A_2}_{A_1}\mathbf{T}\,{}^{A_1}_{A}\mathbf{T}\,{}^A\bar{\boldsymbol{p}}_k$.

B. Choice of Symmetry Without Prior

As mentioned in Section 3.1, as opposed to the mirroring technique and additional symmetry classifier proposed by [28], we need to teach the network to predict the initial keypoints of symmetric objects, before the prior is available. We opt to utilize the set of symmetry transforms to solve this issue in a more concise manner with a simple intuition: when the prior detection is not available for a symmetric object, we can simply instruct the network to choose the orientation which brings the object pose closest to a canonical pose where the front of the object faces the camera, and the top of the object faces the top of the image. This intuition is learned by the network during training by choosing the symmetry for keypoint labels that brings the 3D keypoints closest (in orientation) to those transformed into the canonical view $\{O_c\}$ in the camera frame:

$$
{}^O_S\mathbf{T} = \underset{{}^O_{S_m}\mathbf{T}\in\mathcal{S}}{\operatorname{argmin}} \frac{1}{K}\sum_{k=1}^{K} \left\| {}^C\tilde{\boldsymbol{p}}_k - {}^C\tilde{\boldsymbol{p}}^c_k \right\|_2 \tag{6}
$$

$$
{}^C\boldsymbol{p}_k = {}^C_O\mathbf{R}\left({}^O_{S_m}\mathbf{R}\,{}^O\boldsymbol{p}_k + {}^O\boldsymbol{p}_{S_m}\right) \qquad {}^C\boldsymbol{p}^c_k = {}^C_{O_c}\mathbf{R}\,{}^O\boldsymbol{p}_k
$$

where $\tilde{\boldsymbol{p}}_k = \boldsymbol{p}_k - \frac{1}{K}\sum_{k=1}^{K}\boldsymbol{p}_k$ denotes the kth point of a mean-subtracted point cloud. We provide some visual examples of the effect of Eq. 6, which can be seen in Fig. 7. Remember that Eq. 6 is used to pick the symmetry transform to apply to the ground truth keypoints during training when the simulated prior detection is not given to the network – otherwise a random symmetry transform is applied to the prior and ground truth keypoints together so that the network can learn to follow the prior for the symmetry. The main effect of Eq. 6 is that it will choose the symmetry to apply to the ground truth keypoints that best matches the canonical view in terms of orientation, which essentially tells the network to always pick the symmetry that brings the front of the object closest to the camera and the top of the object closest to the negative y-axis of the camera frame (i.e., the top of the images) if no prior is given.

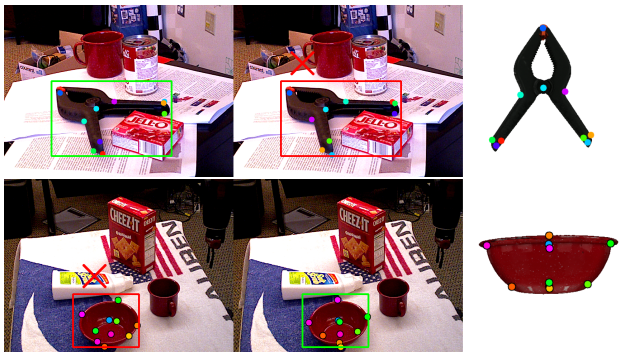

Figure 7. Examples of how we pick the symmetry to use for the keypoints during training when a prior detection is not given to the network. **Top:** the two possible symmetries for the clamp are shown on the left, and the keypoints in the canonical view are shown on the right. The first symmetry is chosen since the points are closer to the points in the canonical view. **Bottom:** the bowl has a continuous axis of symmetry about its vertical axis which are discretized into 64 symmetry transforms. For brevity we only show two – a random symmetry that is not chosen for the label (left) and one that is chosen (center) since it matches the canonical view (right) the best in terms of orientation. Best viewed in color.

Of course there is still the issue of detecting keypoints near the inflection point of a symmetry [28]. While we could utilize the mirroring technique of [28] to avoid this issue, we only need to detect keypoints once without the prior detection in practice (i.e., the first detection), and the mirroring technique of [28] requires an additional classifier during test time – which complicates the pipeline and adds additional computation. If the object is at an inflection point for the symmetry, and it is difficult to decide which symmetry to use, in our full SLAM system we can typically just reject bad measurements until the camera moves to a better viewpoint on the object in order for the network to more confidently choose the initial symmetry based on its training with Eq. 6.

C. Front-End Tracking Details

Besides the first image, whose camera frame becomes the global reference frame $\{G\}$, we need to estimate the camera pose ${}^C_G\mathbf{T}$ with the set of object PnP poses and the current estimates of the objects in the global frame. For each asymmetric object that is both detected in the current frame with a successful PnP pose ${}^C_O\mathbf{T}_{\text{pnp}}$ and has an estimated global pose ${}^G_O\mathbf{T}$, we can create a hypothesis about the current camera's pose as ${}^C_G\mathbf{T}_{\text{hyp}} = {}^C_O\mathbf{T}_{pnp}\,{}^G_O\mathbf{T}^{-1}$ and then project the 3D keypoints from all objects that have both

Figure 8. Our keypoint labels for the YCB-Video dataset. We labeled identifiable features based on the shape class of the objects (box-like, cylinder-like, and hand tool) which are common within different instances of the same shape class (such as box corners, cylinder top/bottom center, etc), and then instance-specific keypoints of other identifiable features such as brand names, bar codes, etc.

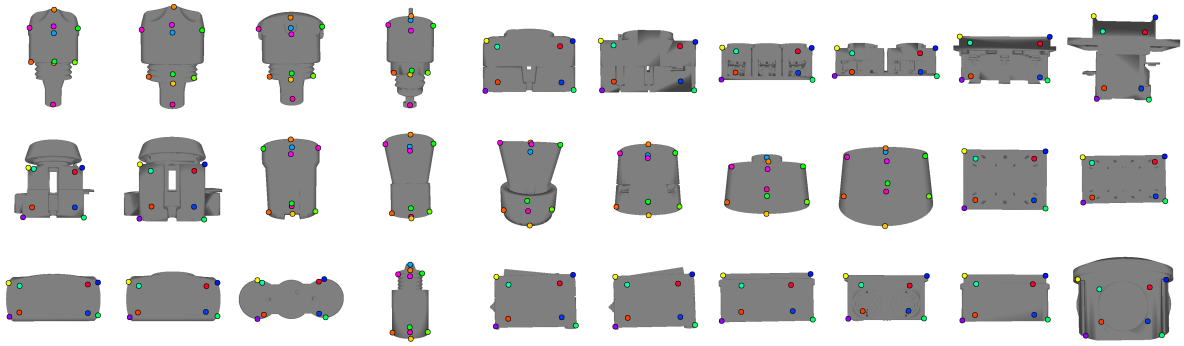

Figure 9. Our keypoint labels for the T-LESS dataset. Here, only shape class-specific keypoints were used due to the lack of texture on each object.

a global 3D estimate and detection in the current image into the current image plane with this camera pose, and count inliers with a χ^2 test using the detected keypoints and uncertainty. We take the camera pose hypothesis with the most inliers as the final ${}^{C}_{G}\mathbf{T}$, and reject any hypothesis that has too few. After this, any objects that have valid PnP poses but are not yet initialized in the scene are given an initial pose ${}^{G}_{O}\mathbf{T} = {}^{C}_{G}\mathbf{T}^{-1}{}^{C}_{O}\mathbf{T}_{pnp}$.

Since each object is initialized with a PnP pose, it is possible that the initialization can be very poor from a PnP failure, and, if the pose is bad enough (e.g., off by a large orientation error), optimization can not fix it due to only reaching local minima. To address this issue, we check if the PnP pose from the current image yields more inliers over the last few views than the current estimated pose, and, if this is true, we re-initialize the object with the new pose. After this, we perform a quick local refinement of the camera pose by fixing the object poses and optimizing just the current camera to better register it into the scene.

D. Keypoint Labeling

Choice of keypoints. The choice of keypoints for the network to learn is important, but there is no general consensus about which choice is best. Some have proposed to detect the corners of the 3D bounding boxes [28], while others chose keypoints that lie on the object [25,26] – which seems to be the more accurate approach [26]. Inspired by [34], we try to pick keypoints that carry some semantic meaning. Our keypoint labels on the YCB-Video dataset can be seen in Fig. 8, and Fig. 9 for the T-LESS dataset. Specifically, we split the objects into three categories based on the overall shape – box-like, cylinder-like, and hand tool – and choose a unified set of keypoints for each of these shape classes based on the most identifiable features. We

found that picking a set of keypoints for each one of these classes can accurately describe the shape of the objects for the YCB-Video and T-LESS datasets, and the keypoint network had a relatively easy time learning the keypoints despite the fact that the shapes of some objects are not exactly rectangular, cylindrical, etc. In order to increase the number of keypoints and their potential usefulness in a downstream application, we also add some instance-specific keypoints, such as brand names, bar codes, and hand grips, which only show up in the YCB-Video dataset. Such keypoints can still be shared among multiple instances of objects in the YCB-Video dataset, but sometimes occur between shape classes (e.g., bar codes show up on the box-like cracker box and also the cylindrical soup can).

Labeling tool. To label the keypoints, we create a simple labeling program which allows the user to pick the same keypoint (say keypoint k) multiple times on the CAD model, and takes the average 3D location in the CAD model frame as the final 3D keypoint location ${}^{O}\boldsymbol{p}_k$. The tool also allows the user to pick the canonical view $\{O_c\}$ used in Eq. 6 by simply rotating the object into the correct view. This is especially important in the YCB-Video dataset, where the object models are not already rotated into a canonical view as they are for T-LESS. The labeling program will be included along with our keypoint labels in the software release, which will be made available upon publication of this work. Detailed instructions for how to reproduce our keypoint labels will also be included in this release (i.e., the rules we used to determine where each keypoint goes), which can also be used to label keypoints on other datasets with objects similar to YCB-Video and T-LESS. We found that, after the user is acquainted with the labeling program, it only takes a few minutes per object to label the keypoints. In the future, we would like to reduce the labeling task for the shape class-specific keypoints, since there should be a simple set of heuristics to automatically label these when given the CAD model in a canonical view.

E. Extended Results

YCB-Video per-object results. As mentioned in Sec. 4.2, we provide more detailed results for each object on the YCB-Video dataset. The results are presented in Table 3. Here our method displays superior AUC of ADD and ADD-S for the majority of the objects. For the five symmetric objects, which are highlighted in bold blue in Table 3, our method has the best AUC of ADD-S for four of them – which shows our ability to handle these symmetric objects effectively. Note that the ADD metric is not very important for symmetric, since it checks for the match to the actual ground truth pose – which is arbitrary due to the symmetry – while the ADD-S simply checks if the shape of the object matches well between the ground truth and estimated poses [35]. This is clear especially for the case of the wood block, where our method actually scores a 0.0 AUC of ADD, while beating all other methods in the AUC of ADD-S metric. This is because our estimated pose for this object correctly aligned the CAD model to the scene to match the shape, but with a symmetry transform that yielded a completely different orientation from the ground truth.

Qualitative results. More qualitative results are shown in Fig. 10. Here we show three success cases and one failure case for both the YCB-Video and T-LESS datasets. Our system is able to estimate correct poses for a wide variety of difficult objects even in the presence of occlusion and bad or missing detections. A common failure case that we saw is the system initializing objects (especially symmetric ones) upside down. While we showed the only such case we found in the YCB-Video dataset, this is especially common in the T-LESS dataset where it is harder to distinguish the top from the bottom for many objects. Reliably solving such edge cases is an interesting question to answer in future research.

Table 3. Detailed results on the YCB-Video dataset. Bold blue objects are symmetric.

	PoseCNN [35]		DeepIM [15]		PoseRBPF [4]		MHPE [5]		Ours	
Objects	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S
002_master_chef_can	50.9	84.0	71.2	93.1	63.3	87.5	67.9	**93.8**	**75.0**	87.8
003_cracker_box	51.7	76.9	83.6	**91.0**	77.8	87.6	67.8	82.9	**84.0**	90.6
004_sugar_box	68.6	84.3	**94.1**	**96.2**	79.6	89.4	83.1	91.3	86.4	91.5
005_tomato_soup_can	66.0	80.9	**86.1**	92.4	73.0	83.6	79.5	92.2	85.3	**93.5**
006_mustard_bottle	79.9	90.2	91.5	95.1	84.7	92.0	81.6	90.8	**94.2**	**96.2**
007_tuna_fish_can	70.4	87.9	**87.7**	**96.1**	64.2	82.7	78.0	92.5	84.3	92.7
008_pudding_box	62.9	79.0	82.7	90.7	64.5	77.2	45.4	71.5	**84.1**	**92.4**
009_gelatin_box	75.2	87.1	91.9	94.3	83.0	90.8	76.1	87.8	**94.0**	**95.9**
010_potted_meat_can	59.6	78.5	76.2	86.4	51.8	66.9	69.1	85.5	**83.7**	**91.7**
011_banana	72.3	85.9	81.2	91.3	18.4	66.9	**87.7**	93.7	87.3	**94.3**
019_pitcher_base	52.5	76.8	**90.1**	**94.6**	63.7	82.1	76.8	88.8	89.4	93.9
021_bleach_cleanser	50.5	71.9	**81.2**	**90.3**	60.5	74.2	47.7	70.3	61.7	70.5
024_bowl	6.5	69.7	8.6	81.4	28.4	**85.6**	**40.2**	80.1	32.8	76.9
025_mug	57.7	78.0	81.4	91.3	77.9	89.0	40.6	72.8	**84.8**	**92.6**
035_power_drill	55.1	72.8	**85.5**	**92.3**	71.8	84.3	39.5	71.2	**85.5**	92.2
036_wood_block	31.8	65.8	60.0	81.9	2.3	31.4	**64.6**	85.5	0.0	**86.3**
037_scissors	35.8	56.2	60.9	75.4	38.7	59.1	64.5	88.9	**79.2**	**91.2**
040_large_marker	58.0	71.4	75.6	86.2	67.1	76.4	81.1	90.6	**84.9**	**94.7**
051_large_clamp	25.0	49.9	48.4	74.3	38.3	59.3	**49.2**	70.7	47.2	**83.0**
052_extra_large_clamp	15.8	47.0	31.0	73.3	32.3	44.3	8.6	47.4	**86.3**	**94.1**
061_foam_brick	40.4	87.8	35.9	81.9	84.1	92.6	75.1	92.6	**87.4**	**93.8**
Mean	51.7	75.3	71.7	88.1	58.4	76.3	63.1	82.9	**76.1**	**90.3**

Figure 10. Supplementary qualitative results for the YCB-Video (left) and T-LESS (right) datasets. The top three rows show some successful pose estimates from our system while the bottom row shows a failure case. The failure in both cases is from initializing objects upside down. The bowl on the bottom left and the orange object on the bottom right is upside down while the are upside down.