Cross-view Semantic Segmentation for Sensing Surroundings Supplemental Materials

Bowen Pan[1], Jiankai Sun[2], Alex Andonian[1], Bolei Zhou[2]
[1]Massachusetts Institute of Technology [2]The Chinese University of Hong Kong

Exp I: Generating the semantic floor maps

Our VPN can integrate the local top-down-view maps into a semantic floor map. Since the ray spread into the top-down-view camera is not in parallel, we can not splice them directly. Thus we crop the central area of the predicted top-down-view map so that we can approximately assume that the ray emitted from this area is in parallel. We integrate the top-down-view maps in a max-pooling manner according to the predicted confidence map. We first discrete the whole map into a 250×250 grid map. At each grid point, we predict the corresponding top-down-view map and the confidence map. We bilinearly upsample it to $H_p \times W_p$ to match the real-to-grid scale. W_p and H_p are given as follows,

$$W_p = W_I \times \frac{L_v}{W_m}, \quad H_p = H_I \times \frac{L_v}{H_m}, \tag{1}$$

where L_v is the actual scope of the field of local top-down-view patch, W_m and H_m are the actual width and height of the whole map, and W_I, H_I are the width and height of the final shown image. W_m and H_m can be computed by the scale factor of each scene. Then we crop the $24P \times 24P$ central area of the H_pW_p upsampled map and then paste it to the whole map in a max-pooling manner according to the confidence map. Some qualitative results are shown in Figure 1. The integrated semantic floor map shows the spatial layout of all the objects in the environment. , which provides valuable information for motion planning and obstacle avoidance in mobile robot.

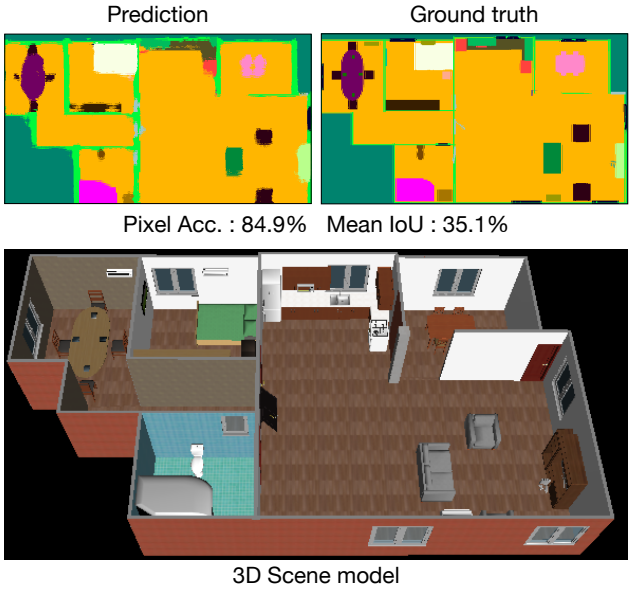

Figure 1: Integrating the local top-down-view semantic maps into a global semantic floor map for two scenes. The ground truth semantic floor map is generated from the ground truth top-down-view semantic maps. As a reference, the 3D scene model is also shown below.

Exp II: Exploration with top-down-view map

Humans exploring space will head to space which they have not visited. This intuition reflects that exploration requires the agent to identify free space as well as remember which areas it has not visited yet. To achieve this goal, we make the agent able to identify the free space by training it to predict the top-down-view free-space map. Along with the state map which records the already-executed action sequence, the agent can remember the unvisited area.

Top-down-view free-space map. We train the VPN to predict Top-down-view free-space map. Different from the semantic map, the free-space map has only two categories, obstacle, and free space, which are denoted by 0, 1 respectively.

State map. Due to the ideal assumption made above, by memorizing the previous actions it has executed, the agent can easily build the state map which contains the information of the already-visited positions. We label the unvisited pixels as 0 and the already-visited pixels as 1 on the state map.

Exploration algorithm. We detail the navigation policy decision algorithm in Algorithm 1. At each time step t, we make the decision a_t and update the agent with the next top-down-view free-space map T_{t+1} and state map S_{t+1}. In both the top-down-view free-space map and the state map, we assume that the agent is always at the center of the map.

Result and comparison

To demonstrate that VPN can help navigation, we compare it with the following baselines for exploration. **Random**

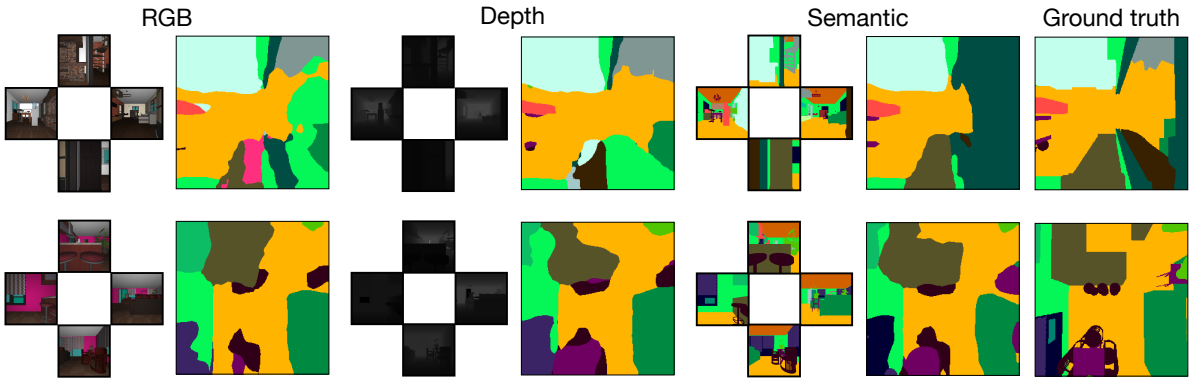

Figure 2: Cross-view segmentation on House3D using the input modality of RGB image, the depth map, and semantic mask respectively. Here the number of input views is fixed as 4. Ground truth is shown on the right.

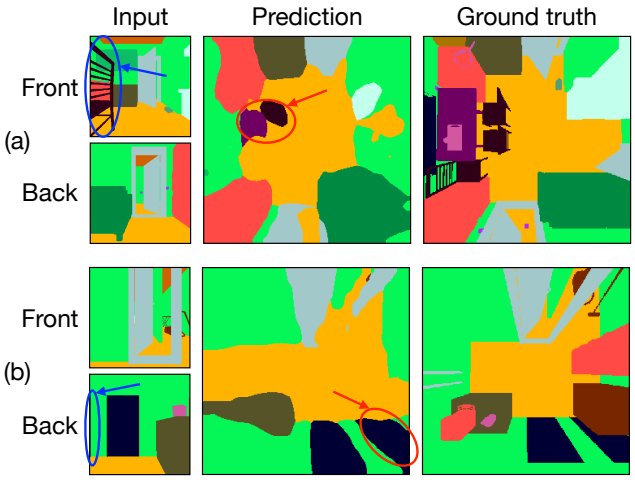

Figure 3: Predicting invisible objects: In (a), although the input semantic masks do not indicate that there is a table beside the chair, our model predicts it on the top-down-view mask based on the prior that tables and chairs usually appear together. In (b), in the Back view input mask, only a very small part of the second door can be observed, our model segments the door fully in the top-down-view prediction based on the shape prior to the door. Here the prediction is done by 2-view VPN with semantic input on the left.

walk: Random walk agent randomly chooses one action from Forward, Back, Right-forward and Left-forward, at each time step. **Top-down-view navigation with ground truth:** By planning on the ground truth top-down-view free-space map with Algorithm 1, we can obtain the upper-bound performance of our method. The difference is that in our case the top-down-view free-space map is predicted by VPN, rather than the ground truth. **Imitation learning without top-down-view:** A reactive CNN network learns to imitate the expert exploration trajectories given the first-view observations. The trajectories are generated by the baseline above with a top-down-view ground truth map. Network inputs are 4 first-view depth images. Here we choose depth map as input since it achieves better performance, 93.5% pixel accuracy on this binary classification task than the semantic mask, which yields 64.2%. We also input the state map to indicate the already-visited area. We extract 729 trajectories for the training set and 121 trajectories for the validation set to train the navigation agent. Each trajectory contains 150 states which are all labeled with expert policy.

Algorithm 1 Exploration policy decision

Input: A top-down-view free-space map T_t and a state map S_t at time step t, where $T_t, S_t \in \{0,1\}^{L\times L}$.

Output: Policy action a_t, where $a_t \in$ {Forward, Back, Left-forward, Right-forward, Done}.

$U_t \leftarrow T_t \bigcap \neg S_t$; $a_t \leftarrow Done$; $d_s \leftarrow +\infty$
$D_t \leftarrow computeDistMap(U_t)$
for a in $\{Forward, Back, Left\text{-}forward, Right\text{-}forward\}$ **do**
    $d = execute(a)$
    if $d_s > d$ **then**
        $d_s \leftarrow d$; $a_t \leftarrow a$

$computeDistMap()$: Compute the shortest distance of each map pixel to the unvisited free-space region.
$execute()$: Return the shortest distance of the pixel to which the agent transit if execute the action a.

Table 1: Comparison on exploration.

Method	Coverage Area
Random walk	260.3 ± 82.7
IL w/o top-down-view	443.8 ± 340.6
Top-down-view navigation	673.8 ± 349.8
Top-down-view navigation with GT	1070.8±326.2

We run the algorithm directly on our predicted top-down-view map. For testing all the methods, we start the episode by initializing the state maps from zero, indicating that all free space is yet to be visited. *Coverage Area* is defined to measure exploration performance. We randomly choose 100 starting points on a scene map. For each starting point, we

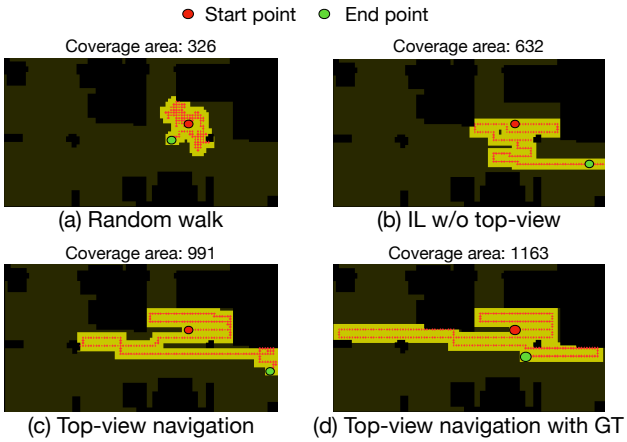

Figure 4: Examples of the surrounding exploration. Each method has the same start point. The trajectory is shown and the explored area is lighted up.

let the agent explore the space for 300 steps and compute the coverage area. Then final results are obtained by averaging the coverage area of these 100 episodes. Table 1 plots the exploration result for different methods and Figure 4 shows some sample trajectories. We can see that equipped with the predicted top-down-view map from our VPN, the agent can act almost like an expert.

More results of paper

In this section, we include some extended results which are mentioned in our original paper. We show the qualitative segmentation results of all modalities in Figure 2. And in Figure 3, we show some interesting cases where we find that our VPN can predict invisible objects. Finally in Figure 5, we demonstrate more adaptation results in the real world.

Visual Interpretation

In order to better understand what the trained model has learned during the task of cross-view semantic segmentation, we apply gradient-based class activation mapping (Grad-CAM) [1] to identify relationships between the generated top-down semantic mask and the input views. In particular, given a particular (x, y) coordinate in the generated semantic mask and the predicted class at that location, we identify the spatial regions across all of the input views that are most relevant to the prediction at that location. Figure 6 and Figure 7 show the highlighted regions across four views when one location is clicked on the predicted top-down-view mask.

Here we use the 4-view model with input modality as RGB images. We visualize the input observation as RGB images. For each scene, we select 4 points on different objects in the scene to interpret. We can see from these examples that our model produces very reasonable heat map results, which shows that VPN can truly understand the spatial configuration of the input spatial scenes.

References

[1] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, Dhruv Batra, et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proc. ICCV*, 2017. 3

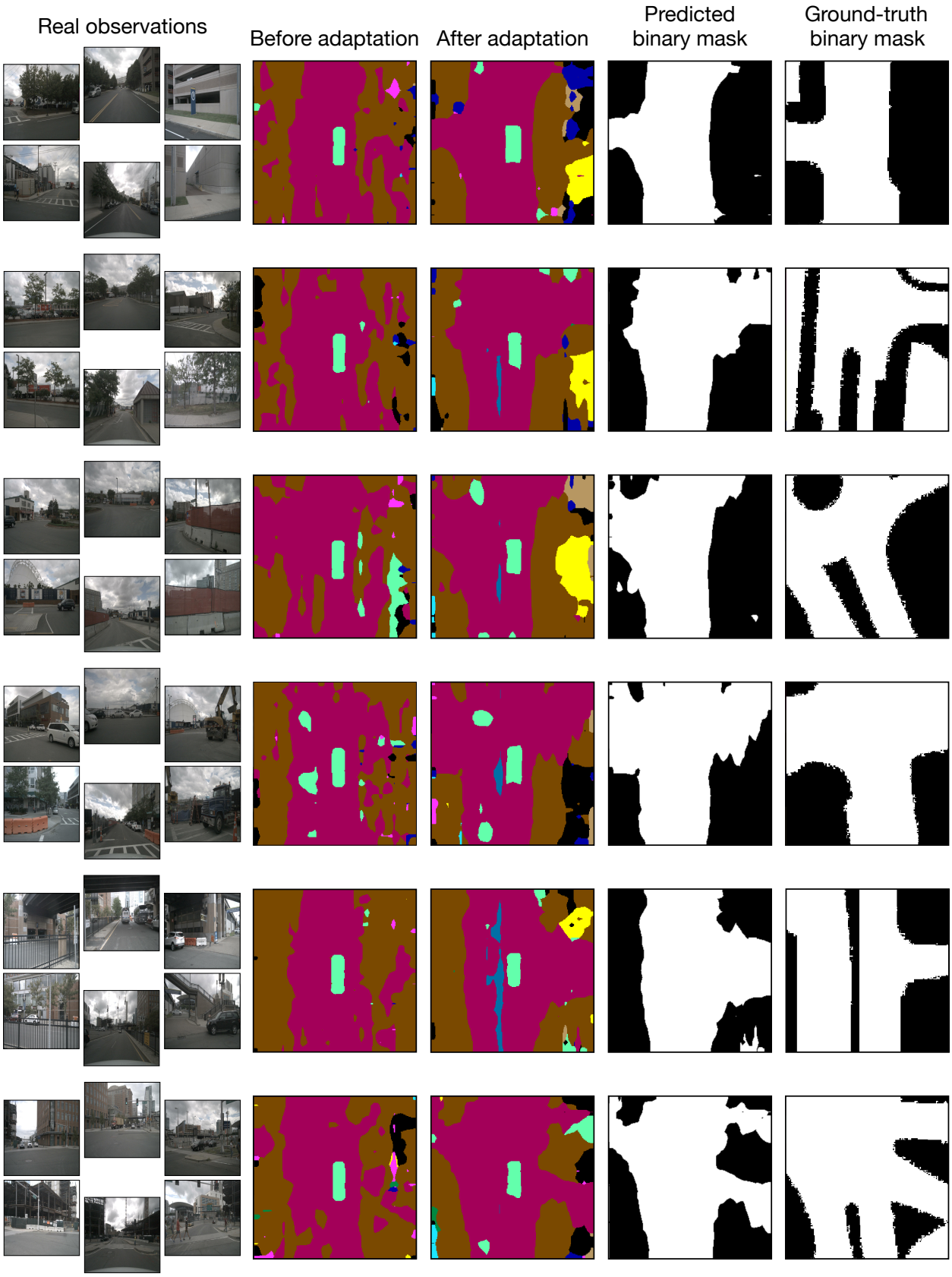

Figure 5: More qualitative results about sim-to-real adaptation.

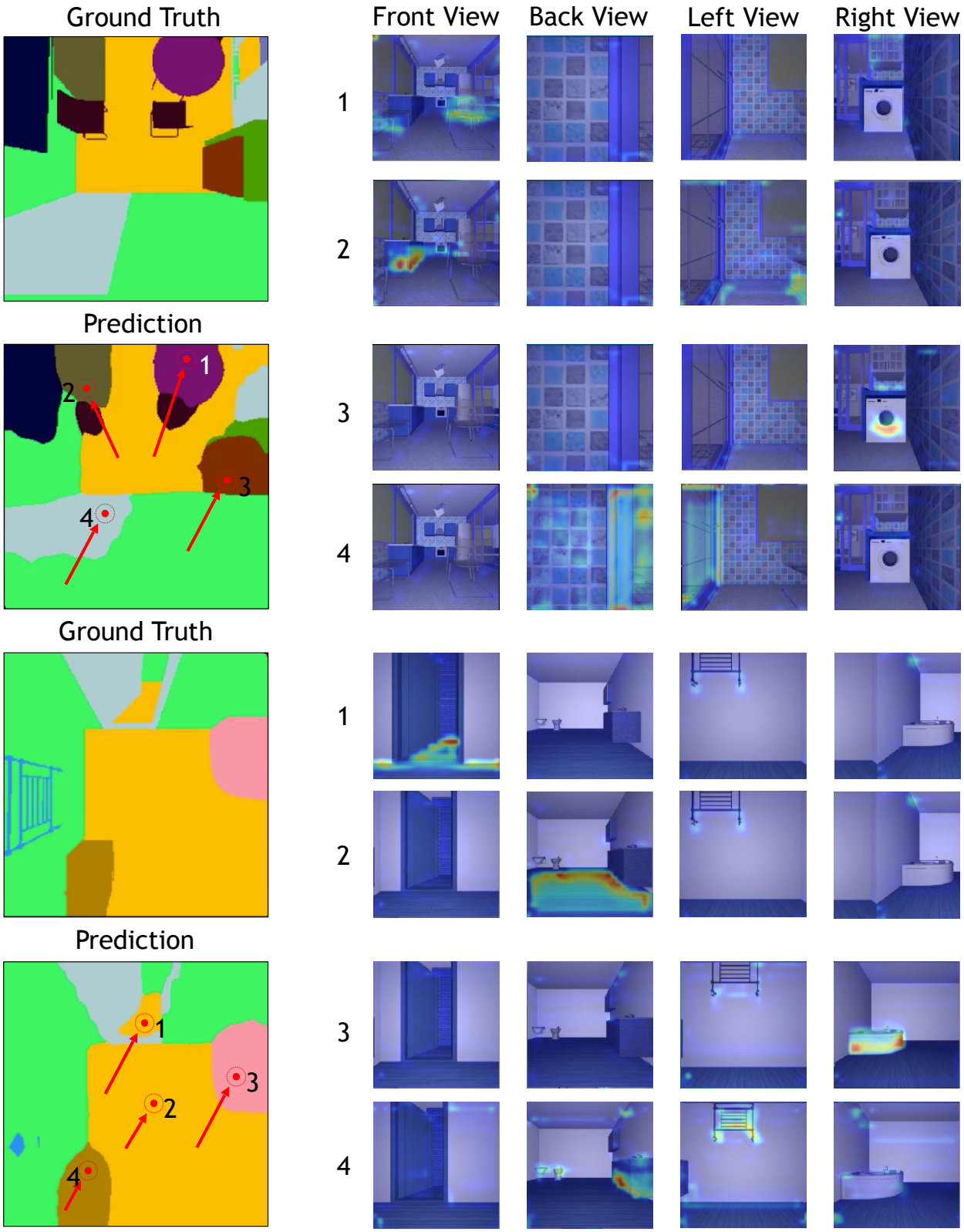

Figure 6: Visualizing the spatial regions of different views that contribute most to the prediction. In each example, for each of the four pointed coordinates on the predicted mask, we highlight the spatial regions that are most relevant to the prediction.

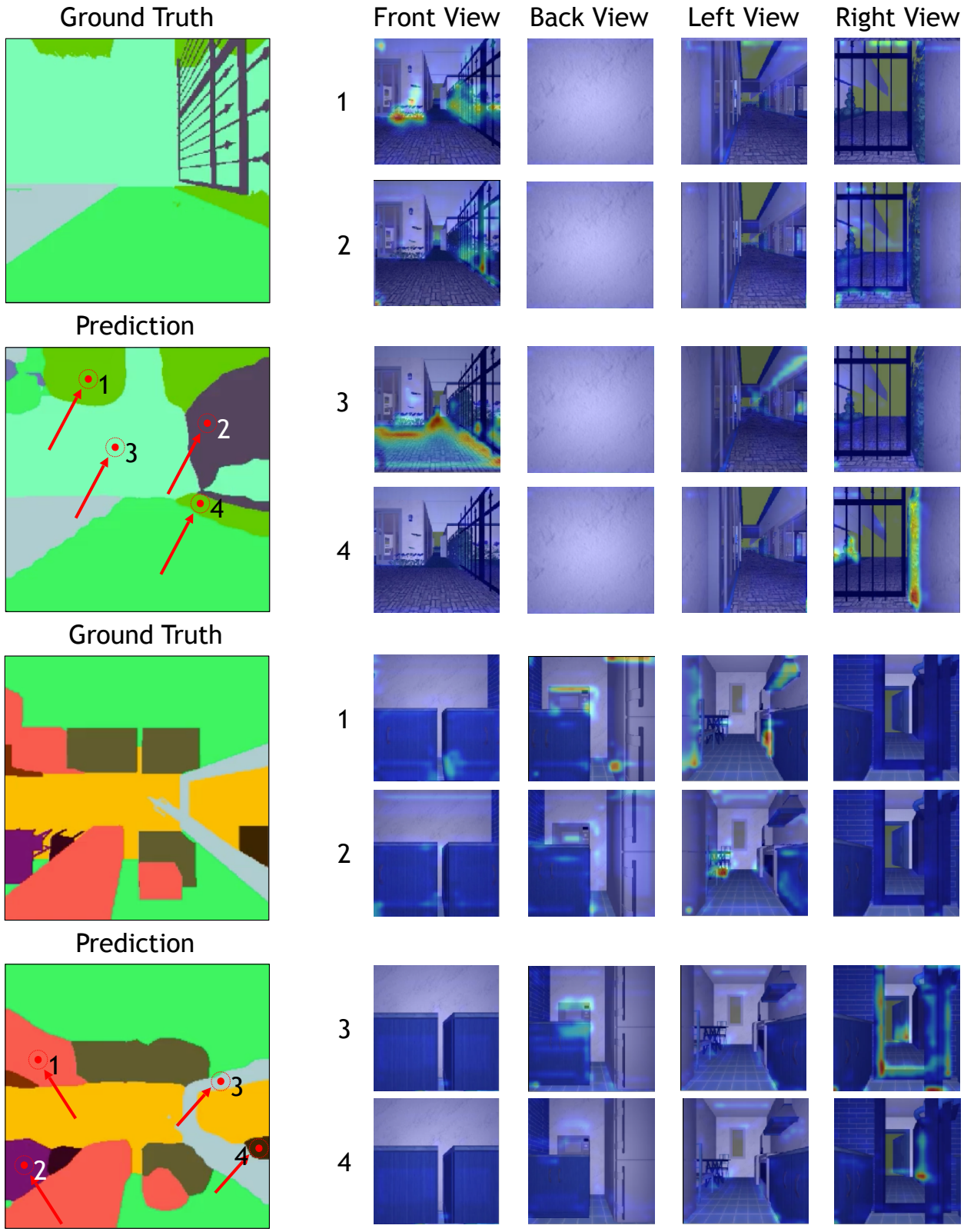

Figure 7: Visualizing the spatial regions of different views that contribute most to the prediction. In each example, for each of the four pointed coordinates on the predicted mask, we highlight the spatial regions that are most relevant to the prediction.