Walls reconstruction from RGB-D images using semantic segmentation deep neural network

Name: Giovanni Bagolin
Id: VR445681
Course: Deep learning

1 Introduction

Walls reconstruction is an interesting topic in computer vision. It has a variety of uses, e.g it allows an AI Embodied agent to navigate in the environment.

Walls reconstruction can be difficult if the scene is complex.

We tested 19 semantic segmentation networks on a small part of the houses dataset, to determine which network could extract better semantic information from houses scene images. Then, we used two semantic segmentation networks, to extract scenes semantic information on the whole dataset. For each house, we did a point cloud reconstruction, using the semantic information extracted before, and RBG-D scene images, to then extract walls from this reconstruction. An evaulation of the two reconstruction with respect to the ground truth reconstruction has been done. Thus we determined which was the best. A further comparison has been done, by using a semantic information extractor integrated in habitat simulator [1].

2 Method

2.1 Data extraction

In order to extract RGB-D images from the houses dataset provided, Habitat [1] has been used. Habitat is a simulation environment for Embodied AI, which allows training agents to perform tasks such as: navigating in the environment, accomplish orders, answer questions etc.

Habitat consists of Habitat Sim and Habitat Lab. Habitat sim, is a 3D simulation environment, with configurable agents and multiple sensors. Habitat sim has been used as an extractor of images. The agent, after navigating to a random point in an area of $1m^2$ takes 12 picutures, one for each angle in the list: [0°,30°,60°,90°,120°,150°,180°,210°,240°,270°,300°,330°], rotating around its y-axis.

In this process, semantic informations have also been estracted, using the semantic segmentation extractor integrated in habitat. The list of classes that habitat uses is available at appendix A. The dataset is composed of 11 houses, each of different size.

2.2 Open MMLab

In literature there are various deep neural networks whose purpose is to segment images extracting semantic information. See [2], [3], [4] as examples. The training of those DNN requires a huge amount of data, and resources. Furthemore, the time needed is long. Therefore, the solution adopted in this project was to use pretrained DNN. Nowadays many semantic segmentation DNN are available in the internet. However, also the frameworks available are many, and therefore, all of sudden, the task of verifying the goodness of each DNN cited and others, became hard. Luckily, OpenMMLab Semantic Segmentation Toolbox and Benchmarks [5], did the hard job for us.

OpenMMLab Semantic Segmentation, is an open source toolbox, which mantains, a number of semantic segmentation deep neural networks. All the networks have been trained and tested on

different datasets: such as CityScapes [6], ADE20K, PASCAL VOC2012 [7], just to cite some. Furthermore, different checkpoints are available for each dataset too. This allows us to select the best network between all, by evaluating the network on our dataset.

Given the huge semantic file size of each image of each house, not all the neural networks have been tested on the complete dataset. Infact, only few images, have been evaluated to check the goodness of the networks. Particularly, six images, have been selected, and tested on the networks. OpenMMLab Semantic Segmentation, provides in total 19 networks pretrained on ADE20K dataset. However, there are many checkpoints for each network, and config files.

Therefore, before evaulating the small part of the dataset selected, a CSV has been created for each network. Each CSV contains the checkpoint link, the checkpoint file, and the path to the config file in OpenMMLab Semantic Segmentation config directory. This came to be usefull for our purpose of evaulting 6 images on 19 networks with different checkpoints and configurations. In total, for each image to be evaluated there are 72 images segmented, one for each network, with a specific checkpoint, and a specific configuration. The code for creating the CSV for each network is available here: [8]. After having evaluted 6 images for each network,checkpoint and configuration file, there are a total of 432 images. Once they all have been compared with the original images, it came up that DANet [2] was the network which segmented better than the others. Thus, DANet , has been used to segment the whole dataset.

2.3 Dual Attention Network

Dual Attention Network integrates local features, with their global dependencies, resulting in an improvement of feature representation, which helps in giving a more precise result in segmentation. Specifically DANet, improves the mIoU score on many different datasets such as Cityscapes [6], Pascal [7], and COCO Stuff Dataset [9], conferming to be on of the state of art segmentation network. The figure 1 represent the danet architecture.

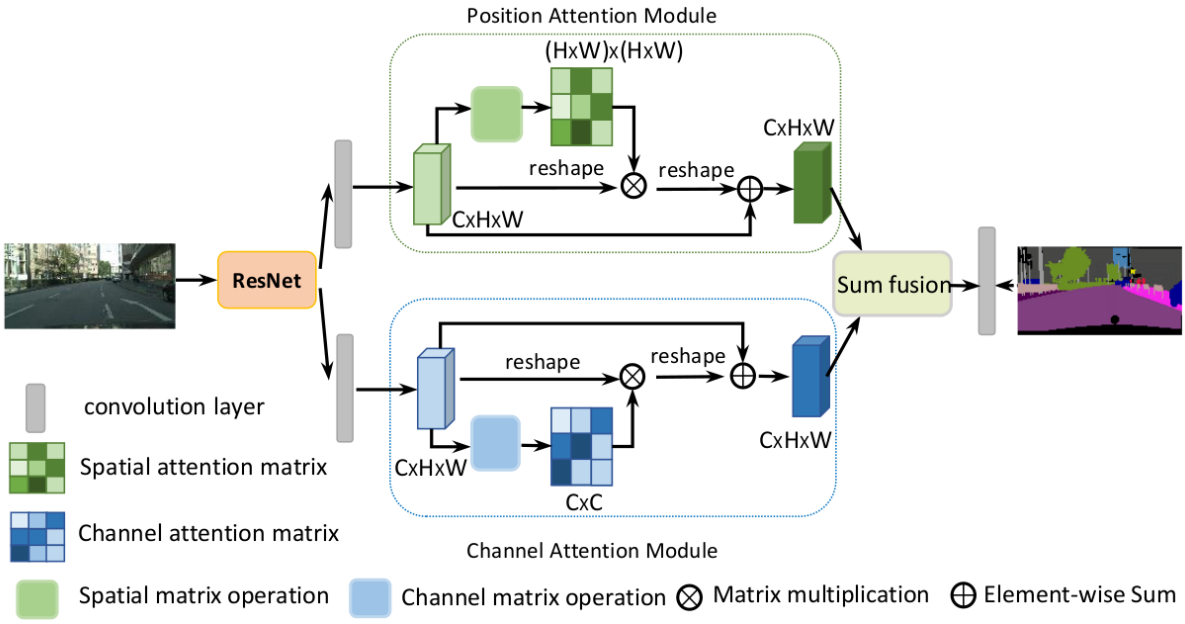

Figure 1: DANet architecture. Source: [2]

The main parts of the network are the two attention modules. The reason of the creation of these two modules, have to be found in real world objects. Infact, two objects having same label, e.g 'door', might be extremely different in size, lights, colors, and views. These differences in objects images, produce intra-class incosistency, which, affects the accuracy of the model. This issue has been addressed by building associations among features with the attentions module represented in figure 1. Specifically, they are needed to aggregate long range contextual information. Furthermore, attention modules are necessary to draw global context over local features, thus, obtaining a better feature representation for pixel level prediction. Note that a pretrained ResNet [10], has been employed as backbone.

2.3.1 Position attention module

The main purpose of the position attention module is to encode a wide range of contextual information into local features, resulting in a more precise representation. The module input is

a feature $\mathbf{A} \in \mathbb{R}^{CXHXW}$ and the module output is a feature $\mathbf{E} \in \mathbb{R}^{CXHXW}$. Not going into detail of what transformations are done, it can be shown that each element of $\mathbf{E}$ is a weighted sum of the features across all positions and original features.

It has a global contextual information and when needed, aggregates context according to the spatial attention map $\mathbf{S}$, where each element $s_{j,i}$ measures the i^{th} position's impact on the j^{th} position.

2.3.2 Channel attention module

The channel attention module has been created to model interdependecies between channels. Input is a feature $\mathbf{A} \in \mathbb{R}^{CXHXW}$ and output is a feature $\mathbf{E} \in \mathbb{R}^{CXHXW}$. It can be shown that $\mathbf{E}$ is a weighted sum of the feautures of all channels and original feature. It models semantic dependencies between feature maps, helping in improving feature discriminability.

2.4 Extract semantic information using DANet

All the houses have been segmented using pretrained DANet. The code for this part is available here: [11]. The specific DANet configuration uses ResNet101 as backbone, and has been trained with 512x512 size images on ADE20K dataset, for 160k epoches. Since the model's output is an image of size 480x640x3, a further transformation is needed to get the semantic matrix of dimension 480x640. The semantic matrix represents the class for each pixel. We are particularly interested in two classes: 'wall', and 'floor'. Each class in the model has a specific color. The wall class has been assigned the RGB color [120,120,120], while the floor class has been assigned the color [80,50,50]. The semantic matrix has been created considering three classes: wall, floor and undefined. Given the segmented image $\mathbf{S}$, each element of the semantic matrix $e_{i,j}$ is assigned following the rule:

$$e_{i,j} = \begin{cases} 0 & if\ s_{i,j} = [120, 120, 120], \\ 1 & if\ s_{i,j} = [80, 50, 50], \\ 2 & otherwise \end{cases}$$

Where $s_{i,j}$ is the element in row i and column j of the segmented image. It should be clear that classes wall, floor, and undefined have respectively labels 0,1,2. Figure 2, 3, 4, shows an image segmented by DANet and the corresponding semantic image.

Figure 2: Input image

Figure 3: DANet output

Figure 4: Semantic image

2.5 Wall reconstruction

After the semantic information for each house have been extracted we moved on to the next task which was wall extraction. To extract wall and floor information, this procedure has been followed. First, a point cloud reconstruction of the house has been done, setting to zeros all the depths that are not wall, or floor. An example of the point reconstruction is available at figure 32. Specifically, we set to zeros all the depth elements which in the semantic matrix are not 0 or 1. Doing so, we are able to extract correctly what has been segmented by DANet as floor and wall. This procedure has been repetead for each scene, thus, for each image in the dataset. Note that, point cloud reconstruction could require huge amount of memory, due to the integration of all the scenes in the environment. The amount of memory needed, depends also on the voxel length wanted. The smaller the voxel length, the higher the memory required is. After the point

cloud reconstruction has been completed correctly, we moved on to the wall reconstruction. We considered to be wall, everything which has a y-coordinate greater than the floor level plus an epsilon. Doing so, we are able to extract all the wall points in the x,z coordinates.

The code for this part is available here: [12].

2.6 PSPNet

Pyramid Scene Parsing Network [4] is the second network considered to extract semantic information on the whole dataset. PSPNet increas the IoU score, on many datasets [6], [9], using a pyramid pooling module as a global context prior. The network architecture starts with a pre-trained ResNet model with dilated network strategy [10], to extract the feature map, which has 1/8 the size of the input image. The feature map pass trough the the pooling pyramid, which extracts global context information. Finally it is concatenated with the inital feature map. A convolutional layer is added at the end, to generate the final prediction map. Figure 5 show the architecture of the Pyarmid Scene Parsing Network.

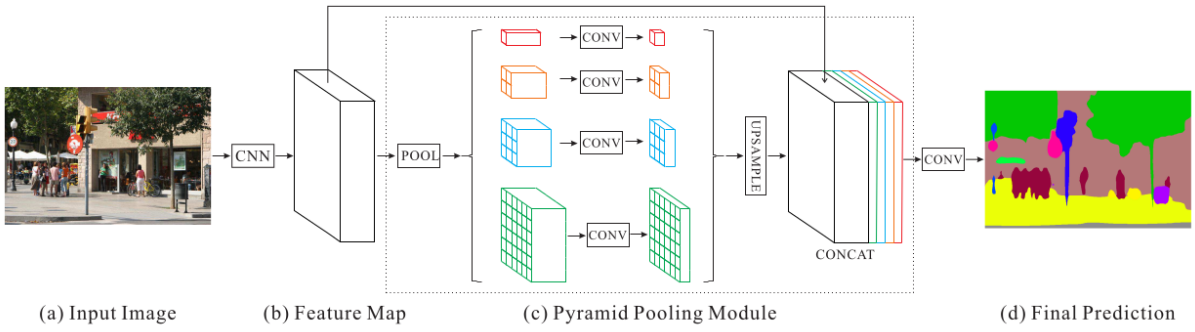

Figure 5: PSPNet architecture. Source: [4]

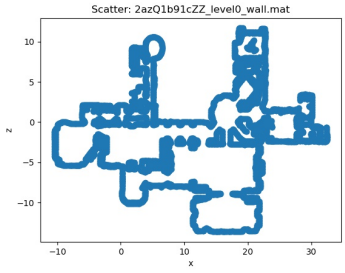

Figure 6: Wall reconstruction using 3D images

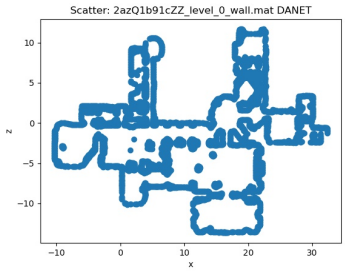

Figure 7: Wall reconstruction using DANet as semantic information extractor

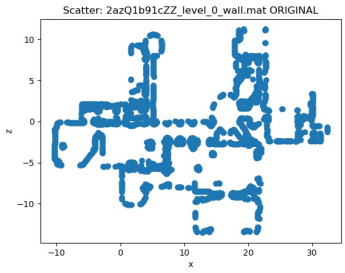

Figure 8: Wall reconstruction using habitat semantic information extractor

3 Results

Figures 6, 7, 8, shows the walls reconstruction, for one house, using three different methods as semantic information extraction. If we consider image 6 as ground truth, then, we can compare how well the other reconstructions, that are represented in figure 7, 8, are wrt to 6. It is clear, that figure 7 is more similar to figure 6. The complete images dataset is available in appendix B

We need a metric to evaluate how good different semantic extraction methods are. The common metric used to evaluate segmentation algorithm is Intersection over Union, also called Jaccard's distance. The metric works as follow: $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ where A, B are two sets. In our case, sets A and B are the sets of wall points in the Cartesian 2D space. Note that only x and z coordinates have been considered, because y is the floor level, and it is constant within the same house. Furthermore, to calculate IoU, we considered A to be the set of wall points of the groud truth, and B to be either the set of wall points of the reconstruction using habitat semantic extractor, or wall points using DANet semantic extractor. Table 1 shows a comparison between, reconstruction with DANet semantic information and habitat semantic information.
It is clear that 5 houses have been reconstructed better with DANet semantic information, while, in other 3 cases, habitat did a better job in extracting semantic information. However, we need to note that the habitat semantic segmentation tool has been trained with images, we have

evaluated. Therefore it has an inner advantage.

Instead, we used DANet pretrained on ADE20K dataset, which has different images. So we could actually be satisfied with this result. To further improve the IoU score, one way would be training DANet on our houses dataset scene images. In order to compare the differences between the image quality of scene houses and ADE20K dataset, two images are available: 31 30

Table 1: IoU metrics comparisons

House Name	**IoU DANet (%)**	**IoU habitat (%)**
2azQ1b91cZZ	35	24
8194nk5LbLH	35	28
EU6Fwq7SyZv	23	35
QUCTc6BB5sX	14	16
TbHJrupSAjP	20	21
X7HyMhZNoso	4	4
x8F5xyUWy9e	24	24
zsNo4HB9uLZ	28	22

The code used to calculate these metrics is available here: [13].

As a further test, PSPNet has been used as semantic information extractor. Actually, it does a good job, at least as well as DANet.

Table 2: IoU metrics comparisons

House name	**IoU PSPNet (%)**	**IoU habitat(%)**
2azQ1b91cZZ	36	24
8194nk5LbLH	35	28
EU6Fwq7SyZv	23	35
QUCTc6BB5sX	16	16
TbHJrupSAjP	21	21
X7HyMhZNoso	2,9	4
x8F5xyUWy9e	23	24
zsNo4HB9uLZ	29	23

Table 2 shows the IoU metric of wall points reconstructed with PSPNet.

Like DANet, we used PSPNet pretrained on ADE20K, so these results can be improved if we could train the network with houses scene images.

4 Conclusion

We evaluated a variety of semantic segmentation networks on a small dataset. Then we chose two of the best networks that could extract semantic information better. The networks used were: DANet and PSPNet. Our purpose was to extract semantic data from houses and reconstruct house's walls from those information. We then evaluated walls reconstructions between three methods used to extract semantic information: habitat sim semantic extractor, DANet, and PSPNet. We understood that DANet and PSPNet in some cases, did a better job than habitat sim extractor. Finally DANet and PSPNet could be improved if trained on habitat sim scene images.

A List of semantic classes habitat extracts

At this link is present a json file with all the classes habitat extracts: [14].

References

[1] Manolis Savva et al. "Habitat: A Platform for Embodied AI Research". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[2] Jun Fu et al. "Dual Attention Network for Scene Segmentation". In: (2019).

[3] Evan Shelhamer, Jonathan Long, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.4 (2017), pp. 640–651.

[4] Hengshuang Zhao et al. "Pyramid Scene Parsing Network". In: *CVPR*. 2017.

[5] MMSegmentation Contributors. *MMSegmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark*. `https://github.com/open-mmlab/mmsegmentation`. 2020.

[6] Marius Cordts et al. “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[7] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[8] Giovanni Bagolin. *Code used for creating CSV for each network*. 2021. URL: https://github.com/gbagolin/walls-reconstruction/tree/master/csv_creation.

[9] Holger Caesar, Jasper R. R. Uijlings, and Vittorio Ferrari. “COCO-Stuff: Thing and Stuff Classes in Context”. In: *CoRR* abs/1612.03716 (2016). arXiv: 1612.03716. URL: http://arxiv.org/abs/1612.03716.

[10] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[11] Giovanni Bagolin. *Code used to extract semantic information from houses scene images*. 2021. URL: https://github.com/gbagolin/walls-reconstruction/tree/master/semantic_segmentation_network.

[12] Marco Cristani’s team. *Code used to reconstruct point cloud, and extract wall*. 2021. URL: https://github.com/gbagolin/walls-reconstruction/tree/master/data_extraction_scene_reconstruction.

[13] Giovanni Bagolin. *Code used to calculate IoU metrics from .mat files*. 2021. URL: https://github.com/gbagolin/walls-reconstruction/tree/master/IoU_metric_calculation.

[14] Giovanni Bagolin. *Habitat semantic classes that are extracted by the simulator*. 2021. URL: https://github.com/gbagolin/walls-reconstruction/tree/master/habitat_sem_classes.

B Compare different reconstructions

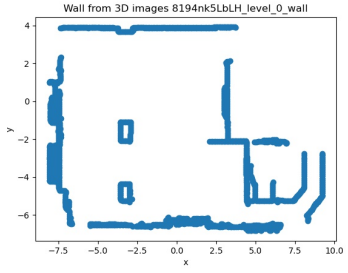

Figure 9: Wall reconstruction using 3D images

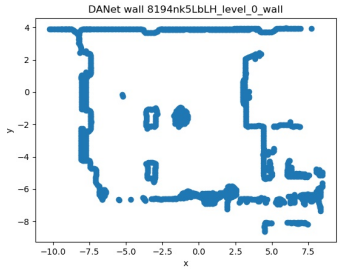

Figure 10: Wall reconstruction using DANet as semantic information extractor

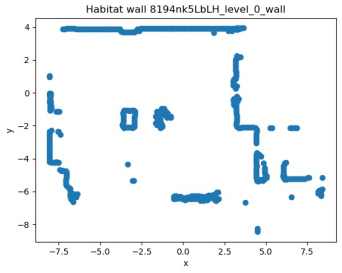

Figure 11: Wall reconstruction using habitat semantic information extractor

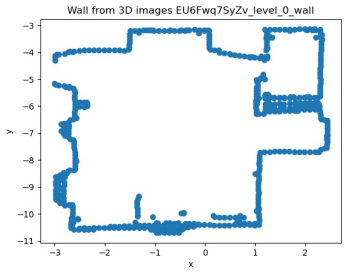

Figure 12: Wall reconstruction using 3D images

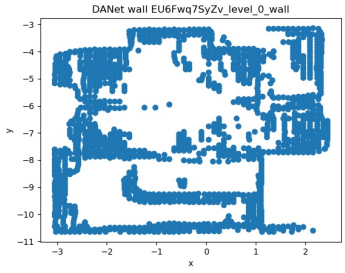

Figure 13: Wall reconstruction using DANet as semantic information extractor

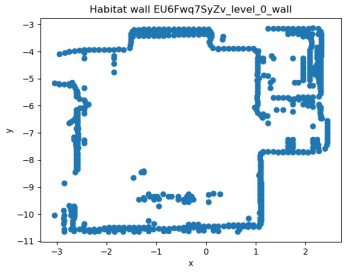

Figure 14: Wall reconstruction using habitat semantic information extractor

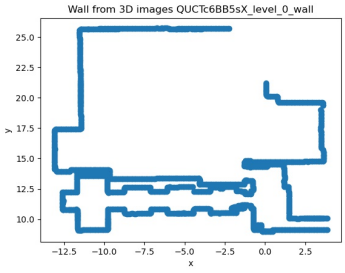

Figure 15: Wall reconstruction using 3D images

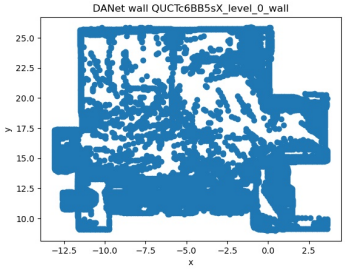

Figure 16: Wall reconstruction using DANet as semantic information extractor

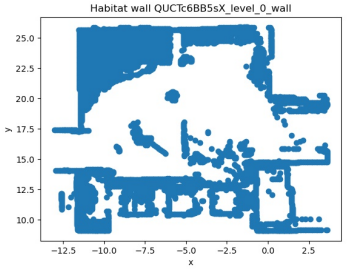

Figure 17: Wall reconstruction using habitat semantic information extractor

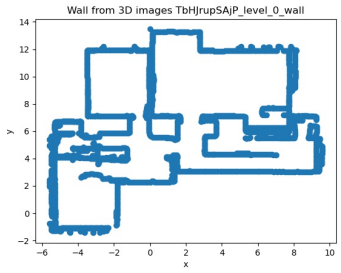

Figure 18: Wall reconstruction using 3D images

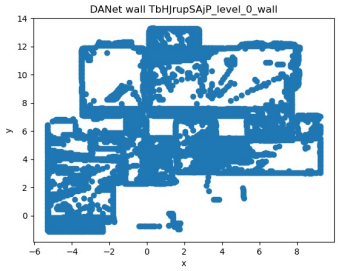

Figure 19: Wall reconstruction using DANet as semantic information extractor

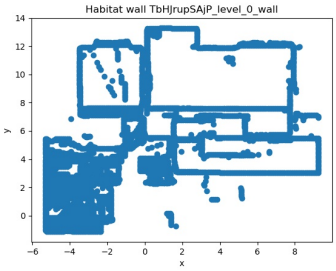

Figure 20: Wall reconstruction using habitat semantic information extractor

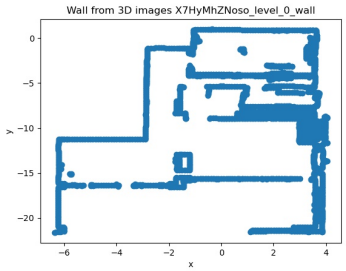

Figure 21: Wall reconstruction using 3D images

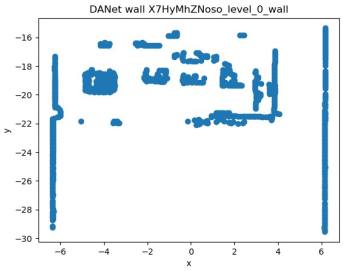

Figure 22: Wall reconstruction using DANet as semantic information extractor

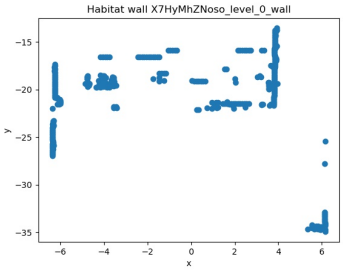

Figure 23: Wall reconstruction using habitat semantic information extractor

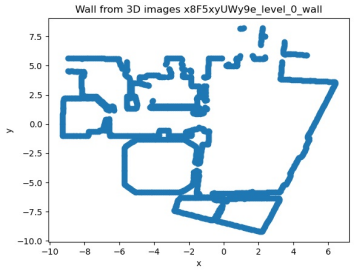

Figure 24: Wall reconstruction using 3D images

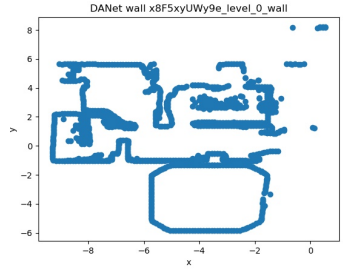

Figure 25: Wall reconstruction using DANet as semantic information extractor

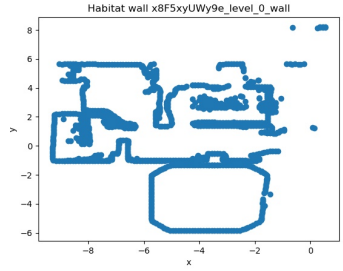

Figure 26: Wall reconstruction using habitat semantic information extractor

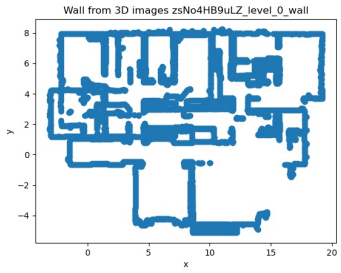

Figure 27: Wall reconstruction using 3D images

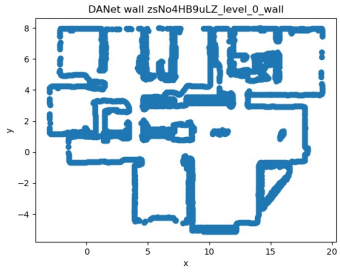

Figure 28: Wall reconstruction using DANet as semantic information extractor

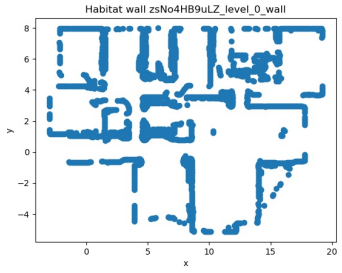

Figure 29: Wall reconstruction using habitat semantic information extractor

Figure 30: Habitat scene image

Figure 31: ADE20K image

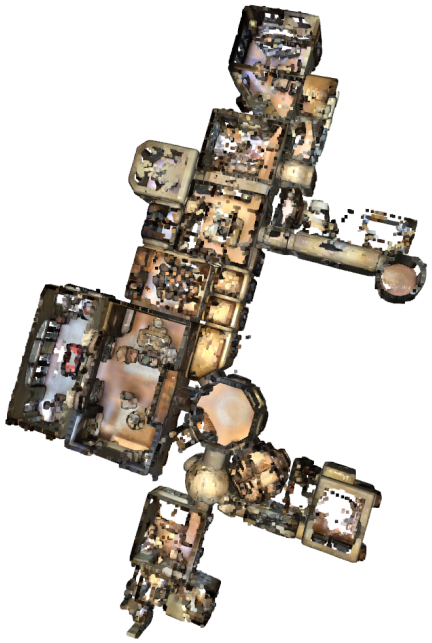

Figure 32: Point cloud reconstruction of house 2azQ1b91cZZ