APPENDIX

A Experiment Details

A.1 Shape Network Training

We generated 3 distinct datasets of voxelized objects with size 64^3 from random axis-aligned boxes (AAB), YCB objects [38], and ShapeNet mugs [39]. Boxes for AAB had width, depth, and height uniformly sampled with 2 to 41 voxels. For YCB and ShapeNet we generated ground truth voxelgrids centered on the object with different rotations using binvox [42, 43]. For YCB we applied all 15 degree increment rotations about both the vertical and a horizontal axis. For Shapenet we applied all 5 degree increment rotations about the vertical axis.

During each epoch of training, each voxelgrid was augmented with translations sampled uniformly from -10 to 10 voxels in each direction. The 2.5D "known occupied" and "known free" voxelgrids were generated assuming a sensor looking down the x-direction. Sensor noise was simulated by sampling IDD 0-mean 2cm-std. deviation gaussian random noise in a depth image of 16x16, scaling that depth image to 64x64 using bilinear interpolation, then applying that noise to the x-direction of the known occupied and free voxelgrids.

We trained separate instances of PSSNet [5] on AAB, YCB, and Shapenet mugs, training for at least 100 epochs ($\sim$1 day), and used the iteration with minimal loss for experiments.

A.2 Robot Motion Generation

The following procedure was used to generate the robot motion which in turn generated the contact and freespace observations. The robot began each trial with a roadmap: a graph of nodes corresponding to configurations, and edges of robot motions connecting the nodes. Each scene contained a Goal Generator function, which mapped the completed objects to a goal Task-Space Region (TSR) [44]. Using 10 sampled worlds, there were a corresponding 10 separate TSRs. In the scene above, the Goal Generator took the mean of the completed object points, and generated a TSR centered 10cm back in the occluded region.

At each iteration if the robot did not currently satisfy any TSR, approximately 80 configurations were sampled from each TSR and added to the roadmap, and the robot would attempt to traverse the roadmap to the closest configuration in a TSR. If the robot satisfied all TSRs the task was considered complete.

If instead the robot satisfied at least one but not all TSRs, the robot took an information gathering action. For each outgoing edge of the robot's node on the roadmap, the Information Gain (IG) was calculated from the existing particles using the method in [45]. The robot took the action with highest information gain, which often (intentionally) contacted an object. The belief was updated and the next iteration began.

Detecting contact in simulation: The voxelized robot was computed using GpuVoxels [46] with a much larger 256^3 voxelgrid with 1cm voxel side lengths. This robot voxelgrid was converted to an occupied point cloud, then transformed to the object frame, and converted into a voxelgrid matching the size and position of the depth image voxelgrid. Contact was determined by checking for overlap between the robot and object voxelgrid. For each configuration visited not in contact, the voxelized robot was added to the known freespace. Each configuration in contact generated a Collision Hypothesis Set, added to $Q_{contact}$.

A.3 Baselines and All Results

Here we describe several baselines in more details. Figures 7 and 8 show all ablations and baselines for all scenes described in Section 5.

DIRECT EDIT: Perhaps the simplest method, in this baseline we apply the contact information directly to the voxelgrid. We sample latent vectors directly from the visual prior and decode into voxelgrid shapes as in other methods. At each measurement the known-free voxels from contact information are directly removed from predicted voxelgrids. In our problem formulation the true

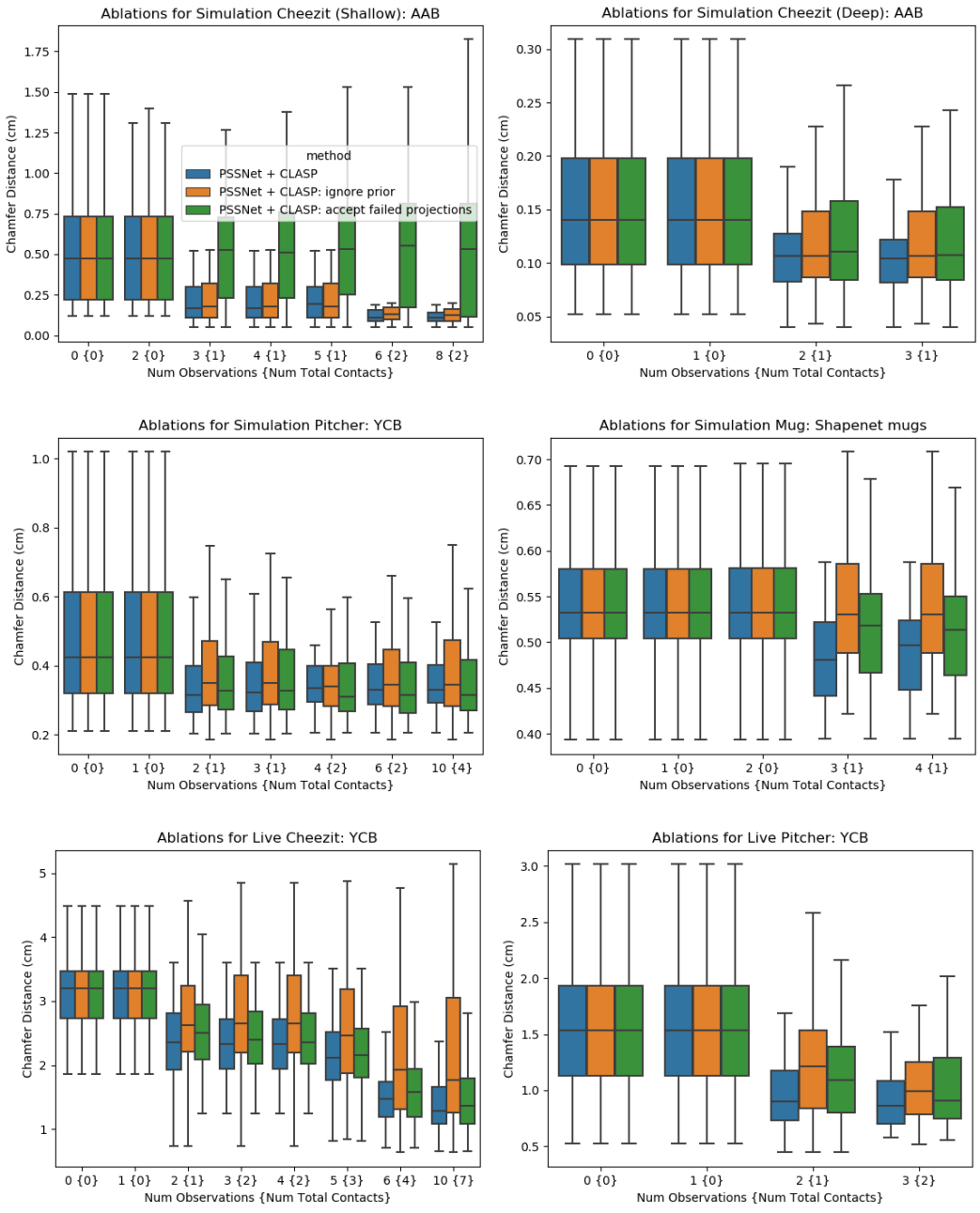

Figure 7: Ablation boxplots showing the Chamfer Distance from sampled particles to ground truth. The mean, middle quartiles (boxed colored region), and outer quartiles excluding outliers are shown.
.

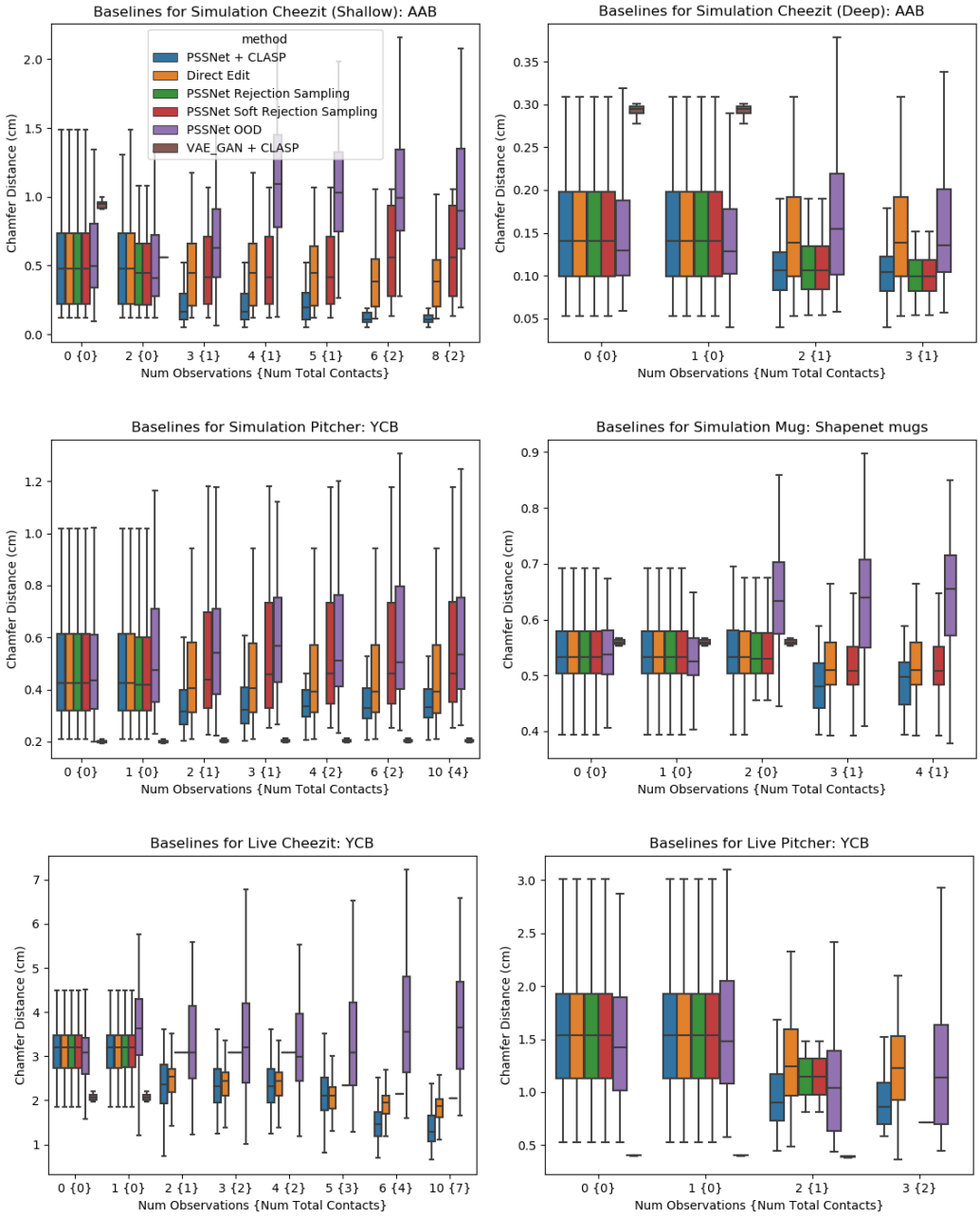

Figure 8: Baseline boxplots showing the Chamfer Distance from sampled particles to ground truth. The mean, middle quartiles (boxed colored region), and outer quartiles excluding outliers are shown. Rejection Sampling and VAE_GAN occasionally produced no valid shapes, in which case no box is displayed.

contact voxels are not known, however for this baseline we break that assumption and use the true object voxels that came in contact with the robot. These contacted voxels are added directly to the predicted shapes.

SOFT REJECTION SAMPLING: Rejection Sampling samples latent vectors from the distribution predicted by the encoder using purely visual data. To condition on the contact information, Rejection Sampling simply discards all samples that do not satisfy the contact constraints. To overcome the limitation that after 1 or two contacts no samples are accepted, reviewers suggested this softer implementation. In the SOFT REJECTION SAMPLING baseline we perform rejection sampling, saving rejected samples but marking them as invalid. If all samples are marked as invalid, we select all samples that violate the fewest number of constraints. This count includes each CHS without an occupied voxel, and each known-free voxel which the sample occupies. We then apply the DIRECT EDIT method on each selected sample to enforce that each sample satisfies all known constraints.

OOD (Direct Out-Of-Distribution Prediction): Our neural network accepts known-free and known-occupied voxels as input. During training the known-free and known-occupied solely came from vision, however contact information provides the same type of known-occupied and known-free information. This baseline takes the union of the known-free voxels from vision and robot motion as the known-free input, and uses the union of known-occupied voxels from vision and contact as the known-occupied input. In our problem formulation the true contact voxels are not known, however for this baseline (as in DIRECT EDIT) we break that assumption and use the true object voxels that came in contact with the robot. This baseline was suggested by many colleagues in early discussions of the paper. It is perhaps not surprising that this baseline performs poorly, as the combined information from vision and contact is out of distribution from all data on which the network was trained.

A.4 Likelihood Results

We consider an alternative analysis of the experiment data presented in Section 5.4. Given that we model scenes by sampling shapes in a particle filter, we consider the likelihood of the ground truth scene given the particles. Since a particle filter models discrete samples, none of which will exactly match the ground truth, we apply a kernel to our particles in workspace. Specifically, we apply a non-normalized kernel based on the Chamfer Distance between two shapes s_1, s_2:

$$k(s_1, s_2) = \frac{1}{CD(s_1, s_2)} \tag{6}$$

The (non-normalized) likelihood of a particular scene occupancy s under the belief of n particles Φ is then

$$p(s|\Phi) = \sum_{\phi \in \Phi} \frac{1}{n} k(f_{dec}(\phi), s) \tag{7}$$

where $f_{dec}(\phi)$ decodes all latent shape vectors $\psi \in \phi$ into a scene.

We plot the likelihood of the true scene in Fig. 9 and Fig. 10, and find similar trends as in Section 5.4. The magnitude of the likelihood is not meaningful, however the relatively likelihoods between the methods are. Initially methods perform similarly, except VAE_GAN which is either better or worse than other methods. With contact and freespace observations, our proposed CLASP with PSSNet tends to increase the likelihood of the ground truth scene, while VAE_GAN tends to decrease the likelihood.

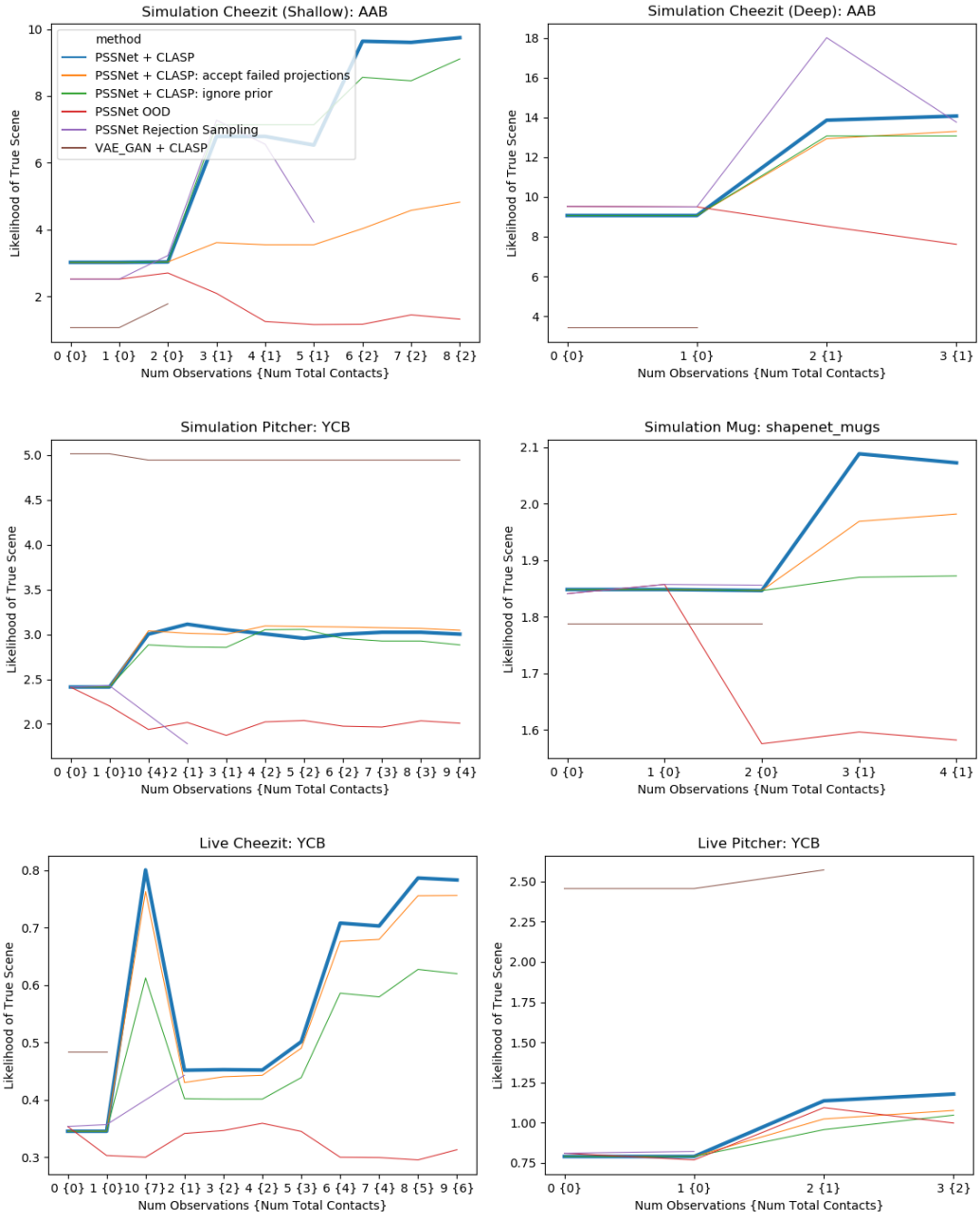

Figure 9: Plots of likelihoods of CLASP and baselines under the particle filter belief and kernel function.

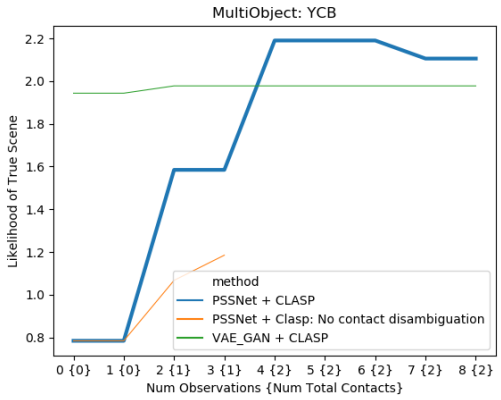

Figure 10: Likelihood of CLASP and baselines for the multiobject scene