

Figure 6: **Grasping Objects and Experimental Setup:** We split objects in each house into two categories: easy and hard. Hard objects are typically transparent, reflective, or translucent, making them hard to perceive with active depth sensors. Each house has 5 objects in each class. We also present the experimental setup in each house for these experiments. Each house has different lighting conditions, tables, and background objects. During grasping experiments, two objects or more are placed on the table, and the robot grasps and extracts the object located in front.

A Stereo Data Collection Rig

We now describe the two physical rigs used to collect stereo images for validation. In each rig, there is also an RGB-D sensor that is also used to collect data to evaluate the depth-based baselines.

A.1 Setup 1: Basler Stereo Pair

To collect data in the homes, we created a stereo pair with two Basler 60uc cameras. Our camera's native resolution is 2560×2048 pixels with a wide field of view fish-eye lens. However, we perform inference on a 4x down-sampled version. To obtain a stereo pair, all images are rectified to a pinhole camera model. The baseline of our stereo camera is 10cm , which was selected to match the average distance between human eyes.

To obtain depth data for our baseline technique, we mount a Microsoft Kinect Azure in wide field of view mode. The Kinect is mounted to a fixed steel base on top of stereo camera pair. We calibrate the Kinect to the left stereo camera using standard checkerboard calibration [14]. Camera calibration allows us to project the Kinect's point cloud into the left camera. We down sample the depth data using a 2×2 median filter to match the resolution inference is performed at. Example images from our data collection rig can be seen on Fig. 7.

A.2 Setup 2: HSR Stereo Pair

For the physical experiments run on the Toyota HSR and for the t-shirt experiments, we use the stereo pair from a mounted Zed2 camera, which has resolution 1920×1024 pixels. We down-sample images by a factor of 2, so the images used for training and prediction have resolution 960×512 pixels. The Zed2 camera is a time synchronized stereo pair camera with a baseline of 12cm . The depth and RGB-D experiments on the HSR are run using a mounted ASUS Xtion, which is placed 5cm below the Zed2.

B Implementation Details

B.1 Baseline Implementation Details

In this section, we will describe implementation details for the baselines used.

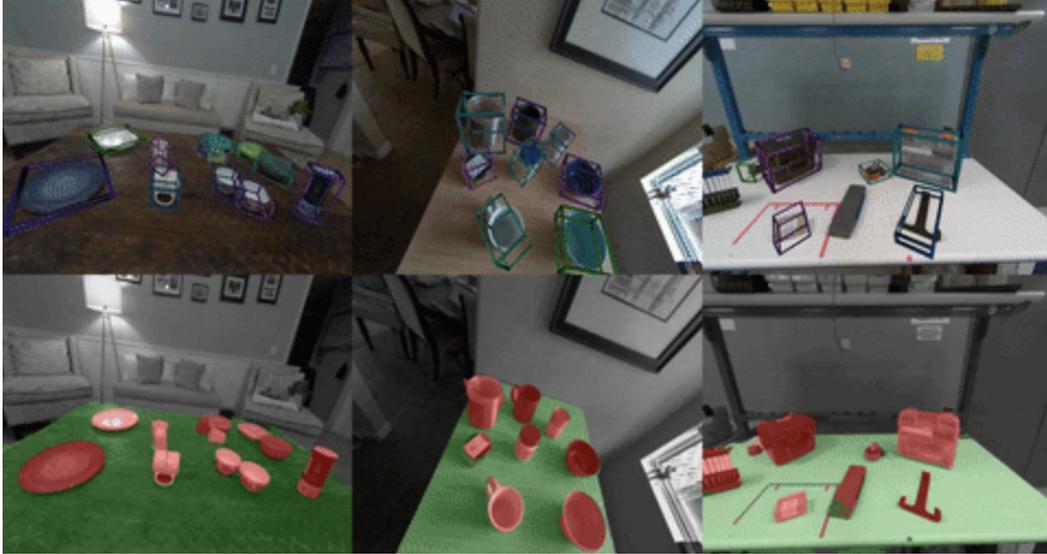
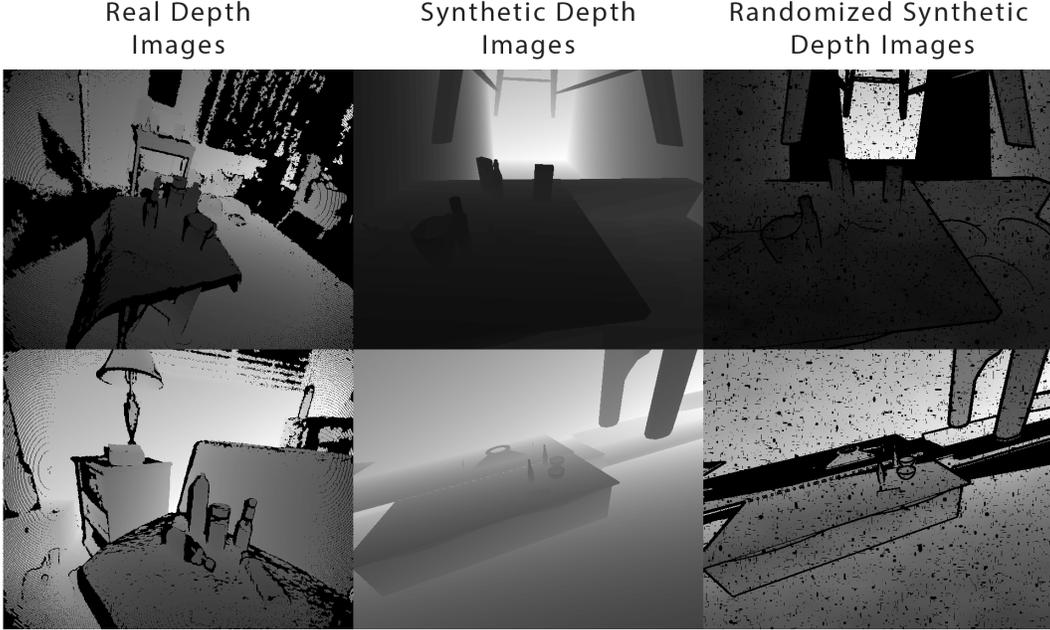


Figure 7: **SimNet Predictions on Indoor Scenes:** We present visualizations of SimNet’s predictions on the dataset used in the sensing ablation studies. This dataset consists of optically easy scenarios, but can have many objects in each scene. We find that the OBB and scene level segmentation predictions from SimNet on this dataset are high quality compared to baselines (Section C).

1. **Monocular (Mono):** This network trains only on the left image from the stereo pair. The left image is featurized as in the fully stereo network to form ϕ_1 and then fed into the backbone. This approach is similar to what is presented in [5].
2. **Monocular, Auxiliary Loss (Mono-Aux):** This network trains only on the left image from the stereo pair. In contrast to Mono, this network takes one of the channels of ϕ_1 and applies the disparity reconstruction auxiliary loss, described in Section 3.1. This forces the monocular network to reason about geometry in the scene.
3. **Depth:** This network only trains on depth images, and is identical to the Mono network, except for the number of input channels. Because depth images from real sensors contain many artifacts, we process each synthetic depth image by adding noise and random ellipse dropouts, as in Mahler et al. [1]. This process is illustrated in Figure 8. The approach is similar to [50].
4. **Depth, Auxiliary Loss (Depth-Aux):** This network only trains on depth images and is identical to the Mono-Aux network, except for the number of input channels (1). We use the auxiliary loss from Section 3.1 on the last channel of the predicted features to predict a low-resolution depth image (not disparity image), before feeding the feature volume to the ResNet50-FPN backbone. This forces the network to try to clean up artifacts in the real or noisy depth images.
5. **RGB-D:** This network separately featurizes the RGB and depth images using separate ResNet stems. It then concatenates computed feature channels and feeds them into the ResNet50-FPN backbone.
6. **RGB-D, Auxiliary Loss (RGB-D-Aux):** This network separately featurizes the RGB and depth images, concatenates the features and feeds them into the ResNet50-FPN backbone. However, on one channel of the depth features, we apply the depth reconstruction auxiliary loss function (Section 3.1).
7. **RGB-D, Stacked Input (RGB-D-Stack):** This network concatenates the RGB and D channels of the input into a single $H \times W \times 4$ array and feeds it into the network as a single input image.
8. **RGB-D, Sequential Input (RGB-D-Seq):** This baseline is heavily inspired by [4], and first trains a depth network as described above. The segmentation and disparity prediction outputs are then fed as input to a mono network in addition to the left image of the stereo pair. The idea with the network architecture is to use RGB information to simply refine the



Additional RGB-D fusion mechanisms can be seen here [52].

Figure 8: **Depth Image Noise Injection:** To simulate the artifacts present in real depth images (right column), we take synthetically generated depth images (middle column), and add process noise and random dropouts, particularly around edges, similar to Mahler et al. [1].

output of the depth prediction network. The depth network is frozen after is trained, and is not updated when the refinement network is trained.

B.2 Training Details

All networks are trained on a Tesla V100 GPU for 400,000 gradient steps with a batch size of 18. Models are trained using the Adam optimizer with learning rate $\alpha = 5e - 4$ and moment estimate decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The network is trained end-to-end using Batch Normalization [53] between each layer on the datasets described in Sec. 3.4. We trained the network from scratch with random weights.

For an input training stereo image (I_1, I_r) and corresponding labels, the network is trained by minimizing $\lambda_{seg} \ell_{seg} + \lambda_{kp} \ell_{kp} + \lambda_d \ell_d + \lambda_{d,small} \ell_{d,small} + \lambda_{cov} \ell_{cov} + \lambda_{inst} \ell_{inst} + \lambda_{vrtx} \ell_{vrtx} + \lambda_{cent} \ell_{cent}$. To tune the loss weights λ , we use a random grid search across 20 single gpu instances for 48 hours. We note that the multi-head training is only for reducing computation at inference time and that a single-headed network would not need this level of hyper-parameter tuning. On a TITANXp GPU, our network can predict room level segmentation, OBBs of unknown objects, keypoints and full resolution depth at 20 Hz.

B.3 Evaluation Criteria

In this section we will describe how mAP scores are computed for the 3D OBB and keypoint prediction tasks.

B.3.1 OBB Prediction

Similar to the 9DOF pose estimation [51], we use 3D intersection over union or (3D IOU) to measure box fit. However, since the boxes computed from OBBs can rotate freely around symmetric axes, we purposely only consider a low acceptance criteria of > 0.25 IOU. In practice, we notice there is significantly high correlation between this metric and grasp success during physical trials. The confidence value used during mAP calculation was the probability density of the predicted Gaussian peak around the object centroid. Due to the freedom of 3-D OBBs to rotate around symmetric axes, a looser acceptance criteria of mAP@0.25 was used when compared to the KITTI 2-D car detection task. The latter was evaluated on a more constrained 2-D bounding box prediction task and therefore a stricter evaluation of mAP@0.5 was used.



Figure 9: **Folded T-shirt Meshes:** Shown above is the fold sequence the robot is expected to perform during task execution. To simulate the distribution of shirt configurations encountered during folding, we source meshes of t-shirts created by artists and available for purchase online. We standardize the mesh dimensions in Blender, randomize their aspect ratios, and label vertices corresponding to keypoints.

B.3.2 Keypoint Prediction

To classify whether a ground-truth keypoint was successfully predicted, we check whether a predicted keypoint for the class is within 20 pixels of the ground-truth keypoint. If no such predicted keypoint exists, then we consider it a false negative. If a predicted keypoint is not within 20 pixels of a ground-truth keypoint of the same class, it is considered a false positive. We average over the individual class scores to obtain mAP scores, and each point on each class’s PR curve is computed by linearly varying the threshold for detection in the heatmap from 0 to 1.

B.4 Manipulation Details

In this section, we will describe how robot manipulation policies are constructed for the object grasping and t-shirt folding experiments.

B.4.1 Novel Object Grasping

Inspired by [26], given an OBB, the robot aligns the gripper with the largest principal axis. Thus, a bottle standing up would have a ”side” grasp where the robot grasps perpendicular to the face of the bottle. However, a stapler would have a ”top” grasp where the robot aligns the gripper with the orientation of the object with a vertical approach direction. If an object has no dominant principle axes, like a cube, the robot favors ”side” grasps, which require less motion for the kinematics of the HSR. We note this is not meant to be an optimal grasping strategy for all objects, but is only meant to handle a wide variety of common household objects shown during the grasping trials. The setup for these experiments can be seen in Fig. 3.

B.4.2 T-Shirt Folding

SimNet predicts the 2-D keypoints of a t-shirt, including the neck, sleeves and bottom corners. Using these, keypoints, a set of grasps and pull behaviors can be encoded that are relative to predicted keypoints. Our t-shirt fold sequence is known colloquially as the Sideways Column method. The fold sequence consists of the following steps: 1) pull the left-most sleeve to the right-most sleeve, 2) pull the left-most bottom to the right-most bottom, 3) pull the sleeves inwards to be aligned with the neck of the shirt, 4) pull the bottom of the t-shirt onto the top. An illustration of this fold sequence can be seen in Fig. 9.

In order to compute the 3D locations of these grasp points, we need to project the 2D keypoints onto the scene. We did this by leveraging the predicted disparity information and segmentation of the table. We first fit a 3D plane to the table using the disparity and segmentation output heads. We then project the 2D keypoints onto the plane, by computing the ray-to-plane intersection point. Given the 3D locations of the keypoint the robot can then infer grasp position and pull locations.

C Perception Ablation Studies

Different sensing modalities (monocular, depth, RGB-D, stereo) are ablated in this section. Descriptions of the implementation of each baseline are in Section B.1. Models are trained on the simulated small objects dataset generated with the Basler stereo pair camera model (Section A.1). We collect a dataset of real images using the Basler stereo pair and annotate them with 3D oriented bounding boxes. This dataset only consists of optically easy scenarios, non-transparent objects with

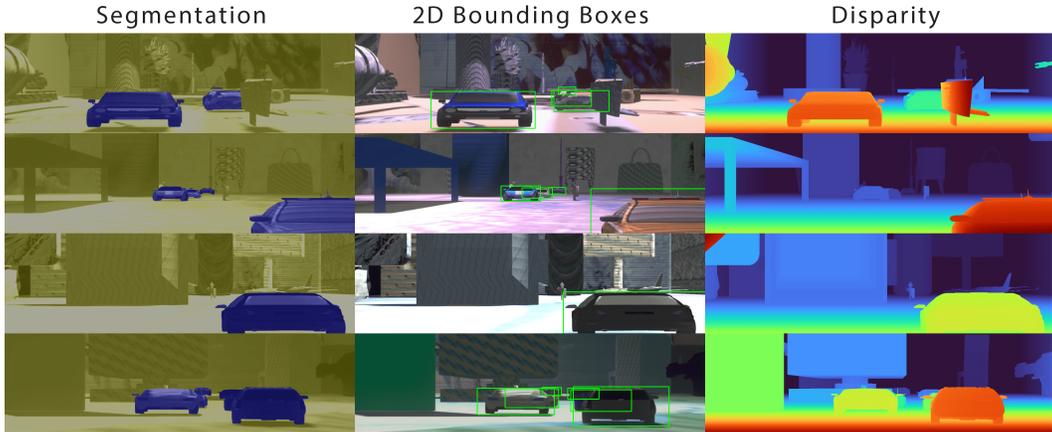


Figure 10: **SimNet Car Dataset:** We simulate cars and large objects in SimNet to train networks for the KITTI object detection benchmark. Each stereo image is labeled in simulation with segmentation masks, bounding boxes, and disparity. We use bounding boxes instead of oriented bounding boxes in this dataset, because the KITTI task requires 2D bounding boxes.

low amounts of natural lighting, and we therefore expect the depth-based networks to perform well. We hope that SimNet is able to roughly match or outperform as well on this task.

We observe that predicting coarse disparity for the monocular, depth, and RGB-D networks results in very little difference in performance. We also find that simply stacking the RGB and D channels performs roughly the same as feeding them into separate feature extractors. Pure monocular reasoning, however, performs quite poorly; this is likely due to the 3-D nature of the problem and is not reflective of sim-to-real transfer. SimNet has comparable performance to the baseline RGB-D transfer techniques, suggesting that sim-to-real performance on optically easy scenarios is similar between SimNet and the RGB-D methods.

Method	3D mAP
Mono [5]	0.164
Mono-Aux	0.169
Depth [50]	0.831
Depth-Aux	0.838
RGB-D	0.855
RGB-D-Aux	0.864
RGB-D-Stack	0.856
RGB-D-Seq [4]	0.774
SimNet	0.921

Table 5: **Small Object Ablation Study:** We perform an ablation study of different perception procedures and modalities on the small objects dataset. We evaluate each model on real, annotated images on the OBB prediction task. We find that monocular networks perform very poorly, while depth and RGB-D perform much better. However, SimNet consistently performs the well on this dataset of optically easy scenarios.

D Dataset Details

2D Car Detection Dataset: To predict 2D bounding boxes for cars on the road, we designed a dataset that enables our network to learn relevant geometric features for cars. Given car poses from held out KITTI [22, 54] scenes, we sampled cars and camera positions in natural configurations. Our car assets were taken from ShapeNet [55]. For distractor objects, we used random meshes from Shapenet and scaled them to be the size of buildings, buses, pedestrians and buses. We then placed them randomly in the scene to be collision free and resting on the ground plane. Images of our generated environments can be seen in Fig. 1. The car dataset generated in SimNet contains 50,000 stereo RGB images with annotations for 2D bounding boxes, segmentation masks, and disparity (Figure 10).

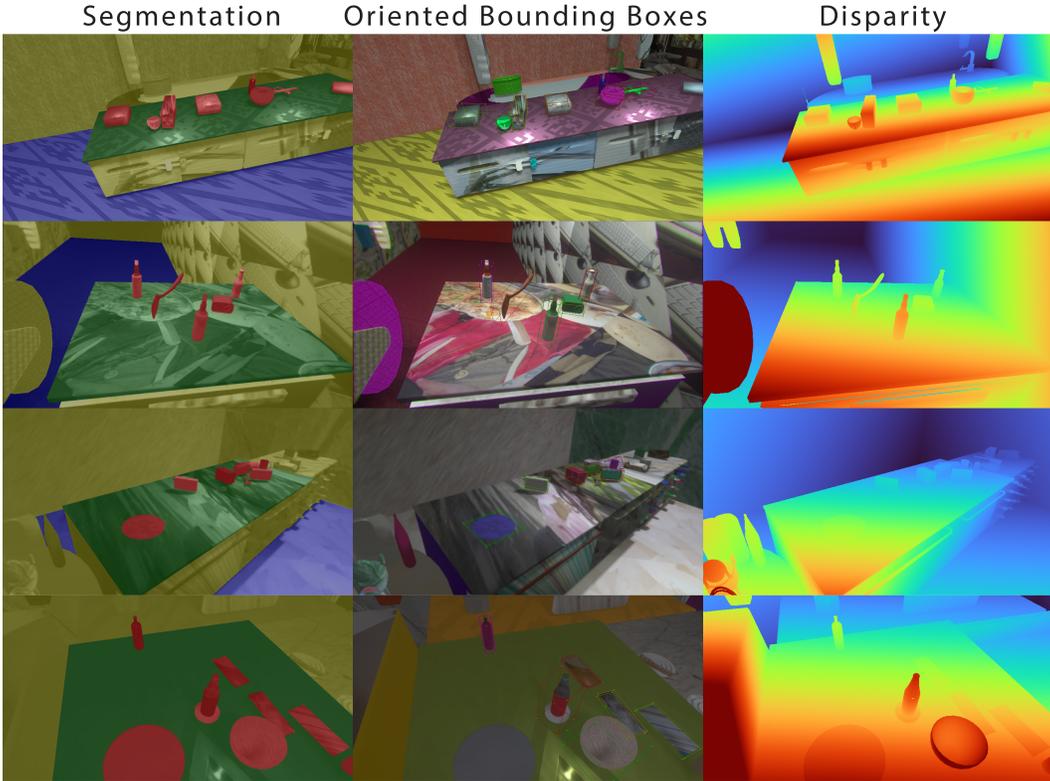


Figure 11: **SimNet Small Objects Dataset:** We simulate small objects placed on tabletops in SimNet. The dataset consists of stereo images with annotations for segmentation masks, oriented bounding boxes, and disparity.

Small Objects Dataset: For grasping objects on a table, we simulate indoor scenes by having a flat table with random objects from Shapenet [55] placed on top of it. The table is then surrounded with other random furniture from Shapenet that is randomly placed nearby the table. We then vary the size of the room and camera position relative to the table. Specifically, random positions are sampled from a half-sphere with a radius that is uniformly sampled between $[0.5, 2]$ meters.

This dataset consists of 50,000 stereo RGB images generated in SimNet. Each stereo image pair contains full annotations for full-resolution disparity images, OBBs, class and segmentation masks, as shown in Fig. 11. The same training set (and networks) is used for evaluation on both types of real objects (easy and hard). To train the RGB-D models, we make an equivalent dataset of simulated RGB-D images instead. We add structured noise to depth images to simulate artifacts commonly found in real images (Figure 8). For the real grasping experiments, we use the Zed 2 stereo pair for SimNet training and prediction and the HSR’s Asus Xtion for RGB-D. For perception ablation studies, we use the Basler stereo pair and Kinect camera models, which are described in Section A.1. This dataset is available here: <https://github.com/ToyotaResearchInstitute/simnet>.

T-shirt Dataset: We compile a dataset of 50 meshes of t-shirts and polos in various stages of folding that were purchased online (Figure 9). We randomize the scale and aspect ratios of each shirt mesh before placing it on a flat table in the scene with surrounding furniture as in the small objects dataset. Each scene has 1-3 t-shirts, and we generate keypoint, segmentation, OBB, and full-resolution disparity annotations in simulation. This dataset consists of 50,000 stereo RGB images. Each stereo image pair contains annotations for full-resolution disparity images projected onto the left image, segmentation masks, oriented bounding boxes, and keypoints.

The validation dataset is collected on the Toyota HSR using a mounted Zed 2 stereo pair and consists of 32 images. Each real image is manually annotated with keypoints. We consider shirt folding as a sequence of 4 folds (Figure 9), and we label each stage of folding with a different semantic segmentation class. This is not strictly necessary, but useful for failure detection when conducting

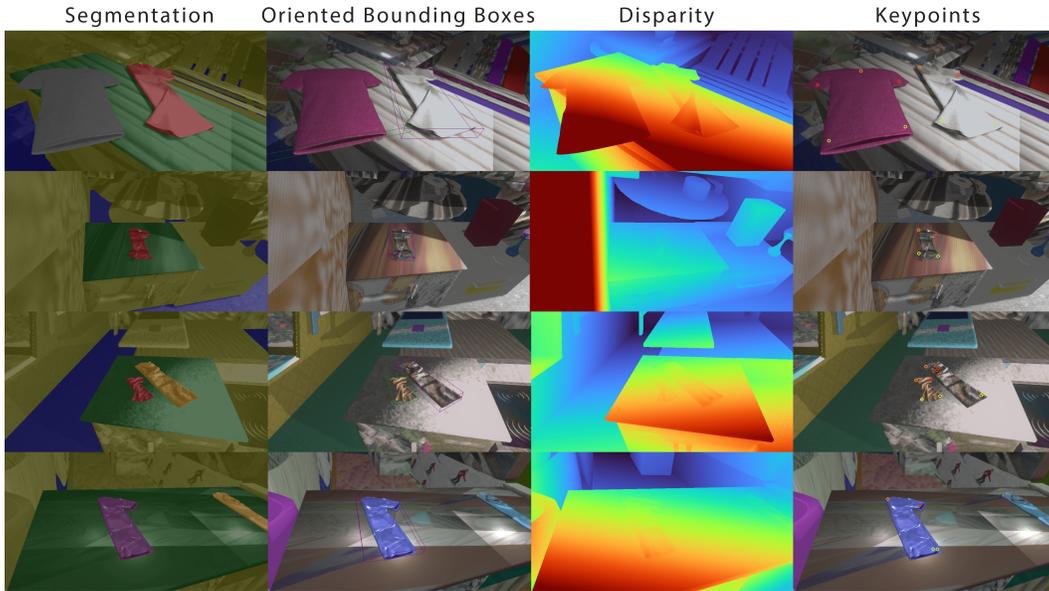


Figure 12: **SimNet T-Shirt Folding Dataset:** We simulate t-shirts in various stages of folding on flat surfaces in the center of the workspace. We procedurally annotate each stereo image with segmentation masks, oriented bounding boxes, depth images, and keypoints. The keypoints correspond to the sleeves, neck, and bottom corners of each shirt. In this dataset, we label each stage of folding with different segmentation classes. This is useful for grasp failure detection when folding with the robot.

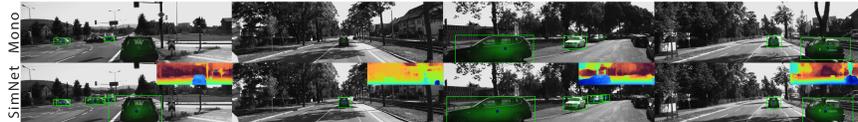


Figure 13: **KITTI bounding box predictions:** We evaluate SimNet on the 2D bounding box detection task in the KITTI benchmark. Top: Is the prediction of monocular-DR. Bottom: The predictions of SimNet with the estimated disparity from the SVCN in the top right corner. By performing approximate stereo matching SimNet achieves high-prediction accuracy via improved transfer from simulation.

robot experiments. If a grasp is missed, which is possible due to the large morphology of the gripper, this enables the planner to recognize that it is still in the same state. For the depth baselines, we use the Asus Xtion camera on the Toyota HSR. We present visualizations in Figure 12.