Single Image Plane Reconstruction using Manhattan World Constraints

Diogo Jordão Rodrigues de Oliveira
Instituto Superior Técnico, Lisboa
diogo.jordao@tecnico.ulisboa.pt

***Abstract*—Many indoor environments have objects with planar proprieties and are arranged as propitious to exploit their planes' normals alignment. These scenarios are ideal for a Manhattan World assumption, stating that all planes in a scene are aligned with one of the three dominant directions. In this master thesis, we propose a novel deep Neural Network, called MW-Net, for Manhattan planes detection and reconstruction, receiving a single RGB image as input. The end-to-end network learns to predict a rotation from the camera to the Manhattan World coordinate system, probabilistic segmentation masks, and an offset/depth map. The proposed method does not have a restriction on the number of planes that can predict. MW-Net was trained on ScanNet, and we extracted over 45000 ground-truth data. It uses a Dilated Residual Network for feature extraction, followed by two ramifications i) Global pooling for rotation prediction; ii) Pyramidal pooling for image segmentation and offset/depth map. MW-Net outperforms PlaneNet on segmentation accuracy, using less architectural complexity, since we do not use a DCRF, unlike PlaneNet.**

***Index Terms*—Manhattan world, Manhattan planes reconstruction, MW-Net, deep Neural Network, plane detection, Dilated Residual Network.**

I. Introduction

Computer vision (CV) is one of the most active research topics in computer science. In the early 2000s, several Machine Learning approaches to CV problems brought some impressive results (see [1]–[3]). These results sparked the interest in ML methods such as Support Vector Machines [4] and, in particular, on neural network architectures [5], [6]. With Deep Learning (DL) emergence in [5], many works have exploited these methodologies, achieving remarkable improvements [7]–[10]. Indeed, DL methods applied to CV topics have been trending, and this relationship translated in many state-of-the-art methods on Object Detection [9], [11], [12], 3D Vision [10], [13], [14], and Tracking [15], [16].

Works [5], [6] in deep neural network architecture have been an essential role in the success of many recent methods, like [9], [11], [17]. Residual networks [6] (or Resnets) made it possible to increase the number of convolution layers, making the neural networks deeper while avoiding the undesirable vanishing gradient problem [18], [19]. This improvement on deep architectures lead to the development of state-of-the-art frameworks for object detection, *e.g.* R-CNN [7], Fast R-CNN [8], Faster R-CNN [11], YOLO [9] or YOLO 9000 [20].

This thesis focuses on the Planes' reconstruction problem, which has been extensively studied in recent works (see [21]–[23]). Although numerous works exploit new DL approaches, non-deep methods use more traditional approaches on Plane Detection topics, such as 3D Piecewise Planar Reconstruction [24] or Semantic Segmentation [25]. One example of a more classical approach is the Manhattan-world Stereo [26], which works under the Manhattan World (MW) constraints, an approach that we also follow.

Concerning DL approaches to Plane detection, methods like PlaneNet [21], PlaneRCNN [22] or PlaneRecover [23] brought significant improvements in terms of accuracy and run-time performance.

Usually, the Human being tends to build objects with planar surfaces on their structures. Many Deep Learning architectures ([9], [11], [17], [20]) can detect these objects, and, consequently, one can use these architectures developed for object detection and extend it to planar surfaces [21], [22]. Although many plane detection methods share some architectural similarities with object detection ones, they also share some problems. For instance, PlaneNet [21] struggles on small plane identification in a crowded planar scene. This difficulty increases with the restriction on the number of planes predicted (PlaneNet only estimates ten planes). Still, PlaneRCNN [22] is an example of significant improvement, having no restriction on the number of planes predicted, allied to an increase on accuracy/time performance. PlaneRCNN uses a more complex architecture than PlaneNet to achieve this purpose. In this master thesis, we propose a novel method that tries to overcome PlaneNet's problems with less complexity than PlaneRCNN.

Therefore, we propose the MW-Net, a novel deep neural network for detecting planes that satisfy the MW constraints. The MW Assumption states that all planes in a scene must be parallel or orthogonal between each other. These planes, Manhattan planes, have their normals aligned with one of the MW coordinate system dominant directions (basis vectors). We are aware that this approach will not recognize some planar surfaces whose normals do not respect the MW constraints. However, since many indoor scenes are composed of a large set of planes aligned with one of the dominant directions, it is possible to reconstruct almost the full planar scene with this approach, trying to neglect the less significant planes. An MW approach gives some flexibility to the proposed method by eliminating any restriction of having a pre-declared number of planes to be predicted.

MW-Net receives an RGB image and outputs: i) a rotation, represented by a quaternion, from the camera to the MW world; ii) probabilistic segmentation masks of each plane; iii) and an offset and depth maps for planar and non-planar, respectively. The quaternion is further converted to a rotation

matrix 3×3 and it is constituted by the MW dominant directions. Since planes' normals are aligned with MW axis, this rotation is used to identify the planes's normal parameters.

II. State-of-the-art

A. Plane detection

Plane detection is a research topic way before DL became a trend. Many research works applied a more traditional approach to this problem [24], [26]–[28]. Manhattan-world stereo (MWS) [26], working within the constrained space of Manhattan-world scenes, uses Multi-view Stereo (MVS) [29] to reconstruct a set of oriented 3D points (positions and normals), where normals extract the dominant axis, and the positions generates axis aligned candidate planes. Candidate planes are going to be used as hypotheses on Markov Random Fields depth-map reconstruction. "NYU-Toolbox" [24] is similar to MWS but does not works within the MW constrained space, and extracts its planes hypotheses using RANSAC [30].

Despite the outstanding results, there was the need to simplify the input requirements, since most of these methods require multiple views or depth information as input.

With the emergence of Deep learning, Plane detection research works [21], [22], [31] start to exploit deep neural network architectures, obtaining remarkable results. Since our method builds on PlaneNet, it will be further described as well as other state-of-the-art methods.

PlaneNet [21] uses a single RGB image as input and predicts plane parameters, their segmentation masks and a non-planar depth map. The network architecture consists of a Dilated Residual Network [32], [33], for feature extraction, followed by two ramifications. The first ramification has a Global pooling followed by a fully connected layer for plane parameter's regression. The second ramification has a pyramidal pooling followed by a convolution layer for image classification and another convolution layer for non-planar depth map modelling. PlaneNet outputs a non-planar depth map, being the planar depth map determined using the plane parameters, only possible knowing the camera intrinsic parameters [34]. Although PlaneNet's remarkable results, it had some limitations such as the number of planes that had to be pre-defined, ten planes per scene.

PlaneRecover [23] also uses a single network for plane detection. It receives an RGB image as input and outputs a planar segmentation map, that segments the input in several planes and non-plane objects and the plane's parameters in 3D space. Similar to PlaneNet it predicts a limited number of planes, only determining five planes per scene. PlaneRecover distinguishes from the other methods by approaching the problem with unsupervised learning, led by difficulties on dataset's ground-truth extraction. A piecewise planar 3D model of the scene can be built, using the network's output.

PlaneRCNN [22] differs from PlaneNet by using a variant of Mask R-CNN [17] for detection, and a refinement network for segmentation Mask improvement. Plane detection is made by predicting each plane parameters and segmentation mask. PlaneRCNN presents a novel loss function, which improves plane-parameter and depth map accuracy via end-to-end training. The referred method presents state-of-the-art results, overcoming PlaneNet's limitation related to the restriction of the number of planes that can be predicted per scene.

Finally but not least, [31] is divided into two stages. In a first stage, it trains a CNN to obtain planar/non-planar segmentation map and pixel embeddings, followed by a mean shift clustering algorithm to generate plane instances. On the second stage, a network branch is trained to predict pixel level plane parameters. It also does not have a restricted number of planes that can be detected.

Both PlaneRCNN and [31] do not have any restriction on the number of planes but they achieve this using a more complex architecture than PlaneNet.

B. Manhattan World assumption

Exploiting environment geometry is not a novel approach, and a MW can take advantage of these characteristics. On 3D reconstruction, there are many MW approaches [26], [35], [36], but this topic is not the only taking advantage of it.

There are research studies using the MW constraints, for instance in navigation [37], [38], where indoor and outdoor scenes are designed on a Manhattan three-dimensional grid. In [37], they state that the important signs for navigation are aligned with one of the directions of MW, and facilitate navigation.

III. Dataset

Many objects are made of planar surfaces, and most of the time, they are arranged with other planar surfaces. An indoor environment layout usually is composed of six planes orthogonal or parallel to each other. Frequently, these planar objects are arranged according to the layout, and many times their planes are aligned with one of the three dominant directions. Situations like these attract MW approaches, which can detect a significant set of planes that have their normals aligned with MW base vectors. The MW base vectors were computed considering the most significant planes, to avoid neglecting many non-constraint planar surfaces.

The MW assumption assumes that all planes in a scene are aligned to one of the three dominant directions. If the plane's normal is aligned, then the plane will be detected. On the other hand, if the plane's normal is not aligned with one MW axis, the plane will be detected but as part of the non-planar region, not being counted as MW plane.

PlaneNet [21] detects planes unconstrained by the Manhattan World restrictions. For PlaneNet training, their authors extracted ground-truth data from the ScanNet dataset, such as the planes' parameters, image segmentation, and the image depth map.

This ground-truth data is not suitable for our network's training since it does not respect the MW constraints. Working over the extraction mentioned, we defined the MW dominant directions to distinguish the Manhattan planes from the non-planar region, in each scene.

A. ScanNet dataset

The ScanNet [39] is a large-scale RGB-D video database of indoor environments. For each scene, this dataset makes available annotations with estimated calibration parameters, camera poses, 3D surface reconstructions, textured meshes, dense object-level semantic segmentations, and aligned CAD models.

For PlaneNet purpose, it was extracted 51000 ground-truth piecewise planar data (50000 for training and 1000 for testing) from ScanNet. For this process, they directly fit planes to 3D points, using RANSAC with replacement, and project them to images. The resulting dataset will make available for each RGB image the image segmentation, plane parameters, image depth, and intrinsic camera parameters.

The resulting planes are not under the MW constrains. For each scene, it becomes necessary to extract the MW planes. With this in mind, it was extracted a rotation from the camera to the MW coordinate frame. This rotation is composed of three dominant directions, and the MW planes' normals are aligned with one of those directions.

1) Manhattan World Assumption & Dataset: The MW assumption states that all planes in a specific scene are orthogonal or parallel to each other, thus planes' normals must be aligned to one of the three dominant directions. The MW coordinate frame defines these dominant directions. The MW base vectors can be arranged to obtain a rotation matrix from the MW coordinate frame to the camera coordinate frame.

To compute the MW base vectors that better suits a specific scene, we had in consideration a similar process as the one presented in [40]. When choosing the MW base vectors, it is desirable to capture the most significant planes. These planes have the largest number of pixels assigned in the image segmentation. For instance, considering a room, and having a broader view of the division, these planes are usually a wall or the floor.

Considering that, after the PlaneNet's dataset processing, we have access up to twenty planes per scene, a planar segmentation of the image and the image depth map. A MW base vectors were computed for each scene individually.

Initially, it was computed how many pixels were assigned to each plane, using the image segmentation, and the inner products between it and all the others 19 planes. Notice that this process is made to all the planes. From the inner products, we obtain the orthogonal planes to each plane.

The first base vector determined is the one associated with the MW X-axis. It is set with the normal of the most significant plane in the scene, *i.e.* with more pixels assigned, under the condition of having at least one orthogonal plane on the scene's image segmentation. If the condition is not fulfilled, this process is repeated to the second largest plane and so on. The MW Y-axis is the second base vector defined, and it assigned the normal of the largest plane from the list of planes whose normal are orthogonal to the MW X-axis. Finally, MW Z-axis is defined by the cross vector between the MW X-axis and Y-axis base vectors. As it is possible to realize, scenes must have at least two orthogonal planes; otherwise, they are discarded.

The X and Y planes' normals are not strictly orthogonal since we gave a threshold to the inner product, 0.1, below which two planes are considered orthogonal. With this in mind, if we organize the base vectors X,Y and Z as columns, we obtain a pseudo-rotation matrix $R^{MW} = [X, Y, Z]$ which points to the need of projecting it to the closest matrix on the SO(3) group.

A SO(3) matrix must respect the following conditions,

$$\begin{aligned} R^T R = R R^T = I \\ det(R) = 1. \end{aligned} \tag{1}$$

We applied the Singular Value Decomposition (SVD) to the pseudo-rotation $R^{MW} = [X, Y, Z]$,

$$[U \Sigma V^T] = SVD(R^{MW}) \tag{2}$$

Being now trivial to obtain the rotation from the camera to the MW coordinate frame,

$$R^C_{MW} = U diag(1, 1, det(UV^T)) V^T \tag{3}$$

where U and V^T are unitary matrices. R^C_{MW} is the rotation from the Manhattan World to the camera coordinate frame.

Once having the R^C_{MW} is now easy to find the rotation from the camera to MW coordinate frame, which is given by its inverse,

$$R^{MW}_C = {R^C_{MW}}^{-1}. \tag{4}$$

It is now possible to apply the MW constraints on each scene, and distinguish which surface is planar or non-planar. This data treatment was made as follows.

At this stage, the planes' parameters available are represented regarding the camera coordinate frame's origin. To verify if a plane's normal is aligned to one of the MW base vectors, we need to have the planes' parameters seen by the origin of the MW frame's origin. We can easily achieve this by applying the rotation R^{MW}_C. Considering a plane's normal considering the camera coordinate frame's origin, N^C, it is now possible to obtain

$$N^{MW} = R^{MW}_C N^C, \tag{5}$$

where N^{MW} is the plane's normal seen by the MW coordinate frame.

To distinguish planes from non-planar surfaces, we have to verify which planes' normals are aligned with one MW axis. For this purpose, it was done the inner product between the planes' normals and each one of the MW base vectors, $[1, 0, 0], [0, 1, 0]$ or $[0, 0, 1]$. If any of the three inner products were over a pre-established threshold, 0.9, the normal would be considered aligned with the respective axis. From that point, the plane's normal would be replaced by the MW base vector which is aligned. However, if none of the inner products was over 0.9, the plane will become part of the non-planar class. The image segmentation is then updated based on this knowledge.

The image segmentation is obtained by assigning the specific class to each pixel. There will be four classes, one for each MW dominant direction and one for the non-planar

region. Pixels that are assigned with a class $C \in [1, 2, 3]$ will belong to a Manhattan plane and the pixels with a class $C \in [4]$ will belong to the non-planar region.

For PlaneNet's purpose, their authors predicted the plane's normal and offset as a three-parameter vector, and the offset was the vector's norm. Still, since we are using MW base vectors to identify planes, predicting a normalized three-parameter vector, we only know the plane's normal but not the respective offsets.

Knowing the Manhattan planes and their mapping in the image, we need to make an association class/offset to each planar pixel. We find a solution to this problem, creating an offset/depth map for each scene.

The offset/depth map ground-truth was obtained by intersecting the MW with the original image segmentation, from which we get MW planes segments.

If a plane is a Manhattan plane, we assign the respective offset to their pixels, otherwise we assign the respective depth value to the pixel. To the non-planar region pixels we assign the respective depth value. This makes it possible to obtain the offset/depth map desired.

After the ground-truth extraction, we add the rotation from the camera to the MW coordinate frame, as a quaternion, Q_C^{MW}, the image segmentation with four classes, and the offset/depth map to the ScanNet dataset, allowing the network to train as expected.

If the rotations were predicted as 3×3 matrices, it would not be possible to guarantee that the network's rotations outputted would fill the SO(3) group requisites. A possible approach would be to project it to the rotation group, but this solution would be computationally more complex.

IV. MW-Net: A plane detection network with Manhattan World constraints

MW-Net is a novel method for plane detection. It takes a 192×256 RGB image, 3 channels, as input and outputs a rotation quaternion, from the camera to the Manhattan World (MW) coordinate frame, four probabilistic plane segmentation masks, and an offset/depth map.

The rotation quaternion, a 1×4 vector, guarantees a rotation belonging to the SO(3) group. The rotation quaternion can be further converted to a 3×3 matrix. This rotation is composed by the MW dominant directions seen by the camera coordinate frame, and we can easily obtain the Manhattan planes' normals making use of it.

For the image segmentation, the network predicts four probabilistic segmentation masks, where each pixel is assigned with four probabilities. There is one probability for each possible class: i) planes' normal is aligned with MW X-axis; ii) planes' normal is aligned with MW Y-axis; iii) planes' normal is aligned with MW Z-axis; iv) planes' normal is not aligned with any of the MW axes. The pixel belongs to the class that has the largest value. The resulting segmentation image output shape is 192×256.

The offset/depth map is a 192×256 matrix where each planar pixel is assigned the offset of the plane it belongs to. For non-planar pixels, it is assigned a depth value.

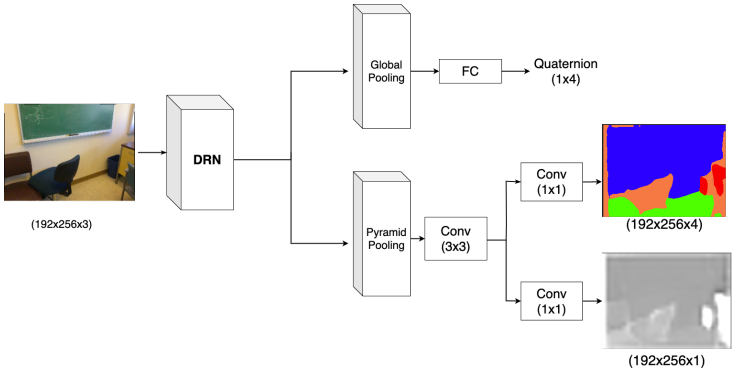


Fig. 1: MW-Net's Architecture. It is constituted by a Dilated Residual Network (DRN), for feature extraction, and by two ramifications, a global pooling for rotation quaternion prediction and a pyramidal pooling [43] for the plane segmentation mask and a offset map prediction.

The network predicts all Manhattan planes, without any conditionality on the number of planes that can be estimated. Our network only detects and reconstructs planes aligned with one of the MW dominant direction. Indoor environments are propitious to this approach due to the large set of orthogonal and parallel planes in each scene, lifting the mentioned restriction.

MW-Net uses a single neural network for the whole process, implemented in Pytorch (see [41], [42]).

A. Architecture

MW-Net's is a novel deep neural network, and its structure can be seen in Figure 1. It is constituted by a Dilated Residual Network (DRN), for feature extraction, and two ramifications: i) a global pooling for quaternion rotation prediction; ii) and a pyramid pooling [43], which, for its turn, ramifies in plane segmentation mask and an offset/depth map prediction branches. Over this chapter, when talking about convolution layers, it is represented its kernel size between parentheses. For example, a convolution layer with kernel size 3 is represented as Conv(3×3). From now on, when talking about tensors shapes, it is referred to their shapes as $width \times height \times channels$.

As already referred, the DRN in Figure 1 is a feature extractor block. DRN has a similar structure to Resnet, but instead of using standard convolutions in some layers, it applies dilated convolution [44]. Dilated convolutions are convolutions where the kernel elements are spaced from each other, skipping some input points. For example, $D = 2$ means that the kernel elements have a gap between them; $D = 3$ means they are spaced by two. The DRN used is a DRN-D-54, which has 35.8M parameters.

The DRN-D-54 structure, represented in figure 2, was divided into seven levels. In level 0, there is a single standard Conv(7×7), and it is the starting layer that receives the RGB image. Level 1 has two convolutions, being the first one a standard Conv(3×3) and the second one is a Conv(3×3), with stride 2.

Levels 2, 3, 4, and 5 have bottleneck blocks in their structure. A bottleneck block [6] has 3 convolution layers,

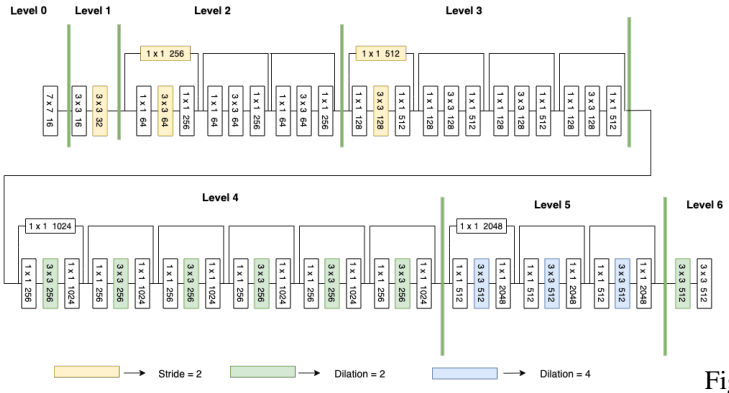


Fig. 2: DRN-D-54 architecture. It is divided in seven levels. Bottleneck blocks form levels 2, 3, 4, and 5. Convolution with color yellow have stride 2, color green have dilation 2, and blue have dilation 4.

TABLE I: DRN-D-54 level description

Level	Channels in	Channels out	Bottleneck blocks
0	3	16	0
1	16	32	0
2	32	256	3
3	256	512	4
4	512	1024	6
5	1024	2048	3
6	2048	512	0

starting with a Conv(1×1), followed by a Conv(3×3) and a Conv(1×1). Generally, the last layer outputs four times more channels than the first two, i.e., if the first two layers output 64 channels, the last layer will output $4 \times 64 = 256$ channels. The bottleneck block has a shortcut from its input to the outcome, allowing to skip the three convolution layers described, giving the block some flexibility to be just the identity if needed. This property helps to avoid the undesirable vanishing gradient problem.

Table I shows the number of output channels for each level, and the number of bottleneck blocks. Level 2 and 3 have 3 and 4 bottleneck blocks respectively, and their first block has the second Conv layer with stride 2. Level 4 and 5 have six and three bottleneck blocks, respectively, and the second convolution layer of each block is a dilated one, by 2 and 4 respectively. Finally, layer six is formed by two convolution layers. The DRN outcome will be of shape $24 \times 32 \times 512$ and will feed the Global pooling and Pyramid pooling block.

Global pooling in Figure 1 is simply an average pooling in two dimensions. The average pooling kernel has shape 24×32, resulting in an output of shape of $1 \times 1 \times 512$. Making this result going through a fully connected layer with 512×4 parameters, it results in the desired quaternion. This is another reason for using the quaternion for predicting the rotation. If we predicted the rotation as a 3×3 matrix, the fully connected would have 512×9 parameters, increasing its complexity.

Pyramid Pooling is a more complex block than Global Pooling, see figure 3. It receives the DRN's feature map and applies four different average pooling to it, using different kernels sizes. After pooling, each output goes through a standard convolution, with size kernel of 1, that outputs 128 channels. Each output goes through an upsample to obtain shapes equal to the input one, 24×32. Finally, the pooling's outputs are concatenated to each other, and with the Pyramid pooling input, obtaining a final tensor shaped as $24 \times 32 \times 1024$.

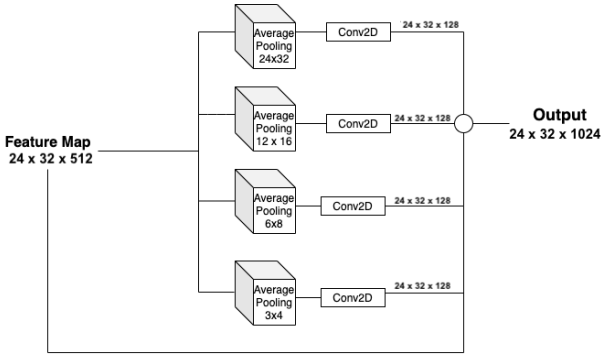


Fig. 3: Pyramid pooling architecture. It receives the DRN's feature map and applies four different average pooling to it, using different kernels sizes. After pooling, each output goes through a standard convolution, with size kernel of 1, that outputs 128 channels. Each output goes through an upsample to obtain shapes equal to the input one, 24×32. Finally, the pooling's outputs are concatenated to each other, and with the Pyramid pooling input, obtaining a final tensor shaped as $24 \times 32 \times 1024$.

Finally, Pyramid's output goes through a standard convolution, with a kernel of size 3, outputting 512 channels. For the Segmentation Masks, there is a standard convolution with a kernel size of 1, and since there are four different classes, it outputs four channels. Bearing in mind that the output is of shape $24 \times 32 \times 4$, it is necessary to upsample it so that the resulting shape is $192 \times 256 \times 4$. For the Offset map, instead of a convolution that outputs four channels, a convolution outputs a single output followed by the bilinear interpolation.

B. Training

MW-Net was trained on an indoor environment dataset, ScanNet, with ground-truth extraction for the sake of MW assumption. For training, it was used a pre-trained model of PlaneNet, trained by us for 50 epochs, since the only pre-trained model available was for their network implemented on Tensorflow [45] . This pre-trained model does not use the Dense Conditional Random Fields (DCRF) [46], used on PlaneNet, to refine the Segmentation prediction.

The proposed model was trained on a GPU GeForce GTX 1070 over 40 epochs, and it was used 46710 samples for training. Network's learning was mini-batch learning, and each batch has eight data samples. Using mini-batch learning, the model updates its weights on a higher frequency; in the MW-Net case, this happens in every mini-batch, i.e., the model updates every eight samples from training data. Each sample includes the RGB image and the respective ground-truth data (quaternion, image segmentation, and offset/depth map), and the camera intrinsic parameters. The RGB image will be the network's input, and the ground-truth data will be part of the loss, as a reference to the network's outputs. Since GPU RAM

is limited, Mini-batch learning is useful because it allows good memory management.

The optimizer used for training was the Adam Optimizer [47] with an initial learning rate set to 3×10^5. The optimizer is responsible for minimizing the loss, updating the model weights, and Adam computes adaptive learning rates per parameter.

C. Loss

The problem at hand is a multi-task learning problem, our network predicts three different outputs at the same time, resulting in three different losses, one for each output. As it is possible to see in section IV-B, the network learning is a mini-batch one, and the batch size, B, is 8 samples of data training. The overall loss is given by the sum of the three losses, regarding the mini-batch in question, leading to

$$Loss = \sum_{i}^{B} Loss^i_{quat} + Loss^i_{seg} + Loss^i_{offset}, \quad (6)$$

where $Loss_{quat}$ is the quaternion loss, $Loss_{seg}$ is the segmentation loss, and $Loss_{offset}$ is the offset loss. The Losses are represented as $Loss^i$ that is the Loss of the i^{th} sample of the mini-batch.

D. Quaternion Loss

Quaternion prediction is a simple regression problem and its approach is made with a l_1 norm,

$$Loss_{quat} = \|(Q - Q^*)\|_1, \quad (7)$$

where Q is the quaternion predicted, Q^* is the quaternion ground-truth. The reason of the prediction being a quaternion, over being a rotation matrix 3×3, is because the quaternion guarantees that the rotation predicted belongs to the SO(3) group, adding to the fact that there are less network parameters that need to be predicted. Considering each prediction's number of parameters, it will be more efficient to predict four parameters than nine.

E. Segmentation Loss

Image segmentation is a classification problem, where to each pixel it is given four probabilities. These four probabilities encode how certain the network thinks the pixel belongs to a specific plane. Classes 1, 2 and 3 correspond to planes that has their normal align to the axis [1,0,0], [0,1,0] and [0,0,1] from MW coordinate frame, and class 4 corresponds to a non-planar region.

For this purpose it was used a cross entropy loss,

$$Loss_{seg} = \frac{1}{K} \sum_{p=0}^{K} -\log \left(\frac{\exp\left(m^{(p)}_{class}\right)}{\sum_{j=0}^{C} \exp\left(m^{(p)}_{j}\right)} \right) \quad (8)$$

which has a softmax inside the logarithm operation. In equation 8 ($m^{(p)}_{class}$ is a probabilistic value predicted by the network for pixel p and class is the class target which the pixel belongs to. ($m^{(p)}_{j}$ is a probabilistic value predicted by the network for pixel p and class j, C is the number of classes, 4, and K is the number of pixels $K = 192 \times 256$.

F. Offset/depth map Loss

For the offset/depth map loss, we use a squared l2 norm,

$$Loss_{offset} = \frac{1}{K} \|(O - O^*)\|_2^2 \quad (9)$$

where O is the offset/depth map, O^* is the offset/depth map ground-truth, and $K = 192 \times 256$ pixels.

V. Results

As previously mentioned, MW-Net is able to reconstruct a planar scene with a single RGB image. It is a competitive method and is an innovative method for MW planes detection. We compared MW-Net with the PlaneNet method. For this comparison, it was applied the two recall metric from [21]. The comparisons were made with the original PlaneNet model, their Github repository, which includes the dense conditional random field (DCRF), for segmentation refinement. MW-Net was trained for 40 epochs, but the model with the best results was on the 26^{th} epoch. For the comparisons made we used the best model.

VI. MW-Net outputs

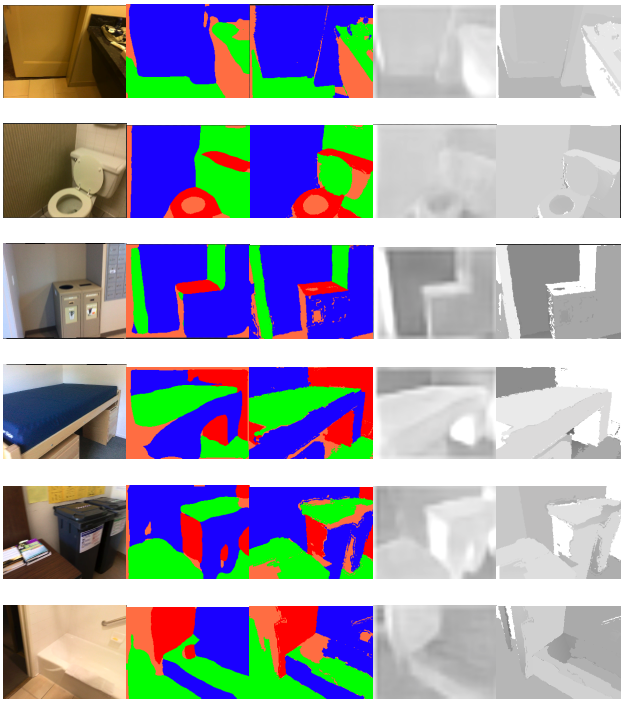
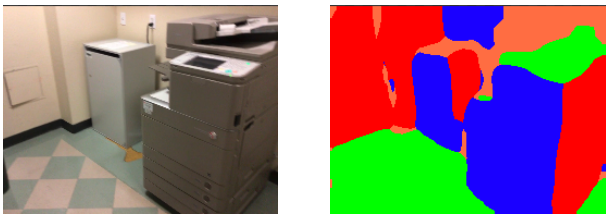
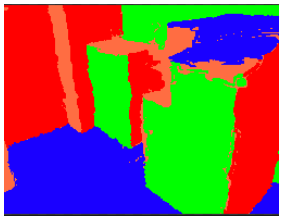


Fig. 4: MW-Net outputs and respective ground-truths. The first images column, from left to right, is the inputs of the network, followed by segmentation predictions, second column, and segmentation ground-truths, third column. The last two columns, from left to right, are the offset/depth-map prediction and the offset/depth-map groundtruth. As it is possible to infer, the segmentation results are very close to the ground-truth.

The proposed network predicts four probabilistic segmentation masks classes, three planar classes, one for each MW



(a) input (b) Segmentation



(c) Segmentation-gt

Fig. 5: Example of image segmentation when classes are swapped comparing to ground-truth. It is possible to see that in image segmentation prediction the classes represented by the color green and blue are swapped, relatively to the ground-truth.

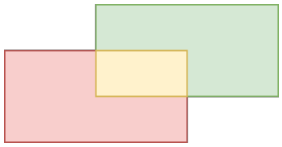
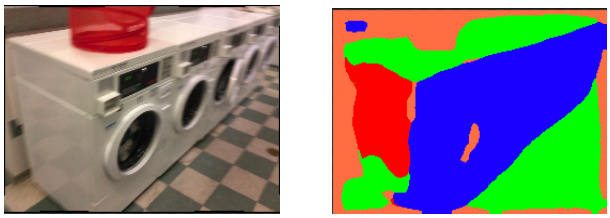


Fig. 6: IOU example. Representation of two planes intersecting. The IOU between these two planes is the ratio between the number of pixels on the intersection of both planes (yellow region) over the number of pixels on the union (red + yellow + green)

axis, and one non-planar class. To each pixel, it is assigned the class with a higher value between the four classes. Looking at the segmentation on figure 4, it is possible to distinguish four colors, corresponding each to a class. The blue color correspond to the MW X-axis, the green color to Y-axis, and the red color to Z-axis. The orange color corresponds to the non-planar class. The network presents incredible results on the segmentation branch, but there is margin to improve. It is possible to see, in many segmentation images, that there are some pixels on the left side that are classified incorrectly, being those pixels classified as non-planar. This can happen for many reasons, one of them may be due to the training dataset, that can have many data elements that are promoting over-fitting. This is one of the aspects that need to be improved in further developments. Nonetheless, MW-Net presents 80,75% of planar accuracy, while PlaneNet has 73,52%.

The metric applied for this accuracy is based on the IOU, see legend figure 6, between the ground-truth with the inferred plane. Having in count the figure 6, the metric applied for each plane is the number of pixels that are in the intersection of the plane ground-truth with the plane inferred, number of pixels of yellow surface, over the total number of pixels that belong



(a) input (b) Segmentation

(c) Segmentation-gt

Fig. 7: In this figure it is represented an bad labeled example. In this figure it is represented a bad labeled example. In figure 7 (a) it is represented the networks input, and on 7b (b) and 7 (c) it is represented the segmentation predicted by the network and the ground-truth, respectively. Cases like this decreases the network segmentation accuracy.

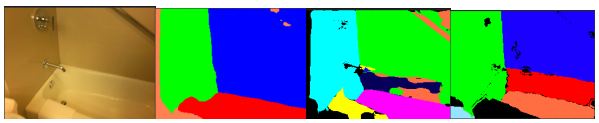
to the plane ground-truth, represented by the number of pixel on yellow+green surfaces.

To be fair with the network's real performance, we have to pay attention to cases illustrated in figure 5, where the segmentation is well made but the classes are swapped. In figure 5 it is possible to verify the dependency of the segmentation on the quaternion prediction. Observing figure 5 (b), it is evident that the class represented by the blue color is swapped with the class represented by the green color, when comparing with the segmentation ground-truth in figure 5 (c). This image segmentation, although having classes swapped, it is segmenting planes correctly.

In order to the metric work as supposedly, to ground-truth planes we have to infer predictions that overlap the most with them. Obviously, two different ground-truth planes cannot have the same plane inferred. The association is made through the IOU, i.e., given a plane prediction it is computed the IOU with all the planes ground-truth, and this plane is associated to the ground-truth with which has the highest IOU.

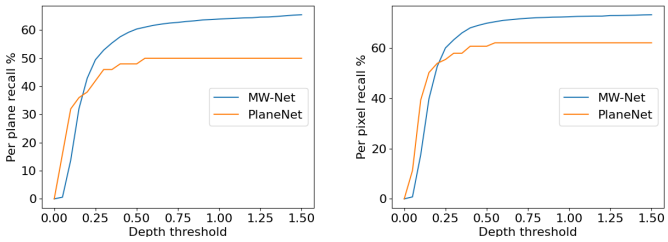
Although the high segmentation accuracy rates, the segmentation results may be harmed by some bad labelled data such as the one in figure 7. In the figure there are three images, figure 7 (a) is the network's input, and in figure 7 (b) and 7 (c) are the image segmentation from the network and ground-truth, respectively. It is obvious that the ground-truth is not accurate, and the segmentation by the network is under the expectation.

In figure 4, the fourth column and fifth column, counting from left to right, are the offset/depth map prediction and ground-truth, respectively. The offset/depth map is basically a offset mapping for planar surfaces and depth for non-planar surfaces, as it was explained. Being a regression problem, the network is responsible to predict 192×256 values, one for each



(a) Input (b) MW-Net (c) PlaneNet (d) Groudtruth

Fig. 8: In this figure it is possible to see the comparison of segmentation between MW-Net and PlaneNet, figures 8 (b) and 8 (c) respectively. It is possible to see that the segmentation does not miss any plane having the indoor scene totally identified.



(a) Plane Recall Accuracy (b) Pixel Recall Accuracy

Fig. 9: In this figure it is possible to compare segmentation evolution in function of Depth threshold using two recalll metrics. The comparison was made between MW-Net PlaneNet. MW-Net significantly outperforms PlaneNet in both metrics.

pixel, resulting, naturally, in some outliers. In addiction to this problem, for planar regions, since it is predicted an offset for each pixel, it is natural that a large number of pixels will have the same offset, and if this prediction it is not correct the error will increase since it is spread to the remaining pixels.

Concluding, it is possible to notice that innumerable indoor scenes presents many planar surfaces that are parallel and/or orthogonal between each other. Having a MW approach to the problem can simplify proposed objective, improving some results. In the images presented, the majority of planes were well predicted and it is possible to achieve remarkable results with it. In the next section it is made the comparison with the PlaneNet method.

A. *MW-Net vs PlaneNet*

PlaneNet outputs are the plane parameters, the image segmentation, and a non-planar depth-map. The plane parameters are the plane's normal and offset, in the form offset$\times$normal, only needing three parameters to identify the planes. But since their parameters depend on the offset, it struggles to distinguish parallel planes with different offsets, when these planes are close from each other, harming their segmentation results. This can happen, for instance, because of network difficulties on distinguishing different textures. PlaneNet only predicts ten planes per scene, if there are a crowded planar scenes, it will fail to perform as expected. In its turn, MW-Net uses MW base vectors to identify the planes' normals, being the segmentation independent of the offset, and the offset is offered by the offset/depth map prediction. The network do not have a limitation on the number of planes that can be predicted.

In figure 8, it is possible to see the comparison between PlaneNet and MW-Net. The ground-truth represented in the figure 8 (d), segments the image in the same way PlaneNet does, where planes with different offsets are identified by different classes. In figures 8 (c) and 8 (d), the non-planar class is represented by the black color. The difficulty on identifying parallel planes with different offsets, when they are close from each other, is evident in the figure 8 (c), where it struggles to identify distinguish the planes represent by the colors red and blue on figure 8 (d), while MW-Net do not face this problem.

To compare against PlaneNet, it is applied two recall metrics to both methods, the same as in [21]. To understand the metric it is necessary to have a notion of what it is the Intersection Over Union IOU, see legend figure 6. For the figure 9 (a), the metric presented is the percentage of the correctly predicted ground-truth planes. A ground-truth plane is correctly predicted if the IOU with the inferred plane is over 0.5 and the mean offset/depth difference, from the overlapping region, is less than a given threshold. The offset/depth difference is the difference between the offset/depth of plane prediction pixels and the corresponding plane ground-truth pixels.

The second metric, figure 9 (b) is the number of pixels, that are in the overlapping regions, over the total number of pixels from all the planar surfaces in the scene, being similar to the metric presented in section VI. This measure it is not the same as the one in the previous section, because only the pixels that are in the planes well predicted by the metric in figure 9 (a), will count as pixels well predicted. This means that if the IOU between the planes ground-truth and the plane inferred is lower than 0.5, and they have pixels in common, this pixels will not count as well predicted. Although the denominator still is the total number of planar pixels in the scene.

In figure 9, it is presented the MW-Net and PlanNet performance when applies the metrics described previously. We vary the depth threshold from 0 to 1.50, and it is possible to verify that MW-Net significantly outperforms PlaneNet, except when the depth threshold is small and PlaneNet can fit planes accurately for those thresholds, lower than 0.2. It is seen that, considering threshold values above 0.2, the MW-net outperforms significantly, meaning that our image segmentation is much better than PlaneNet, but PlaneNet outperforms MW-Net on depth prediction.

MW-Net obtain these results with less architecture complexity than PlaneNet. Although PlaneNet uses a single network for planar reconstruction, on segmentation branch, it uses a dense conditional random field (DCRF) (see [46], and train the DCRF module with precedent layers (see [48]), as a way to refine segmentation results. MW-Net outperforms PlaneNet without using any DCRF, as it is possible to verify in the comparisons made.

VII. Conclusion

This thesis presented a novel method for planar reconstruction using a MW approach. MW-Net receives an RGB image as input and outputs a rotation matrix from camera to MW

coordinate frame as a quaternion, four image segmentation probabilistic masks and an offset/depth map. MW-Net predicts planes segments with high accuracy rates, and without any restriction on the number of MW planes that can predict. It was proven that MW approach is reliable since the innumerable quantity of planes that are parallel/orthogonal to each other, and almost all planar surfaces were detected. MW-Net outperforms PlaneNet, a state-of-the-art method, in terms of segmentation, achieving remarkable results. MW-Net not just outperform PlaneNet, also it does it with less network architecture complexity.

As future work, a comparison with PlaneRCNN and PlaneRecover should be made. There are space for improvements, such as on offset/depth map. On segmentation, there are miss-classified pixels, on most scene images left side, that need to improve.

REFERENCES

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2001, pp. I–I. 1

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2005, pp. 886–893. 1

[3] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008, pp. 1–8. 1

[4] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998. 1

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017. 1

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. 1, 4

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587. 1

[8] R. Girshick, "Fast r-cnn," in *IEEE Int'l Conf. Computer Vision (ICCV)*, 2015, pp. 1440–1448. 1

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *cvpr*, 2016, pp. 779–788. 1

[10] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 652–660. 1

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 91–99. 1

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conf. Computer Vision (ECCV)*. Springer, 2016, pp. 21–37. 1

[13] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, 2015, pp. 922–928. 1

[14] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 770–779. 1

[15] N. Soans, E. Asali, Y. Hong, and P. Doshi, "Sa-net: Robust state-action recognition for learning from observations," in *IEEE Int'l Conf. Robotics and Automation (ICRA)*, 2020, pp. 2153–2159. 1

[16] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4293–4302. 1

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *IEEE Int'l Conf. Computer Vision (ICCV)*, 2017, pp. 2961–2969. 1, 2

[18] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994. 1

[19] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256. 1

[20] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7263–7271. 1

[21] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, "Planenet: Piece-wise planar reconstruction from a single rgb image," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2579–2588. 1, 2, 6, 8

[22] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, "Planercnn: 3d plane detection and reconstruction from a single image," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4450–4459. 1, 2

[23] F. Yang and Z. Zhou, "Recovering 3d planes from a single image via convolutional neural networks," in *European Conf. Computer Vision (ECCV)*, 2018, pp. 85–100. 1, 2

[24] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European Conf. Computer Vision (ECCV)*, 2012, pp. 746–760. 1, 2

[25] C. Zhang, L. Wang, and R. Yang, "Semantic segmentation of urban scenes using dense depth maps," in *European Conf. Computer Vision (ECCV)*, 2010, pp. 708–721. 1

[26] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Manhattan-world stereo," in *cvpr*. IEEE, 2009, pp. 1422–1429. 1, 2

[27] D. Gallup, J.-M. Frahm, and M. Pollefeys, "Piecewise planar and non-planar stereo for urban scene reconstruction," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1418–1425. 2

[28] S. Sinha, D. Steedly, and R. Szeliski, "Piecewise planar stereo for image-based rendering," 2009. 2

[29] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2006, pp. 519–528. 2

[30] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981. 2

[31] Z. Yu, J. Zheng, D. Lian, Z. Zhou, and S. Gao, "Single-image piecewise planar 3d reconstruction via associative embedding," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1029–1037. 2

[32] Yu, Fisher and Koltun, Vladlen and Funkhouser, Thomas, "Dilated residual networks," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 472–480. 2

[33] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014. 2

[34] R. I. Hartley, "Self-calibration from multiple views with a rotating camera," in *European Conf. Computer Vision (ECCV)*. Springer, 1994, pp. 471–478. 2

[35] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Reconstructing building interiors from images," in *IEEE Int'l Conf. Computer Vision (ICCV)*, 2009, pp. 80–87. 2

[36] E. Delage, H. Lee, and A. Y. Ng, "Automatic single-image 3d reconstructions of indoor manhattan world scenes," in *Robotics Research*, 2007, pp. 305–321. 2

[37] J. M. Coughlan and A. L. Yuille, "Manhattan world: Compass direction from a single image by bayesian inference," in *IEEE Int'l Conf. Computer Vision (ICCV)*, vol. 2, 1999, pp. 941–947. 2

[38] P. Denis, J. H. Elder, and F. J. Estrada, "Efficient edge-based methods for estimating manhattan frames in urban imagery," in *European Conf. Computer Vision (ECCV)*, 2008, pp. 197–210. 2

[39] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5828–5839. 3

[40] R. Guo, K. Peng, D. Zhou, and Y. Liu, "Robust visual compass using hybrid features for indoor environments," *Electronics*, vol. 8, no. 2, p. 220, 2019. 3

[41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, 2019, pp. 8026–8037. 4

[42] N. Ketkar, "Introduction to pytorch," in *Deep learning with python*. Springer, 2017, pp. 195–208. 4

[43] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2881–2890. 4

[44] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015. 4

[45] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016. 5

[46] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," *Advances in neural information processing systems*, vol. 24, pp. 109–117, 2011. 5, 8

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 6

[48] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *IEEE Int'l Conf. Computer Vision (ICCV)*, 2015, pp. 1529–1537. 8