UNIVERSIDADE D
COIMBRA

Dylan Ângelo Lopes

MULTI TO SINGLE PPR

UNSUPERVISED GENERATION OF PPR DATA FROM IMAGE COLLECTIONS FOR LEARNING SINGLE-IMAGE PPR

VOLUME 1

Dissertation supervised by Professor Dr. João Pedro de A meida Barreto and Dr. Michel Antunes, submitted to the El ectrical and Computer Engineering Department of the Facul ty of Science and Technol ogy of the University of Coimbra, in partial ful fil l ment of the requirements for the Degree of Master in El ectrical and Computer Engineering, branchof Automation.

July 2020

Abstract

Automatic dense 3D reconstructions has long been a challenge in computer vision, being fundamental for a wide variety of applications, e.g. robotics, object recognition, etc. However, developed algorithms have difficulties in handling poorly textured or specular surfaces slant, etc. To overcome these issues, many authors employ the planarity assumption, since most of these challenges occur in man-made environments which are predominantly composed by planar surfaces. The resulting models have a better accuracy and a lower complexity, which is important for real-time applications rendering. With the advances in deep learning (DL), recovering depth from a single image (SIDE) has become a major research topic in computer vision, achieving recently high performance indicator. Though, these algorithms still have important limitations in terms of accuracy and generalization. Very recently, DL based approaches were proposed for single-image piece-wise planar reconstruction (SI-PPR), requiring a single RGB image and the camera intrinsic parameters for computing a piece-wise planar segmentation and the respective planar equations. These algorithms improve over the performances obtained with the traditional approaches based in geometric reasoning, which usually require multiple-views. However, these approaches require large training datasets and the existing ones are relatively small. For this purpose, we aim to create a new pipeline for PPR data generation that is completely automatic, allowing to generate training data for SI-PPR in an unsupervised manner. The generated data was evaluated and experimentally compared with Ralho's dataset [1], created with a semi-automatic pipeline that requires manual labeling of key frames. At last we re-trained a DL-based SI-PPR approach (PlaneRCNN) and evaluated its performance, proving that it is possible to obtain similar accuracy performance as approaches that require time-consuming manual labeling.

Resumo

A reconstrução densa 3D tem sido um grande desafio na área de visão por computador, tendo um papel fundamental numa grande variedade de aplicações, e.g. robótica, deteção de objetos, etc. No entanto, os algoritmos desenvolvidos têm algumas dificuldades em lidar com superficies com pouca textura, espelhadas, etc. Para ultrapassar estes problemas, muitos autores têm assumido a geometria planar, sendo que a maior parte destes desafios se encontram em ambientes construídos pelo ser humano, predominantemente compostos por superfícies planares. Os modelos resultantes têm uma maior precisão e são menos complexos, um aspecto importante para a renderização em aplicações de tempo real. Com os avanços em *deep learning* (DL), a estimativa de profundidade a partir de uma única imagem (SIDE) tem sido um tópico importante na investigação em visão por computador, atingindo mais recentemente, uma alta performance. Contudo, estes algorímtos ainda têm algumas limitações em termos de precisão e generalização. Foram propostas recentemente abordagens para a estimativa de reconstrução planar, a partir de uma única imagem (SI-PPR), sendo que a rede apenas necessita de uma única imagem RGB e dos parâmetros intrínsecos da câmara, processando a segmentação planar e as respetivas equações dos planos. Estes algorítmos superam as abordagens tradicionais baseadas em métodos geométricos que, geralmente necessitam de múltiplas frames. No entanto, estes algorítmos requerem muitos dados de treino, sendo que os mesmos ainda são bastante limitados. Posto isto, pretendemos criar um novo *pipeline* para geração de dados PPR de uma forma totalmente automática, possibilitando a geração de dados de treino de uma forma não supervisada. Os dados gerados foram analisados experimentalmente comparados com com o *dataset* do Ralho [1], criado com um algorítmo semi-automático que requer labeling manual de imagens-chave. Por último, treinámos um algorítmo baseado em DL para a estimativa de PPR a partir de uma única imagem e avaliámos a sua performance, comprovando que é possível obter uma precisão parecida com os resultados obtidos usando métodos que requerem *labeling* manual, sendo o mesmo demorado.

Contents

List of Tables

List of Figures

Acronyms

AI - Artificial intelligence

BT - Birchfield-Tomasi dissimilarity

CNN - Convolutional Neural Networks

DL - Deep-Learning

DL-PPR - Deep-learning approaches for piece-wise planar reconstruction

DRN - Dilated residual network

MRF - Markov Random Field

MVS - Multi-view Stereo

NCC - Normalized cross-correlation

PPNPR - Piece-wise planar and non planar reconstruction

PPR - Piecewuse-planar reconstruction

RGB - Red-blue-green

RGB-D - Red-blue-green frame together with depth information

SfM - Structure from Motion

SI-PPR - Single-image piece-wise planar reconstruction

SIDE - Single-image depth estimation

Chapter 1

Introduction

1.1 Motivation

Depth estimation is a crucial task in order to perceive and navigate in a 3D scene [6]. Geometric relations allow a better understanding of the environment and objects that appear in the field-of-view [7], being fundamental for a wide variety of applications, such as object recognition [8], robotics [9], etc. Due to this reason, 3D reconstruction is a relevant problem in computer vision, robotics and artificial intelligence approaches [3,9–13]. In the past few decades, quite a few approaches have been developed for modeling a 3D scene from 2D images [7, 14–16]. These approaches usually rely on geometric reasoning, where they are usually divided in two different stages: (1) establishing correspondence between points in different frames [7] and then, (2) via triangulation, determine their position in 3D space [6,7]. However, these algorithms have some difficulties in handling situations of poor and repetitive textures, variable illumination, surface slant and specularities [2, 7, 14, 17], which results in depth maps with noise, errors and holes [18]. These effects are shown in Fig.1.1, where we present the 3D reconstruction from two different views, where low textured surfaces contain a lot of depth reconstruction ambiguity. Additionally, we can easily observe the amount of existent noise on the 3D reconstruction.

Figure 1.1: Example of a low textured surface (blue wall in first image) and the resulting 3D reconstruction from different views: front-view (second image) and top-view (third image). This data belongs to the Coimbra dataset.

These challenges generally occur in urban scenes, indoor places, etc. Since most of these scenes are man-made environments, and those are predominantly composed by planar surfaces, many authors [2,3,18–22] employ the planarity assumption in order to overcome these issues. The objective is to obtain a Piecewise-planar reconstruction (PPR) of the scene that is more accurate, geometrically simpler and visually more compelling than point-based approaches [2, 18, 19, 21, 22].

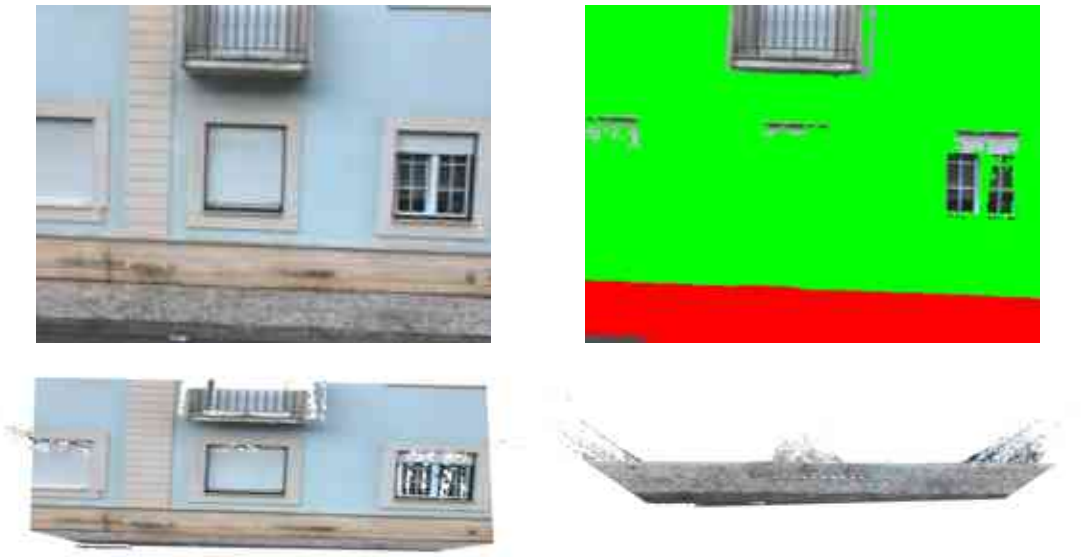

Figure 1.2: Example of PPR segmentation. First row - RGB image and planar segmentation. Second row - 3D Reconstruction where: (1) depth values were computed from plane equations for planar regions and (2) the original depth values were used for the remaining pixels. On this row we can observe the 3D reconstruction from two different views (front and top views, respectively).

As previously mentioned, traditional approaches based on geometric reasoning [2, 18–22] either require multiple views and/or depth information as input, together with the respective camera model and parameters. The general idea of PPR is to generate plane hypotheses by fitting planes to 3D cues, and then assign a plane label to image pixels using global inference [3, 12]. In Fig. 1.2 we can observe an example of PPR segmentation, and a 3D reconstruction after the depthmap refinement process, using the planar equations obtained with our PPR algorithm.

With the advances in Deep Learning (DL) through the years, recovering depth using trained neural networks has been a major task in 3D computer vision [12]. The existing solutions [11, 23], exhibit state-of-the-art performance when evaluated in benchmark datasets [6, 24, 25]. However, the available solutions still have important limitations in terms of accuracy and generalization when employed in real applications [10]. T. Koch et al. [10] presented a new evaluation of Convolutional Neural Networks (CNN)-based single-image depth estimation (SIDE) methods. For this purpose, the authors created a new evaluation dataset, acquired from various indoor scenes, containing high-resolution RGB images together with highly accurate depth maps obtained using laser scans. They also introduced a set of new interpretive error metrics with the objective of analysing different characteristics in depth maps, which are crucial for many applications. With these new performance measures, the experiments have shown that the prediction of planar surfaces is lacking accuracy. Additionally, the edges present in the depth maps tend to be over-smooth for most state-of-art SIDE algorithms [10].

Another task in the computer vision field that as received considerable attention is the planar structure segmentation from a single RGB image [3], which is an ill-posed problem due to depth ambiguity, requiring rich scene prior [3]. However, human vision has an outstanding perceptual capability in understanding 3D scenes when observing a 2D image [12]. Very recently, inspired on this idea, DL based approaches were proposed for PPR (DL-PPR) by suggesting the use of CNNs or other architectures to (1) identify planar surfaces, and (2) estimate their parameters in 3D space [3, 12, 13]. In this type of approaches, the authors design a deep neural architecture that, during the training,

receives as input images and the respective calibration information, together with label and/or plane annotations. The network is optimized by comparing its output with respect to ground-truth depth information and/or image plane labeling [3, 12, 13].

For this purpose, inspired by Megadepth [9], a SIDE data generator, our objective is to create a PPR data generator that does not need any human intervention, enabling the automatic generation of large-scale training datasets for SI-PPR DL networks.

1.2 Contributions

In contrast to SIDE [11, 23], where there are techniques [9] for generating large-scale datasets, the available data for training DL-PPR approaches is limited. Due to the lack of training data with annotated 3D planes, there is a high demand of the community to create new datasets and benchmarks in order to train/test this type of networks [13]. Due to this reason, PlaneNet [12] had to create a new DL-PPR training dataset from ScanNet, an RGB-D video dataset annotated with 3D camera poses, surface reconstructions, and instance-level semantic segmentations, which were manually annotated [26]. However, each depthmap respective to a single RGB-D frame contains failures (holes), as well as their quality gets worse at far distances [12]. For this purpose, PlaneNet [12] developed a new pipeline that fit planes to a consolidated mesh, projecting them back to each frame together with the associated semantic annotations.

Further, in PlaneRCNN [3] a new dataset was built, based on the PlaneNet benchmark. The main problem is that the size of the training dataset is relatively small (containing about 100k frames), due to the use of ScanNet dataset [26], which requires manual labeling.

The starting point for this thesis was the thesis of Ralho [1], that proposes a semi-automatic pipeline for generating PPR data, developed last year in the Instituto de Sistema e Robótica (ISR). The thesis of Ralho had as objective to create PPR datasets for training DL based SI-PPR approach. The main drawback of Ralho's pipeline [1] is that it requires the manual labeling of keys frames, which are selected by the algorithm according to various factors, described in Section 2.3.2. The steps of the labeling process are illustrated in Figure 2.8.

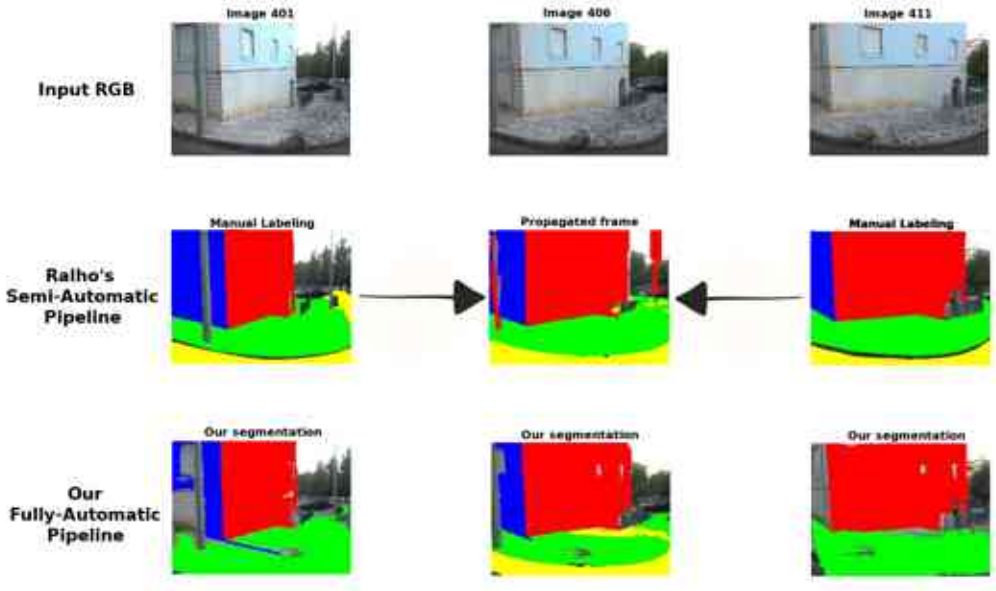

Figure 1.3: Example of Ralho's PPR dataset [1]. (a) Original RGB image. (b) Key-frame used for propagation of the current frame. (c) Final PPR result of the image (a).

In contrast to Ralho [1] that, inspired by Megadepth [9], created an algorithm that requires human intervention, our main objective was to develop a new pipeline for PPR data generation for training SI-PPR that is completely automatic.

During the research work, we started by evaluating and benchmarking existing multi-view PPR approaches based on geometric vision that do not require training, e.g.,Ralho's pipeline [1] and D. Gallup et. al. [2]. First results showed that this thesis is a proof-of-concept, being possible to generate accurate PPR data for training DL in a completely automatic way. Therefore, we combined the resulting data from COLMAP (refer to Section 3.2.2), with D. Gallup et. al. [2] pipeline and two DL based object segmentation networks (3.2.1). The next stage was to train a state-of-art SI-PPR architecture, PlaneRCNN [3], using the generated training data.

Finally, we have evaluated the network performance, demonstrating that using the proposed training data generator it is possible to obtain similar accuracy performance as approaches that require time-consuming manual labeling.

In summary the contributions are:

- We have trained and tested a neural network, namely PlaneRCNN [3], with Ralho's PPR data [1], validating the purpose of this thesis.
- The creation of a new fully-automatic pipeline that does not require any human intervention.
- Generation and evaluation of a new PPR dataset with the created pipeline.
- Training PlaneRCNN with the generated PPR data.
- Evaluation of the network performance, showing that it is possible to obtain similar accurate performance and generalization with our data, when compared to a neural network trained with Ralho's data [1]. It is important to mention that Ralho's pipeline require time-consuming manual labeling [1].

1.3 Organization

This thesis is organized in the following manner:

Chapter 2 reviews the relevant literature on three main topics: geometric multi-view PPR, where we present some related work, on the field of PPR data generation. Then we'll approach some DL based PPR networks. Lastly, we'll talk about the Megadepth [9], which inspired us in the creation of a fully automatic pipeline for data generation.

Chapter 3 presents and describes the proposed PPR pipeline. Additionally, we will present some qualitative results and perform a quantitative evaluation.

In Chapter 4 we will re-train PlaneRCNN [3], with the whole pre-processing implementation, required for the custom training. Furthermore, we will exhibit some experimental results and evaluation.

In Chapter 5 we will take some conclusions about the whole scientific project, including: the quality of PPR data generated, how successfully was the re-training of PlaneRCNN [3], etc. Additionally we will present some ideas for future work in order to improve results and to obtain a better generalization.

Chapter 2

Literature review

Over the past decades, computer vision researchers placed great emphasis in the understanding and 3D modelling of a scene [2, 27]. The goal of multi-view 3D reconstruction is to infer geometrical structure of a scene captured by a collection of images [27].

The usual input to many algorithms is a set point correspondences from multiple views. From point matches between frames, it is possible to determine the motion of the camera and obtain a first projective reconstruction of the scene. Using auto-calibration, the camera is calibrated and the scene is transformed to its true Euclidian structure. Finally, knowing the projective structure of a scene, it is possible to obtain a dense point match between frames and, via triangulation, create a dense 3D model of the imaged scene [27].

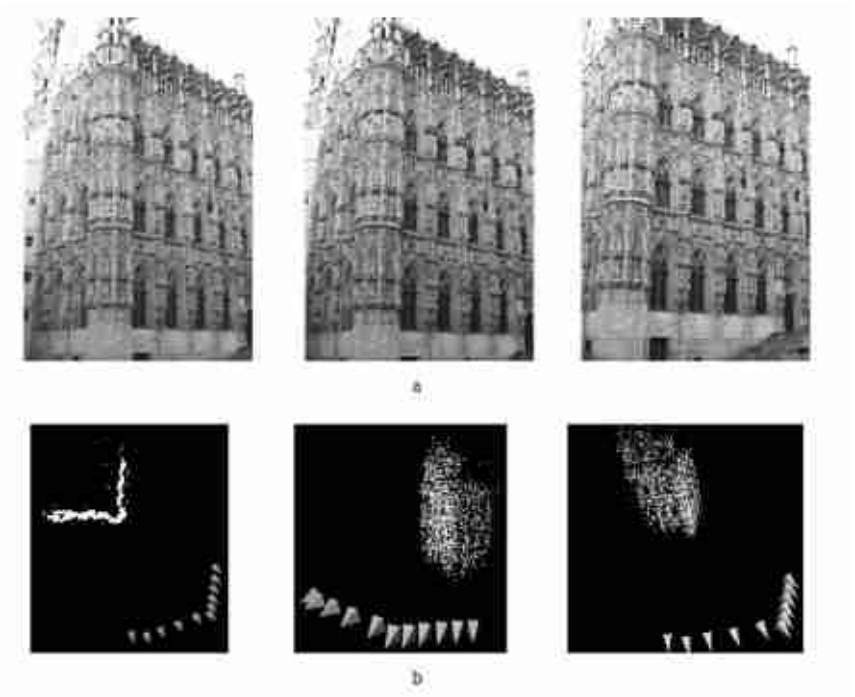

Figure 2.1: (a) Three high resolution images (3000x2000pixels) from a set of eleven of the city hall in Leuven, Belgium. (b) Three views of a Euclidean reconstruction computed from the image set showing the 11 camera positions and point cloud. Figures courtesy of Hartley [27].

It is possible to reconstruct scenes from a single image, using techniques that involve the analysis of features, e.g. parallel lines, in order to determine the affine structure of the scene. Knowledge about orthogonal lines or planes, can be used to upgrade the affine reconstruction to Euclidian, however such techniques are not fully-automatic. This process have been used to reconstruct 3D models from old-master paintings [27].

Figure 2.2: Single view reconstruction.(a) Original painting – St. Jerome in his study, 1630, Hendrickvan Steenwijck (1580-1649), Joseph R. Ritman Private Collection, Amsterdam, The Netherlands. (b)(c)(d) Views of the 3D model created from the painting [27].

With the advances in artificial intelligence (AI), CNNs have been used in the last years with the objective of modeling and training 3D reconstructions of a scene, using a single RGB image [28, 29].

2.1 Geometric-based PPR

For 3D plane detection and reconstruction, most traditional approaches [2, 19, 20] require multiple views and/or depth information as input [3]. Raposo [19] receives as input a sequence of images acquired by a calibrated stereo rig, retrieving both camera motion and PPR. This method starts by computing a semi-dense PPR model using a simplified version of Antunes method [30]. Moreover, plane primitives are generated over point correspondences using a RANSAC-like algorithm. Then, an energy-based multi-model fitting algorithm is used in order to refine and optimize the results. Antunes [20] created a pipeline that generates PPR models using only two calibrated views. The pipeline starts by using a SymStereo framework [31] "for matching pixels across stereo views using symmetry analysis instead of traditional photo-consistency" [20]. The algorithm uses M virtual cut planes, that intersect the baseline of the stereo rig, computing M joint energies that contains the matching cost of pairs of pixels. This energy is then used to detect line cuts, which are lines with a certain possibility of being the intersection between a virtual cut plane and the planar surfaces in the scene. Following this step, these lines cuts are combined and processed to select the most likely planes and refine their pose. Lastly, a standard Markov Random Field (MRF) algorithm assigns the detected planes to image pixels, obtaining a semi-dense PPR and a dense 3D model of the scene.

Furuwaka [21] requires planes to be orthogonal, due to the use of a specific Manhattan-World and Sinha [18] uses a general piece-wise planar model, where both these approaches have difficulties handling non-planar surfaces.

2.1.1 Gallup piece-wise planar and non planar reconstruction (PPNPR)

The algorithm of Gallup [2], is a traditional PPR approach that receives as input RGB images, together with the respective depthmaps and camera parameters.

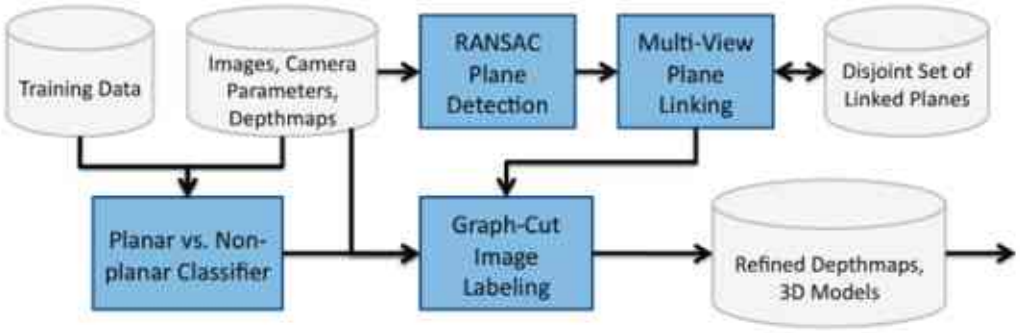

Figure 2.3: Gallup's piecewise planar and non-planar stereo system [2]

Gallup's approach [2] has three main steps, layed out in Fig. 2.3:

- **Plane Hypothesis Generation** - Plane hypotheses are generated for every input frame using a RANSAC method that will be sequentially executed in order to obtain a set of planes for each image. A plane model is defined by selecting three random points in the depthmap: (1) the first point is randomly selected over the image, (2) the other two points are selected from a normal distribution centered at the first point, with a preset standard deviation. In order to rate each plane, instead of simply scoring by the inlier count, they are scored according to the MLE-sac method [32].
- **Multi-View Plane Linking** - After the plane hypothesis are generated for each frame, similar planes in different images are linked using a 3D similarity analysis approach. Each time planes are linked, a new plane hypotheses is obtained as the least-squares fit to the inlier points belonging to both planes and data is saved into a disjoint set data structure. This disjoint set is used to ensure that planar surfaces seen in multiple frames have the exact same plane hypothesis. It also enables the linkage of single planes which appear disjoint in the images due to occlusion.
- **Graph-Cut Labeling** - A MRF problem is set up, where each pixel is assigned with a label corresponding to one of the previously obtained plane hypotheses. A non-plane label is added for labeling non-planar regions and objects. The Label likelihoods are then defined as the photo consistency induced by the plane homography, in case the plane label is assigned. Additionally, a classifier was trained to differentiate surfaces that appear planar, and those that do not. At last, a multi-label graph-cut method [32] is used in order to assign labels to every unlabeled pixels that belong to a certain planar surface.

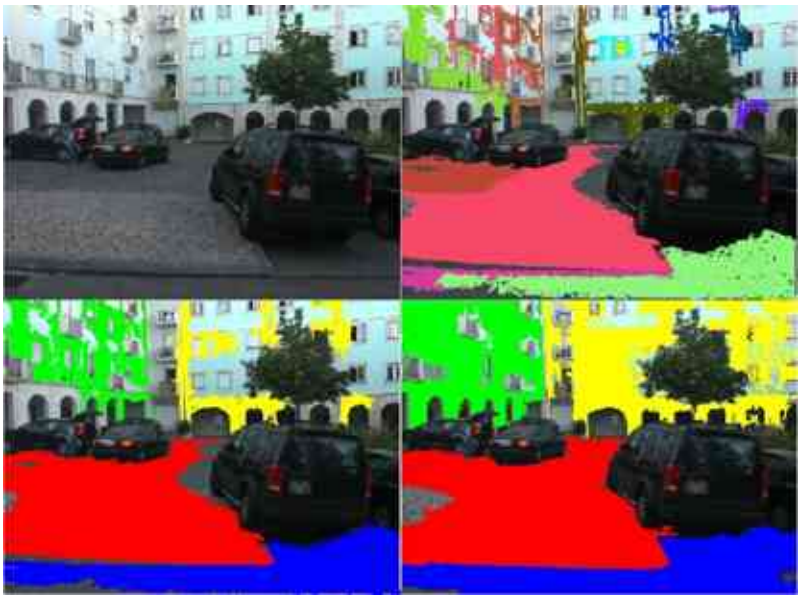

Figure 2.4: Gallup PPR: example illustrating the various stages. First row: (left) Original RGB image, (right) RANSAC plane detection. Second row: (left) Multi-view Plane-Linking, (right) Graph-Cut Image Labeling.

2.2 DL-based PPR

In the last few years, DL based PPR [3,12,13] has been a focus in computer vision research. Recently, PlaneRecover [13] and PlaneNet [12] introduced the use of CNNs and formulate the problem as a plane segmentation tasks. However, these approaches require a maximum number of planes in a single image a priori and both have poor generalization [3].

PlaneRecover [13] proposed a novel end-to-end trainable DL network to detect 3D planes from a single image. The algorithm simultaneously predict plane parameters and the respective segmentations. For training purposes, the authors used the SYNTHIA dataset [33], containing a wide-range number of photo-realistic synthetic images. Thus, depthmaps are free of noise, enabling the use of a simpler algorithm to obtain plane hypothesis, called J-Linkage [34], similar to the RANSAC technique. Further, PlaneRecover achieves real-time performance, being suitable for a wide range of applications [13].

PlaneNet [12] built a novel DL network upon dilated residual networks (DRNs), which is a flexible framework for both image classification and semantic segmentation. From the DRN output feature maps, the authors composed three output branches for three prediction tasks: (1) predict a K number of planar hypotheses, estimating the respective plane parameters from which it is possible to create a depth image. (2) The algorithm model non-planar structures and creates a standard non-planar depthmap. (3) Finally, this pipeline also outputs segmentation masks for the respective estimated planes. As mention previously, due to the lack of PPR data needed for training DL-based frameworks, the authors created a new benchmark from the ScanNet dataset, briefly described in Section 1.2.

2.2.1 PlaneRCNN

Succeeding PlaneNet [12], Liu et. al. created PlaneRCNN [3], where they found that Mask R-CNN works surprisingly well in detecting planes, requiring a single input RGB image. Mask R-CNN was originally designed for semantic segmentation, where instances have vary categories (e.g., person, car and more), but on our case the problem only has two categories: "planar" and "non-planar". CNNs have been successful for depthmap

and surface normal estimation however, direct regression of plane offset turns out to be a challenge [3]. This way, instead of direct regression, the authors solve the problem in three steps: (1) predict a normal per planar instance (modifying Mask R-CNN), (2) estimate a depthmap for an entire image, and (3) use a simple algebraic formula to calculate the plane offset.

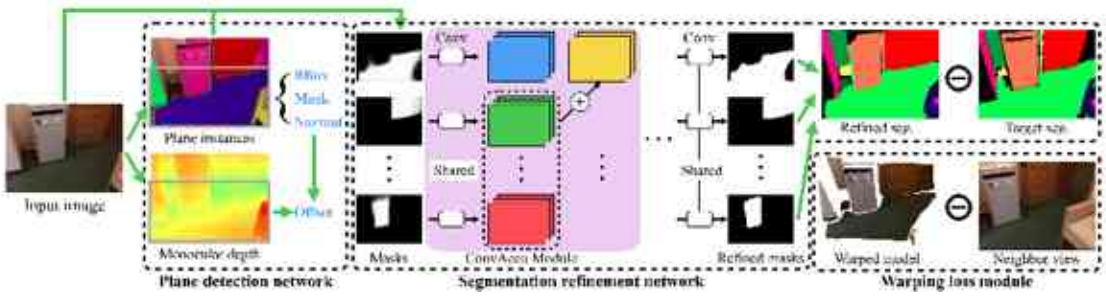

Figure 2.5: Pipeline of PlaneRCNN framework [3]

The second stage of PlaneRCNN (Fig. 2.5) [3] is a segmentation refinement, where a neural network optimizes all the segmentation masks. Finally, the warping loss module enforces the consistency of reconstructed planes with a second view of the same scene. This module is only executed during the training phase, improving the accuracy of plane parameters and depthmap estimation.

In order to train and test PlaneRCNN, the authors followed the steps described in PlaneNet [12] and created a new benchmark from RGB-D videos in ScanNet [26]. Additionally, they added some modifications to recover more fine-grained planar regions. In the images of Fig. 2.6 we can observe some examples, obtained by running PlaneRCNN [3] with three example images, provided by the authors.

Figure 2.6: Results (second row) obtained by running PlaneRCNN [3] over the images shown in the first row.

2.3 Generation of training data

2.3.1 MegaDepth

As mentioned in Section 1.2, DL based approaches require large amounts of training data [35]. Recently, Megadepth [9] implemented a pipeline with the objective of creating a large dataset in order to train a DL based depthmap estimation. The authors first downloaded Internet photos and then obtained a 3D reconstruction using COLMAP [36,37], producing high quality depthmaps for each photo and also estimating the camera parameters.

However, these depthmaps have significant noise and outliers. To address these problems, the authors developed a modified multi-view stereo (MVS) algorithm in order to counter the adverse effect where the "background depths can tend to "eat away" at foreground objects", Li et. al. [9] (see Fig. 2.7).

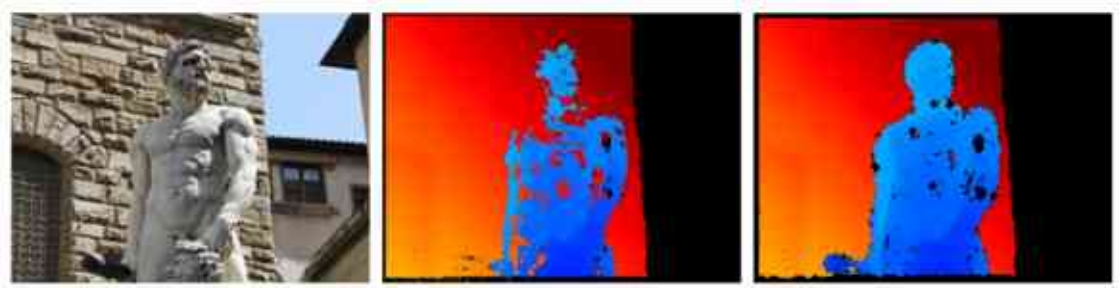

Figure 2.7: Comparison between depthmap results. Original RGB image (first column). Depthmap obtained with COLMAP (second column). Refined depthmap (third column).

Second, the authors enhanced depth via semantic segmentation, using PSPNet [38]. This way, object depths that are not viable for 3D reconstruction are deleted. Also, images where only a small part of the respective depthmap is considered valid, are removed from the original dataset.

Megadepth [9] evaluated their pipeline by re-training a DL approach, namely VGG [39], with the obtained dataset. The results were very satisfying, demonstrating that this data can be used to predict state-of-art depthmaps for other locations, never observed during training, as well as generalizing well to other datasets.

Inspired by Megadepth [9], the proposed work also tackles the problem of large-scale 3D data generation. In addition we went to go one step further, and model the 3D scene using planes, obtaining high accuracy PPR.

2.3.2 Ralho's PPR data generator

Ralho's algorithm [1] uses data from the Coimbra dataset, requiring at an initial phase, human intervention in order to define regions on key frames that belong to planar surfaces (Fig.2.8). The number of key frames depends on various factors, e.g. camera rotation, frame capture rate versus translation speed of the camera, dynamic objects (people, cars, etc.). Moreover, small planar regions (e.g. windows, doors, etc) will only be taken into account if the user assigns a label for each region, which is a longer and exhausting task.

Following this first step, a RANSAC method is used to generate plane hypotheses for each region of these previously labeled frames. Then, the algorithm propagates the obtained planes to neighbour frames using the available extrinsic parameters and depth These planes are then projected into neighbour frames, where a RANSAC algorithm is executed again for the estimation of the new respective planes.

The main drawback of [1] is that it requires the manual labeling of key frames (Fig. 2.8), taking about thee minutes to fully label each frame, relying on the number of planes

and details on the image.

Given that, in average, the user needs to label a key frame for each twenty-five images, it would take about eight days of exhaustive work to generate a hundred-thousand images, which is impractical for generating large-scale datasets required for training deep networks (in the order of millions) [9, 35].

(a) Input RGB (b) Manual Labeling

Figure 2.8: Input RGB (left) and the respective fully labelled frame (right), where each color refer to a different plane.

Chapter 3

Automatic generation of PPR data for SI-PPR

This chapter presents a algorithm for generation of PPR data in a complete automatic unsupervised manner, selecting Gallup's PPNPR [2] as the state-of-art algorithm, which is the base of our pipeline.

Just like Gallup's PPNPR [2], the algorithm receives as input a sequence of RGB images, together with the respective depthmaps and camera parameters. Additionally, two DL architectures mentioned in Section 3.2.1, were used with the objective of detecting objects and background, e.g. sky, creating masks that will be used to filter the input depthmaps, avoiding the detection of planes belonging to objects, etc.

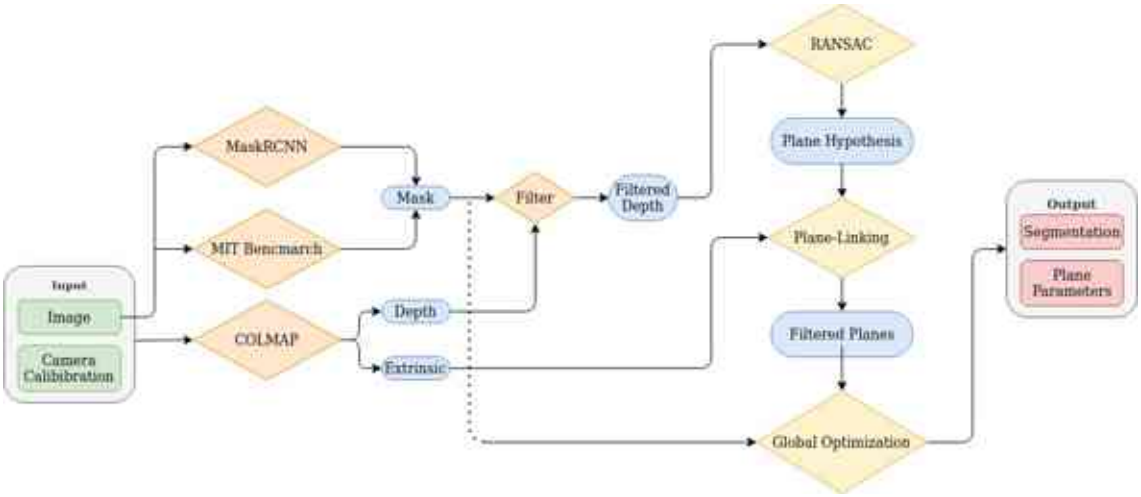

Figure 3.1: Our piece-wise planar data generator system.

3.1 Multi-view data

For the whole investigation we have used a dataset called Coimbra [40, 41], which contains various sequences of images taken in diverse places in the city of Coimbra. These images were taken by mounting a stereo camera setup on a car with well known intrinsic parameters. However, there is no information about any ground truth depth maps or extrinsic parameters. To obtain this information, a state-of-art SfM system, called COLMAP [36, 37] was used, which will be described in the section (Section 3.2.2). In the figure bellow (Fig. 3.2) we can observe some examples of this dataset, together with the respective 3D reconstructions estimated by COLMAP algorithm [36, 37].

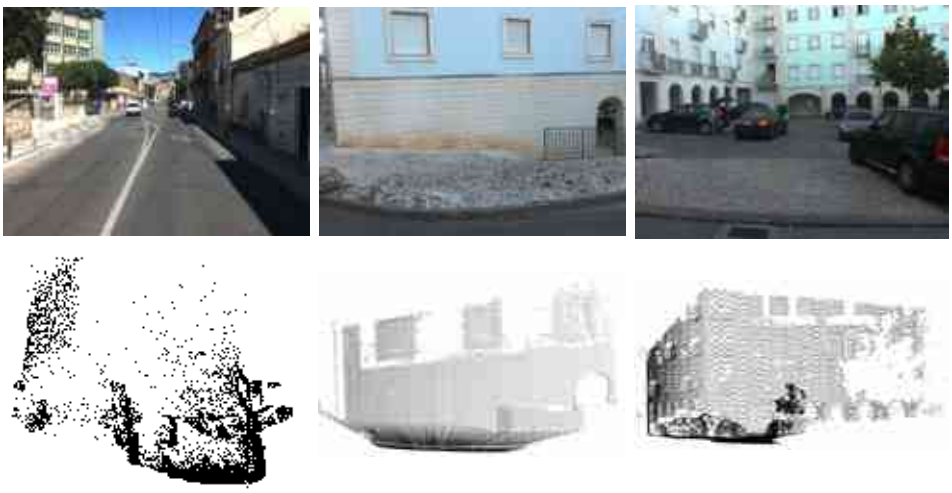

Figure 3.2: (1st row) Examples of the Coimbra dataset and (2nd row) the respective depthmaps obtained using COLMAP.

3.2 Manipulation of data using standard methods

3.2.1 Detection of objects and background

As mentioned before, two DL networks were used with the objective of detecting unwanted depth information, which do not belong to planar surfaces: MaskRCNN [4] and MIT Scene Parsing Benchmark [5].

MaskRCNN [4] was used by PlaneRCNN [3] to implement a variant of the algorithm for SI-PPR. MaskRCNN [4] is a deep neural network that efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance [4]. This architecture was used in order to detect outdoor objects, e.g. persons, cars, etc. However, MaskRCNN [4] was not designed for background detection neither detect trees/vegetation. For this purpose we decided to use a novel network design called Cascade Segmentation Module [5], from MIT Scene Parsing Benchmark. This network was proposed to parse a scene into stuff, objects, and even object parts. Additionally, the authors created a new dataset, with a total of ∼25k images, annotated in detail, with 150 object and stuff classes. We have implemented a new script that create masks from the resulting parsing data, selecting segmentations with the label 'sky' and 'tree' which, in some occasions, also includes vegetation.

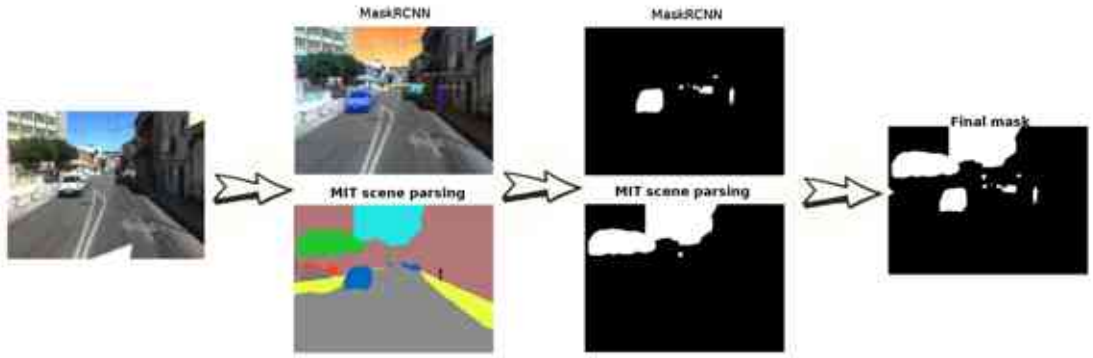

Figure 3.3: Obtaining masks using MaskRCNN [4] and MIT Scene Parsing Benchmark [5]

.

3.2.2 COLMAP

COLMAP is a general-purpose structure-from-motion (SfM) (for reconstructing camera poses and sparse point clouds) and MVS (for generating dense depth maps) pipeline. It offers a wide range of features for the reconstruction of ordered and unordered sequence of images [36, 37].

COLMAP is usually used in with the objective of retrieving camera parameters, estimate both sparse and dense 3D reconstructions projecting them on each frame, obtaining individual depthmaps. Ralho did use this algorithm to estimate the previous information and to retrieve normal maps as well, containing a vector for each pixel, normal to the respective surface.

In our case, this algorithm allow us to work with datasets that do not contain all the required information for running PRR algorithms (Coimbra dataset,etc), such as camera calibration, depthmaps, etc.

3.3 Proposed algorithm

Our pipeline is dividing into three main stages: RANSAC, that uses depthmaps and camera intrinsic parameters in order to obtain the 3D points for each image, which will be filtered with the masks created in 3.2.1. After estimating the planar hypotheses, the second stage establish links across multiple views, fusing all the linking planes to give a single estimation for the linked planes. Lastly, we optimize the segmentation labels by defining an MRF, which leads to the standard energy minimization problem involving data and smoothness terms. In the figure below (Fig. 3.1) we present a diagram showing our pipeline system together with the inputs/outputs of each stage.

3.3.1 Generation of plane hypotheses

Similar to Gallup's PPNPR first stage [2], we used a RANSAC based method to generate planar hypotheses which, on each iteration, randomly chooses three points and fits a plane to these points. The first point is randomly selected from a uniform distribution over the image. The remaining two points are selected from normal distributions centered at the first point. Then, a plane is fit to these three points and, instead of scoring the plane by simply counting the number of points within a threshold distance to the plane, we score it according to the MLE-sac method [42]. Unlike Gallup [2] which, on the following iterations, only chooses points within M pixels from the first ones, we choose points from all over the image. During our experiments, we found that we obtain better results choosing points from all over the image, instead of applying this limitation.

Our RANSAC algorithm was set to estimate fifteen planar hypothesis for each image, where each plane is chosen according the combination with the best score. After the RANSAC returns a plane, the inliers are determined by computing the distance between every 3D point and the plane.

As previously mentioned, our input information was estimated using COLMAP, resulting in depthmaps with the presence of noise, which can be associated to these planes. For this purpose, we have implemented a post-processing that creates a mask, composed by the pixels respective to the inlier set. The new inliers are obtained by choosing the larger connected area, discarding the remaining points (refer to Fig. 3.4).

Figure 3.4: Segmentation refinement, after RANSAC returns a plane with the respective inliers. On this example we can observe that some noise was included in the inlier set, which was discarded during the refinement process.

3.3.2 Plane-Linking

Succeeding the planar hypotheses generation, this next stage is going to fuse planes across nearby views. A planar surface visible in multiple frames will produce low variations on the resulting planar equations, due to small variations on depthmaps. This stage is used to link planes belonging to the same planar surface and to improve the accuracy of plane parameters. First, planes are linked in the current frame (I_i) where, a planar hypothesis π_i is compared with every other planes. For every plane π_j in the same view, if 90% of the inliers belonging to π_i falls within a threshold camera-to-point distance Δ_c of π_j, then the planes are linked. The process is repeated for every plane (π_i) obtained with RANSAC, with the exception of the ones linked in previous iterations. A new plane is fit with the combined points, belonging to all the linked planes. Then, the algorithm creates a micro-database, where linked planes across views are saved. It serves to insure that linked planes across views, have the exact same plane hypothesis.

Following a similar process, in the next iteration of the algorithm, planes are linked in the current image (I_j). Then the resulting planes are going to be compared with the micro-database and the linked planes are saved in the database. This stage was set to link planes in groups of a hundred frames, improving the efficiency of the algorithm, since time-consumption grows exponentially according to the number of images. This process is exemplified in the figure bellow (Fig. 3.5).

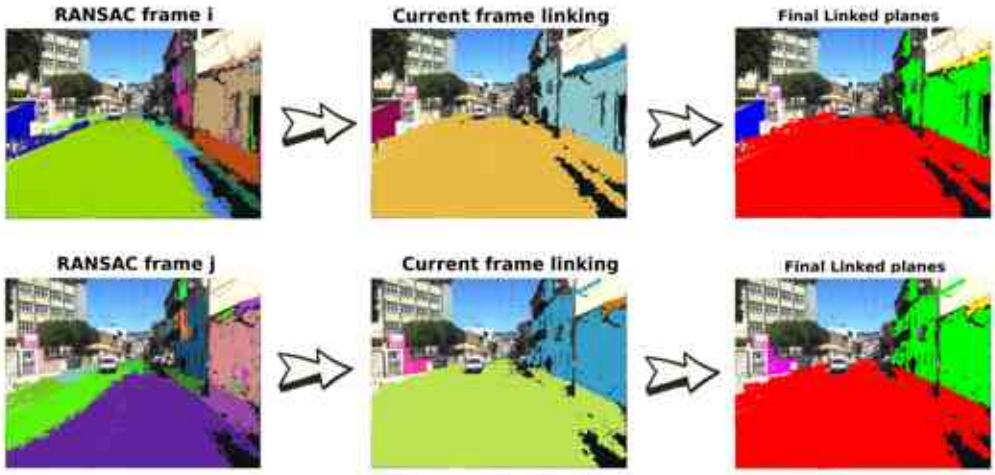

Figure 3.5: Results of the various steps taken in Plane-Linking: (column 1) Segmentation results of RANSAC stage, (column 2), segmentation after planes are linked in the current frame and (column 3) final results after planes are linked acrros nearby views. In the third column, planes having the same colour in both images, means that the planes are linked.

3.3.3 Global optimization

Once the plane hypotheses are linked, the next step is to optimize the labeling of each image. In contrast to the last step, this algorithm solves each image independently, but since this step depends on a 'micro-database' coming from the plane-linking stage, it will only process a hundred images, after each plane-linking stage is executed.

As previously mentioned, for each image we define an MRF, optimizing multi-label energies via graph-cuts [43]. Our goal is to optimize the labeling, minimizing the energy functional

$$E(L) = \sum_{p \in I} E_{data}(L(p)) + \sum_{p,q \in N} \lambda_{smooth} E_{smooth}(L(p), L(q)) \tag{3.1}$$

where I is the set of pixels in the image, N is the standard neighborhood (up, down, right and left), and L is the labeling, identifying the label for every pixel in the image: $\mathrm{L} \rightarrow \{\pi_1, ..., \pi_n, \pi_\infty, \text{non-plane}, \text{discard}\}$.

For E_{data} and E_{smooth} we have used the same functions defined in Gallup's PPNPR [2]. E_{data} is defined as

$$E_{data}(l) = \begin{cases} min(p(l), p_{max}) & \text{if } l \in \{\pi_1, ..., \pi_\infty\} \\ min(p(l), p_{max}) + p_{bias} & \text{if } l = \text{non-plane} \\ \alpha p_{max} & \text{if } l = \text{discard} \end{cases} \tag{3.2}$$

where p is a photoconsistency measured between pixels in nearby views being related by the assigned plane, or by the original depthmap. For photoconsistency we decided to test two algorithms: normalized cross-correlation (NCC) and the Birchfield-Tomasi dissimilarity (BT) [44]. During our experiments we found that NCC works better for scenes containing both small and larger planar surfaces. In the other hand, BT generalises better for most outdoor scenes with bigger planes. However, BT is ten times faster than NCC and, since bigger planar surfaces are abundant in outdoor scenes, we decided to use the BT approach for photoconsistency measuring.

E_{smooth} function is defined as

$$E_{smooth}(l_p, l_q) = g. \begin{cases} 0 & \text{if } l_p = l_q \\ d_{max} & \text{if } l_p \text{ or } l_q \in \{\pi_\infty, \text{discard}\} \\ d' & \text{otherwise} \end{cases} \tag{3.3}$$

$$d' = min(d, d_{max}) + d_{min} \tag{3.4}$$

$$g = \frac{1}{\gamma \parallel \partial I / \partial u \parallel^2 + 1} \tag{3.5}$$

where g is the gradient magnitude between two neighbors and d is the distance between 3D points according to their labels.

The resultant labeling is processed by a refining algorithm, which consists in four main steps: (1) the results are filtered by the same masks described in Section 3.2.1. (2) Next, we use the planar equations obtained in the previous stage (plane-linking), to discard pixels which the respective 3D points are too far from the plane. On this step we use

two different distance thresholds: (i) the first one is set to determine the inliers used to fit the new plane. (ii) Then, we use a bigger threshold to define the segmentation label, including an higher amount of pixels, which are still near the plane. This way, we are able to include pixels which seem to belong to the plane but, due to the estimated depthmaps and camera intrinsic parameters, are to far away to meet the first threshold requirement. In this step, pixels with a certain planar label that have no corresponding depth aren't discarded. This way, it is possible to segment low textured surfaces that were not reconstructed by COLMAP (refer to figures 1.1 and 1.2). (3) Then the algorithm discard small labeling areas, which have an high likelihood to consist in noise, present on the depthmap, or to belong to objects, e.g. vegetation, etc., that were not detected by the CNNs used in Section 3.2.1. (4) In this next step the algorithm fills small 'holes' in the different labels. Planar equations are defined by fitting a plane to the inlier set obtain in step (2). These four steps can be observed in the Figure bellow (Fig. 3.6).

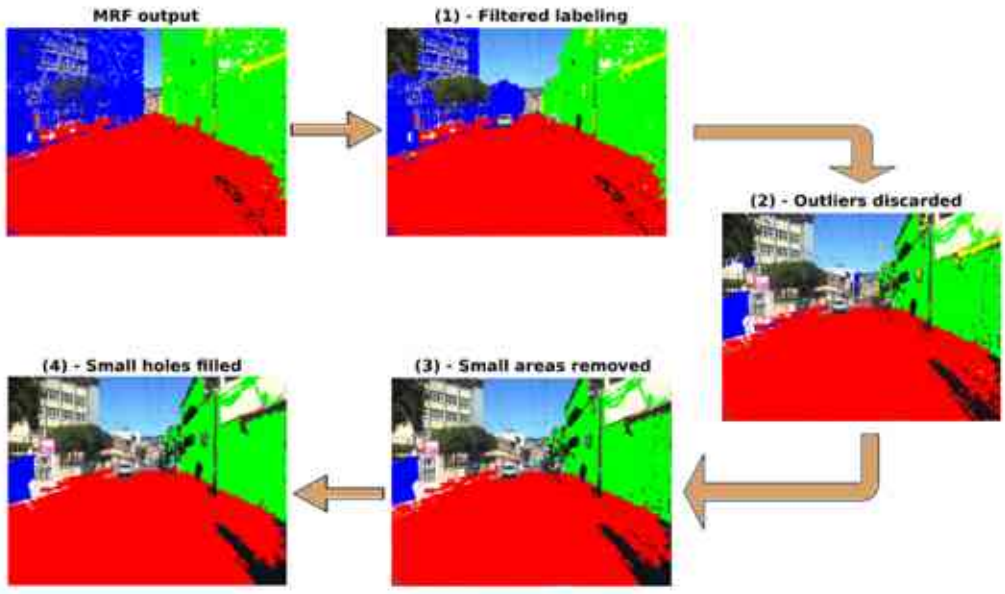

Figure 3.6: Global optimization - steps taken in the refinement process: (1) We use the masks obtained in Section 3.2.1 to filter our MRF output, then (2) we discard points that are too far from the planes, respective to each label. (3) Small areas are discarded and, finally, (4) the refinement process fills small holes, inside each label.

3.3.4 Final processing

After PPR data is generated, and once every input information was estimated using COLMAP (refer to Section 3.2.2), we had to scale both planar parameters and depthmaps to represent real values. For this purpose, we have estimated the camera height to be 1888mm [40, 41, 45] and calculated the mean distance value from the ground plane to the camera for each of the four subsets. At last, we have used these values to scale our data.

3.4 Qualitative and quantitative results

To evaluate the outcome of our data generator we have used two different metrics for geometric and plane reconstruction accuracy. For this purpose, we propose the well-known intersection over union (IOU) metric for geometric accuracy, where we assume Ralho's dataset [1] as the ground-truth. In our case, the IOU metric is calculated as the percentage of pixels that our resulting labels have, intersected with the number of pixels

of the respective Ralho's label, over the union of both labels (refer to eq. 3.6).

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{3.6}$$

On each frame, for each planar hypothesis generated with our pipeline, we search for the most likely label to be the respective plane in Ralho's segmentation. Then, we calculate the IOU value for each of our labels. At last, the final IOU value for each frame is obtained by doing the average of every IOU values computed for each planar segmentation.

For plane detection accuracy we have created a script that compiles the resultant masks, showing an interface that requires the user to manually select pairs of planes that seems to be orthogonal, e.g. walls-ground, building corners, etc. Then, for each of these pairs, the algorithm calculates the angle between both planes, and returns the difference between ninety degrees and calculate angle. For this metric we have identified orthogonal planes in 45 frames per subset, in average. Figure 3.7 describes the selection of planes that are nearly orthogonal, used for the evaluation of the detected planes accuracy.

To evaluate the generated data, we have divided the quantitative results into two groups: (1) a 'manual' subset, where we compare our data with a ground-truth, composed by the manually labeled frames. (2) Then, 'both' group, contains evaluation values using both frames that where manually labeled and the ones that where obtained using Ralho's propagation method [1].

Figure 3.7: Example of a PPR segmentation frame, generated with our algorithm, where we have selected the following combination of planes, for an error measuring: green-yellow, green-red, green-blue, yellow-red, yellow-blue.

3.4.1 Qualitative results

Figures 3.8 and 3.9 demonstrate a compilation of results generated with our fully-automatic pipeline. In Figure 3.8 we can observe some PPR segmentation with IOU values near the average (refer to Table 3.1). Figure 3.9 contain results near the maximum and minimum IOU values, where: in the first two rows, the images have high IOU values, and the last two have low IOU values.

Observing the figures bellow, we can conclude that most of the labels are well segmented, having in consideration that our pipeline is fully-automatic, in contrast to Ralho's algorithm [1]. We can also observe that our algorithm avoids the inclusion of some building sub-parts that do not belong to the planar surface, e.g., entries, balconies, etc. Additionally, we obtain better results in places with a lot of objects, like in the Sé Velha subset (refer to last row in Fig. 3.8). Our pipeline generates a less number of planes per frame, since

the depthmaps tend to be less accurate when there is a longer distance to the supposed planes, which are detected by Ralho's algorithm, due to the manual labeling. Another negative point is that our algorithm, sometimes shows to be sensitive to shadows and/or surfaces with a lot of luminosity. We can observe that sometimes there are some labels that include image pieces that has nothing to do with the labeled plane. This fact happens due to the pixels without corresponding depth being associated to planar labels, which is a negative point of (2) in Section 3.3.3. However, this step allows the algorithm to segment really low textured surfaces. At last, we can observe that Ralho's pipeline has a poor performance in a few propagations, which affects the IOU metric (refer to Section 3.4).

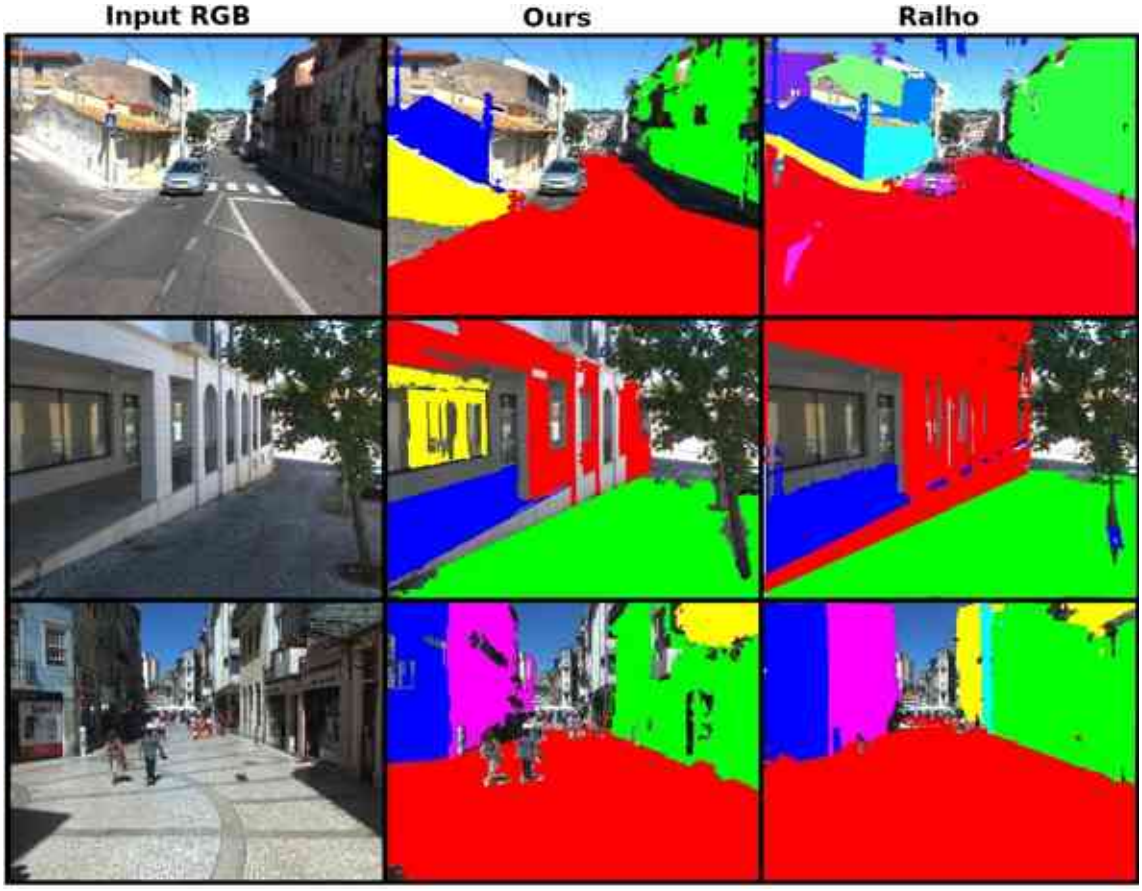

Figure 3.8: Example of PPR segmentations generated with our pipeline with IOU values near the average (refer to Table 3.1).

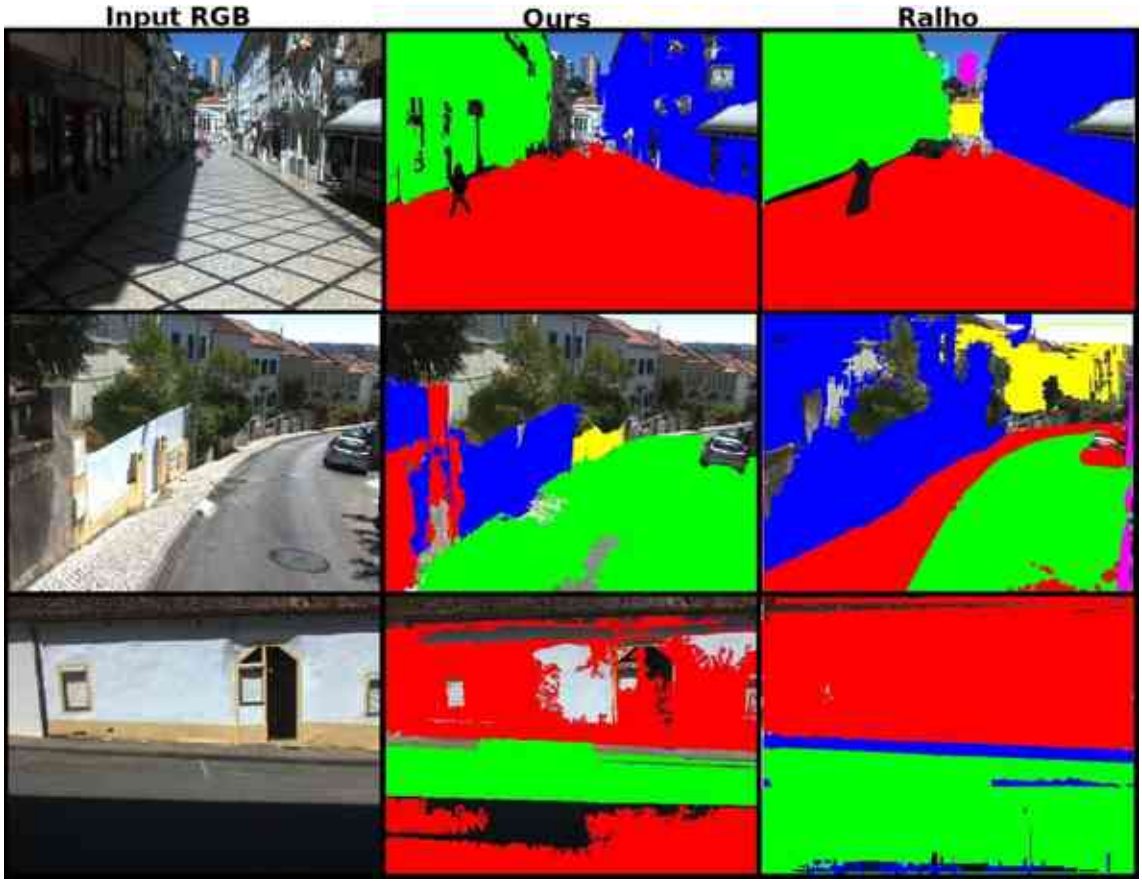

Figure 3.9: Example of PPR segmentations generated with our pipeline with good and bad IOU values (refer to Table 3.1. The first two rows contain accurate results and the last two rows contain outlier estimations: (3rd row) due to a bad segmentation, obtained with Ralho's propagated and (4rd row) due to a bad segmentation, obtained with our pipeline. In this case, our pipeline had trouble dealing with the shadow crossing the road and with a low textured wall.

Both evaluation results can be observed in the tables bellow (Tables 3.1 and 3.2), comparing the values in two sets of images: (1) manual labeled frames and (2) propagated frames in Ralho's dataset [1].

	Manual	Both
mean (%)	68.85	67.94
max (%)	94.63	95.77
min (%)	19.80	6.16
median(%)	72.60	71.08
Ratio - Ours (%)	60.21	61.70
Ratio - Ralho (%)	79.02	77.60

Table 3.1: Results of the evaluation metric, IOU, for planar segmentation accuracy of the generated data using our automatic pipeline. With this metric, we have evaluated our segmentation accuracy comparing with two groups of Ralho's results [1]: (left) Manual labeling frames - ground truth (GT) and (right) with the whole dataset (both manual and propagated frames). The 'Ratio' refers to the % of the image that is labeled.

	Ours				Ralho			
	mean (º)	max (º)	min (º)	std (º)	mean (º)	max (º)	min (º)	std (º)
Manual	**1.676**	7.028	1.378	1.296	2.804	12.494	0.028	3.427
Both	1.852	25.992	**0.003**	2.672	3.431	82.573	0.005	9.094

Table 3.2: Orthogonal planes metric, for plane detection accuracy of the generated data using our automatic pipeline. With this metric we have evaluated planes accuracy, comparing our results with Ralho's planes [1]. Similar to Table 3.1, the obtained results were compared with (1) Ralho's [1] labeled frames - Ground Truth and (2) the whole dataset.

With the IOU table we validate every topic described in Section 3.4.1: we have obtained reasonable IOU values (∼68% in average) and the total image segmentation were lower due to the poor accuracy of depthmaps in a longer distance from the camera. Additionally we can observe that Ralho's propagation was well performed, since IOU values are similar for both manual and propagated evaluation, when compared with our fully-automatic pipeline. For the orthogonal planes, we can observe that in average, our accuracy is better than Ralho's pipeline [1] since some of Ralho's planes seem to have a really low accuracy (refer to the maximum angle column in Ralho's values - Table 3.2).

Appendix I and II, respectively, Table I and II, contain more detailed information about the evaluation of our generated data, comparing the four different subsets of the Coimbra Dataset.

Chapter 4

Training SI-PPR network

At this stage, as mentioned in Section 1.2, we re-trained a DL network for SI-PPR. The selected state-of-art algorithm was the PlaneRCNN [3], described in Section 2.2.1.

As briefly introduced in Section 1.2, PlaneRCNN [3] uses an updated version of PlaneNet [12] benchmark, fitting planes to a consolidated mesh, that represents the same scene observed in every frame belonging to the actual subset. Therefore, PlaneRCNN [3] requires the following inputs: camera information (camera parameters, image size, etc.), a file containing every planar equation estimated for the actual scene in a World/common coordinate system and plane segmentation for each image. This information will be pre-processed, projecting every plane seen by the actual camera to the respective coordinate system, and associating each label/segmentation to the correspondent planes. However, our benchmark estimates planar equations for each image in the respective frame coordinate system. This way, we had to modify PlaneRCNN [3] pre-processing script in order to adapt it to our data. The remaining required inputs are RGB images and the respective depthmaps.

4.1 Training and test datasets

For the training dataset we have defined three of the four subsets present in the Coimbra dataset. The fourth subset (Santa Clara - front view) was used as a test dataset, which images were never seen from the neural network during the training phase.

In Section 3.3.4 we have created depthmaps in 'png' format, which supports a maximum value of 16 bits per pixel, corresponding to a maximum depth of 65536mm. However, PlaneRCNN has a maximum depth threshold along the whole code of 10 meters, which we have tried to change, unsuccessfully. Therefore, we have changed our data (depthmaps and planar equations) in a scale of 1:10.

In this chapter we compared the network outcome on four different checkpoints. These checkpoints are the actual weights of the neural network, which start point for every re-train we have performed are the original PlaneRCNN weights [3]. The four checkpoints that we've tested are:

- The original PlaneRCNN weights, after the network was trained with 100k indoor frames of ScanNet dataset [26].
- After re-training the network with every Ralho's PPR data (∼1300 frames).
- After re-training the network with PPR data generated with our algorithm (4487 frames).

- After re-training the network with our PPR data with the respective frames that Ralho has generated. This way, we obtain a fairest evaluation, since the training data dimensions are the same.

Additionally, we evaluated the network performance by testing a different dataset completely different than the training datasets, the Kitti dataset [46]. Ralho also generated some PPR data using this dataset, which depthmaps are sparse, obtained with a velodyne and the camera information (intrinsic and extrinsic parameters) is well known, in contrast with our data, which was estimated using COLMAP (refer to Section 3.2.2).

4.2 Qualitative results

Figure 4.1 demonstrates the results obtained by testing the neural network in the four different checkpoints (refer to Section 4.1) with our testing dataset (Santa Clara - front view), each checkpoint is identified above the respective column. The figure is divided in three main groups, where each one contains two examples of planar segmentation with good and bad IOU values (refer to Section 4.3) for three different checkpoints, respectively: (1) after re-training using all our PPR data, (2) after re-training with our PPR data, but with the same frames than Ralho's dataset and (3) after re-training with Ralho's PPR data.

With these results we can observe that after re-training with Ralho's dataset, the network is capable to detect more planes, which was not possible to estimated with our generator pipeline (refer to section 3.4. However, when re-training with the whole dataset generated with our algorithm, the network starts to label buildings while discarding some parts, e.g. balconies, etc. Another positive point is that the sky is almost never labeled as a planar surface on the same checkpoint. In contrast, the sky is often labeled in the other three checkpoints, since the network was previously trained with indoor datasets, and the re-training has been made with only a few images (about 1k).

Figure 4.1: Qualitative results for the network output (Santa Clara - front-view testset). The figure is divided into three main groups: first and second row contain good and bad results, respectively, obtained with the neural network, after training with all our 4487 frames. Similar to the first two rows, the remaining rows contain good and bad results obtained with the network after training with 1300 PPR segmentation frames from our dataset and Ralho's [1] data, respectively.

4.3 Quantitative results

To evaluate the network performance, we have used the same metrics presented in Section 3.4: IOU and the angle between planes that seem to be orthogonal. The Santa Clara - front-view subset was used as a test dataset, where the PPR data generated with Ralho's pipeline is considered to be the ground-truth (refer to Section 4.1).

We have tested the network between each epoch[1], measuring the IOU values (maximum, average and minimum) with the point of finding the optimal number of epochs to obtain better results. In the figure 4.2, we can observe that the results start to diverge after training for five epochs. This effect occurs since we have used the original PlaneRCNN [3] training configuration, which is prepared for a training data with a specific size, containing indoor scenes. We have tried to re-train the network using different configurations, in order to obtain results that converge to a better IOU value. However, we have been unsuccessful and, by observing the Figure 4.2 we decided to train the network with the original configurations for five epochs.

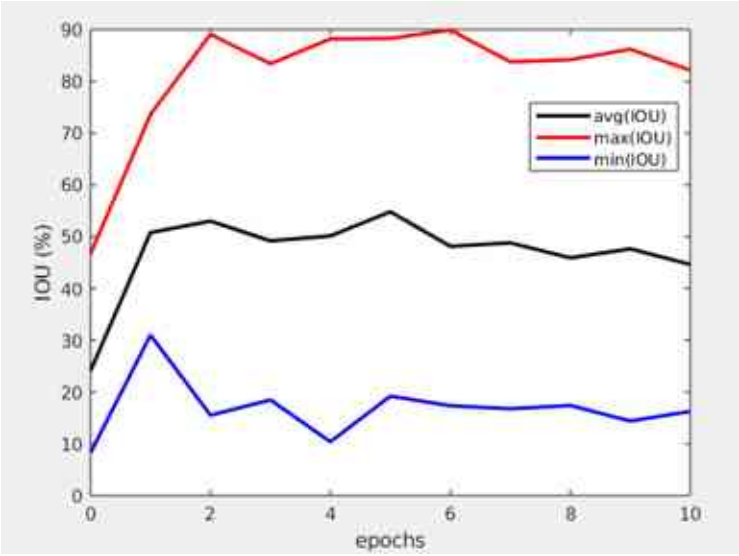

Figure 4.2: IOU values for the testing data, among the ten epochs trained with our whole dataset. In this graphic, we can observe that the results obtained along the training phase do not converge. For this reason, we have determined the optimal number of training epochs as five.

The results for both metrics are presented in the tables bellow (Table 4.1 and 4.2). With these results we can observe a similar IOU values for both checkpoints (2) and (3), where the network was re-trained with the same number of images, with Ralho's data and our data, respectively. Even though the results observed in Figure 4.1 show that the algorithm does not associate some building parts to the facade plane, e.g. doors, windows, etc., we still obtain better IOU values in the checkpoint (4), where we re-trained the network with ∼4500 frames. Therefore, these validates the fact that these kind of DL-based algorithms require a lot of training data.

In Table 4.2 we can observe that with the original PlaneRCNN we obtain a better average error between a ninety degree angle and the real angles. However, as observed in Figure 4.1, many times big planar structures are segmented into various pieces/labels. Another fact, is that most of the times the network segment objects, e.g. cars, as a planar surface. Comparing the other three checkpoints, after the re-training, we can observe that after re-training with ∼4500 frames, we obtain a worse average error value, however the

[1]One epoch is when the whole training dataset was processed by the network.

maximum error of checkpoint (4) is less than half the value of the other two checkpoints, where we observe the existence of planes that should be orthogonal, actually are parallel.

		Original weights	1300 frames Ralho	1300 frames Ours	4487 frames Ours
Manual	mean (%)	23.83	46.94	47.91	54.18
	max (%)	42.02	91.42	75.68	79.34
	min (%)	12.49	22.23	21.50	21.71
	median (%)	22.92	40.25	48.88	56.88
	Ratio (%)	48.57	55.49	51.81	62.27
Both	mean (%)	24.09	43.42	45.94	52.50
	max (%)	46.67	92.97	88.78	83.81
	min (%)	8.27	16.74	15.35	18.36
	median (%)	23.06	42.06	46.18	52.54
	Ratio (%)	48.58	56.14	50.99	63.47

Table 4.1: Results of the evaluation metric, IOU, for planar segmentation accuracy of PlaneRCNN outcome (Santa Clara - front-view testset). With this metric, we have evaluated our segmentation accuracy comparing with two groups of Ralho's results [1]: (left) Manual labeling frames - ground truth (GT) and (right) with the whole dataset (both manual and propagated frames). The 'Ratio' refers to the % of the image that is labeled. Refer to Table 3.1 for ground-truth ratio

avg (º)	max (º)	min (º)	std (º)
Original PlaneRCNN weights			
8.439	27.873	0.825	6.836
Re-trained with Ralho's data (∼1300 frames)			
17.155	88.891	0.920	18.479
Re-trained with out data (∼1300 frames)			
13.816	81.8159	0.1137	18.666
Re-trained with our data (4487 frames)			
16.867	41.7959	0.5051	9.7824

Table 4.2: Orthogonal planes metric, testing the neural network using for that purpose the Santa Clara - front-view testset. With this metric we have evaluated planes accuracy, comparing our results with Ralho's planes [1]. Similar to Table 3.1, the obtained results were compared with (1) Ralho's [1] labeled frames - Ground Truth and (2) the whole dataset.

The maximum error obtained with the orthogonal planes metric for both checkpoints, where we have trained the neural network with 1300 frames from Ralho's [1] dataset and our data, was about ninety degrees, meaning that the detected planes are almost parallel, when they should be orthogonal. We have investigated the issue, and found that, sometimes, the neural network estimates a wrong plane for surfaces with low texture. In Figure 4.3 we can observe and example of a wrong detected plane.

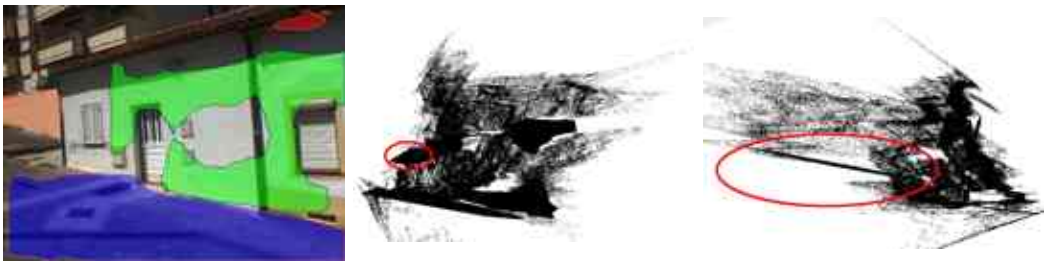

Figure 4.3: Example of a completely wrong plane equation, detected with PlaneRCNN [3], after training the neural network with $\sim$ 1300 frames, from our generated dataset. In the first image, we present the estimated segmentation. In the second and third images, we can observe the erroneous plane in a 3D reconstruction from two different views, respectively, front-view and a left side-view. The highlighted plane that we can observe in the 3D reconstruction, correspond to the orange label, in the first image.

4.4 Testing SI-PPR generation capabilities

4.4.1 Kitti dataset - qualitative and quantitative evaluation

As previously described in Section 4.1, we have tested the network with a different dataset, the Kitti Dataset [46]. The results will be evaluated and compared with the results from Ralho's pipeline, which we have considered to be the ground truth. In Figure 4.4, we present the qualitative results where, similar to the Figure 4.1, we divided the figure into three main groups, presenting good and bad results for three different checkpoints of the network (refer to Section 4.2). By observing Table 4.3 and 4.4, we can conclude that we obtained similar accuracy after re-training the neural network with our data (4487 frames) and Ralho's data. However, the network has a much better performance (almost twice) for planar segmentation (refer to Table 4.3).

Figure 4.4: Qualitative results for the network output using the Kitti dataset as testset. The figure is divided into three main groups: first and second row contain good and bad results, respectively, obtained with the neural network, after training with all our 4487 frames. Similar to the first two rows, the remaining rows contain good and bad results obtained with the network after training with 1300 PPR segmentation frames from our dataset and Ralho's [1] data, respectively.

avg (%)	max (%)	min (%)	std (%)	% Seg. Ours	% Seg. Ralho
Checkpoint 1 - Original PlaneRCNN weights					
18.46	34.64	6.82	7.04	50.90	57.30
Checkpoint 2 - Re-trained with Ralho's data (∼1300 frames)					
32.21	56.35	13.37	11.25	41.97	57.30
Checkpoint 3 - Re-trained with out data (∼1300 frames)					
25.43	47.68	7.26	10.17	37.11	57.30
Checkpoint 4 - Re-trained with our data (4487 frames)					
40.24	70.64	20.31	11.98	67.77	57.30

Table 4.3: Results of the evaluation metric, IOU, for planar segmentation accuracy of PlaneRCNN outcome (Kitti dataset). In this table we can observe that the neural network generalises well, when tested with a completely different dataset, that has never been seen during the training phase.

avg (º)	max (º)	min (º)	std (º)
Checkpoint 1 - Original PlaneRCNN weights			
8.439	27.873	0.825	6.836
Checkpoint 2 - Re-trained with Ralho's data (∼1300 frames)			
10.538	49.37	0.047	12.736
Checkpoint 3 - Re-trained with out data (∼1300 frames)			
13.200	25.015	0.725	7.402
Checkpoint 4 - Re-trained with our data (4487 frames)			
13.230	37.509	0.540	10.557

Table 4.4: Orthogonal planes metric, testing the neural network using for that purpose the Kitti dataset.

4.4.2 Other datasets: ETH and ScanNet

In this section we present some results obtained testing other datasets: ScanNet, the training dataset used by the PlaneRCNN [3] authors and the ETH dataset. However, the scene from ScanNet that we have tested was not used to train the neural network by the authors. Observing the figure 4.5, we can conclude that, since we are re-training the neural network for outdoor processing, the indoor results tend to be worse than the ones obtained with the original PlaneRCNN [3] weights. Though, the results for ETH outdoor scenes, show that the algorithm keeps to return well segmentated labels for outdoor scenes (refer to Fig. 4.6, fourth column).

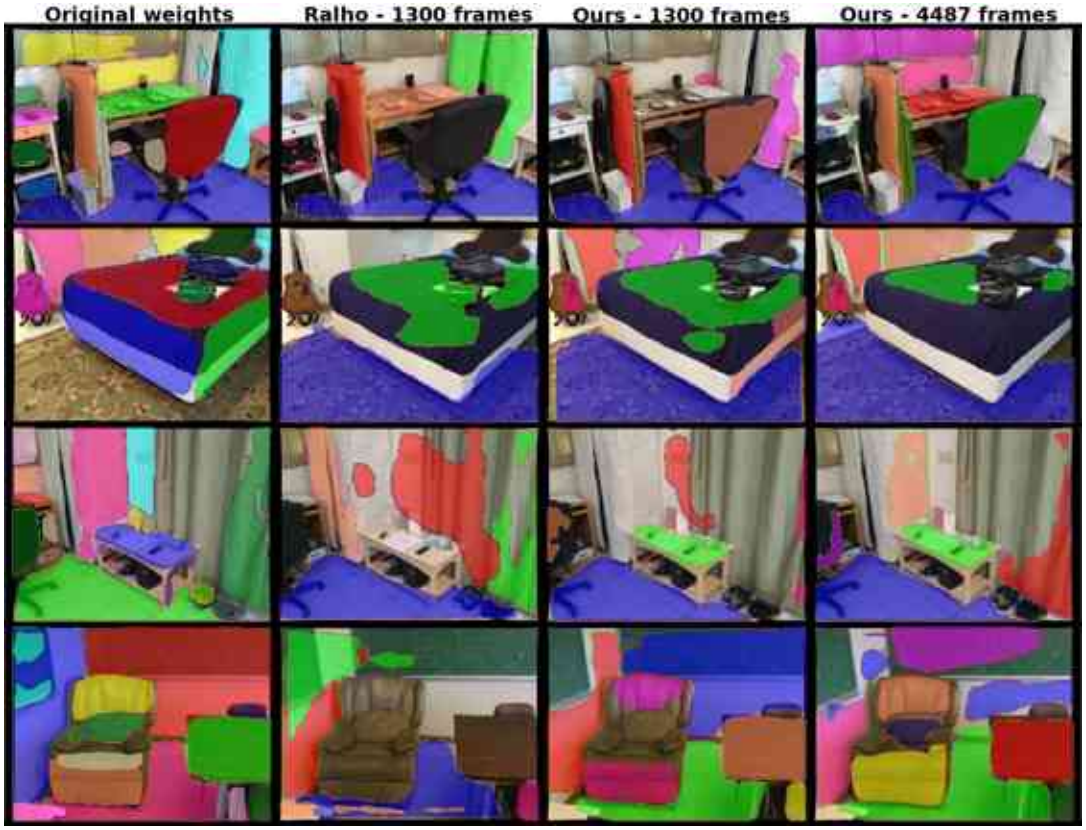

Figure 4.5: Testing PlaneRCNN with the ScanNet dataset.

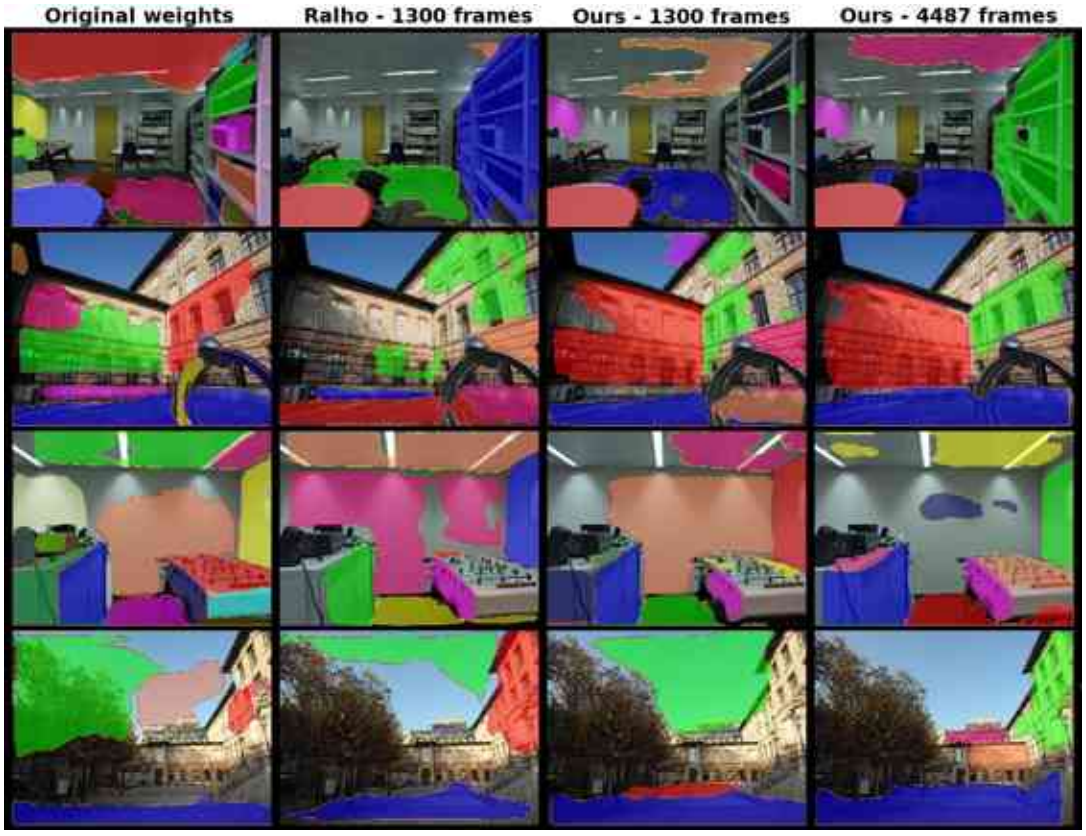

Figure 4.6: Testing PlaneRCNN with the ETH dataset.

Chapter 5

Conclusion and future work

We presented a new pipeline for PPR data generation that does not need any human intervention, enabling the automatic generation of large-scale training datasets for SI-PPR DL networks. By evaluating the results obtained, we have shown that it is possible to obtain similar accuracy performance as approaches that require time-consuming manual labeling. The only downside, when comparing to Ralho's results [1], is that our algorithm detects a less number of planes, since most of the times, it is not capable to distinguish similar planes, e.g. sidewalks from roads, etc., and cannot detect planes that are too far from the camera, which contain a lot of noise. However, our pipeline works better when segmenting buildings, by not including entrances, balconies, etc, that do not belong to the planar surface (refer to Fig. 3.8).

After re-training the neural network with our generated data, we have demonstrated that the network is capable to predict PPR segmentations for locations never observed during the training phase, and generalizes very well for other datasets.

During this thesis we were able to generate ∼5200 frames and, since DL-based approaches require large amount of training data, an interesting future direction would be to generate a large-scale dataset using our pipeline and train PlaneRCNN [3] or other DL-based approaches from scratch, validating the purpose of this thesis. Another interesting idea would be the use of MIT Scene Parsing Benchmark [5] as a starting point (refer to Fig. 3.3 - MIT scene parsing) and use a modified version of our algorithm to detect planes on each parsed area, e.g. road, buildings, etc. This way, we enforce the algorithm to find more planes, even in surfaces far from the camera, which are not detected with our current pipeline, due to high presence of noise (refer to Section 3.4.1). Additionally, this technique would allow the algorithm to distinguish similar planes, e.g. sidewalks from the road which, most of the times, our actual pipeline have some difficulties to differentiate. This idea was inspired by the work of Ralho [1], that uses manual labeling to identify different areas of the image containing planar surfaces. However, just like our pipeline, the purpose of this new concept would be to have no human intervention, generating PPR data in a fully-automatic manner.

Bibliography

[1] J. Ralho, "Learning Single-View Plane Prediction from autonomous driving datasets," Master's thesis, Universidade de Coimbra, Portugal, 2019.

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

[3] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, "Planercnn: 3d plane detection and reconstruction from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4450–4459, 2019.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[5] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[6] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 824–840, 2008.

[7] H. Jin, S. Soatto, and A. J. Yezzi, "Multi-view stereo reconstruction of dense shape and complex appearance," *International Journal of Computer Vision*, vol. 63, no. 3, pp. 175–189, 2005.

[8] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European conference on computer vision*, pp. 746–760, Springer, 2012.

[9] Z. Li and N. Snavely, "Megadepth: Learning single-view depth prediction from internet photos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2041–2050, 2018.

[10] T. Koch, L. Liebel, F. Fraundorfer, and M. Korner, "Evaluation of cnn-based single-image depth estimation methods," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 0–0, 2018.

[11] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, pp. 2366–2374, 2014.

[12] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, "Planenet: Piece-wise planar reconstruction from a single rgb image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2579–2588, 2018.

[13] F. Yang and Z. Zhou, "Recovering 3d planes from a single image via convolutional neural networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 85–100, 2018.

[14] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, vol. 1, pp. 519–528, IEEE, 2006.

[15] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[16] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An invitation to 3-d vision: from images to geometric models*, vol. 26. Springer Science & Business Media, 2012.

[17] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International journal of computer vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[18] S. N. Sinha, D. Steedly, and R. Szeliski, "Piecewise planar stereo for image-based rendering," in *2009 IEEE 12th International Conference on Computer Vision*, pp. 1881–1888, Sep. 2009.

[19] C. Raposo, M. Antunes, and J. P. Barreto, "Piecewise-planar stereoscan: structure and motion from plane primitives," in *European Conference on Computer Vision*, pp. 48–63, Springer, 2014.

[20] M. Antunes, J. P. Barreto, and U. Nunes, "Piecewise-planar reconstruction using two views," *Image and Vision Computing*, vol. 46, pp. 47–63, 2016.

[21] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Manhattan-world stereo," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1422–1429, IEEE, 2009.

[22] M. Antunes and J. P. Barreto, "Semi-dense piecewise planar stereo reconstruction using symstereo and pearl," in *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, IEEE, 2012.

[23] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 Fourth international conference on 3D vision (3DV)*, pp. 239–248, IEEE, 2016.

[24] T. Schops, J. L. Schonberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, "A multi-view stereo benchmark with high-resolution images and multi-camera videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[25] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–13, 2017.

[26] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5828–5839, 2017.

[27] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. USA: Cambridge University Press, 2 ed., 2003.

[28] H. Fan, H. Su, and L. Guibas, "A point set generation network for 3d object reconstruction from a single image," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[29] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," *Lecture Notes in Computer Science*, p. 628–644, 2016.

[30] M. Antunes and J. P. Barreto, "Semi-dense piecewise planar stereo reconstruction using symstereo and pearl," *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pp. 230–237, 2012.

[31] M. Antunes and J. P. Barreto, "Symstereo: Stereo matching using induced symmetry," *International Journal of Computer Vision*, vol. 109, pp. 187–208, 09 2014.

[32] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[33] G. Ros, L. Sellart, J. Materzynska, D. Vázquez, and A. López, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," pp. 3234–3243, 06 2016.

[34] R. Toldo and A. Fusiello, "Robust multiple structures estimation with j-linkage," vol. 5302, pp. 537–547, 10 2008.

[35] G. Hu, X. Peng, Y. Yang, T. M. Hospedales, and J. Verbeek, "Frankenstein: Learning deep face representations using small data," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 293–303, 2017.

[36] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[37] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

[38] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[39] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015.

[40] C. Raposo, M. Antunes, and J. P. Barreto, "Piecewise-planar stereoscan:structure and motion from plane primitives," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), vol. 8690 of *Lecture Notes in Computer Science*, pp. 48–63, Springer International Publishing, 2014.

[41] "Urbanscan: 3d modeling of urban scenes." `http://montecristo.co.it.pt/PPR_Rec/`. Accessed: 2020-07-03.

[42] P. Torr and A. Zisserman, "Mlesac: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138 – 156, 2000.

[43] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[44] S. T. Birchfield and C. Tomasi, "A pixel dissimilarity measure that is insensitive to image sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, pp. 401–406, 1998.

[45] "Clio 4 specs." `https://www.ultimatespecs.com/pt/car-specs/Renault/73847/Renault-Clio-4-12-16v-75-Limited.html`. Accessed: 2020-07-03.

[46] "Kitti dataset." `http://www.cvlibs.net/datasets/kitti/raw_data.php`. Accessed: 26-01-2020.

Appendix

Appendix I - Detailed results of the IOU metric for the generated data

	avg (%)	max (%)	min (%)	std (%)	% Seg. Ours	% Seg. Ralho
	Sequence 1 - Santa Clara - front view					
Propagated	69.28	94.32	16.79	16.93	53.95	68.13
Manual	68.84	92.19	23.14	18.05	53.01	69.08
Both	69.24	94.32	16.79	16.99	53.87	58.22
	Sequence 2 - Santa Clara - side view					
Propagated	69.44	95.77	6.83	16.41	70.34	85.72
Manual	69.99	90.44	19.80	16.41	66.71	86.83
Both	69.50	95.77	6.83	16.40	69.99	85.83
	Sequence 3 - Seminário					
Propagated	63.32	91.20	6.16	16.45	59.41	79.34
Manual	66.52	83.07	35.41	15.25	57.77	80.19
Both	63.61	91.20	6.16	16.33	59.26	79.41
	Sequence 4 - Sé Velha					
Propagated	64.74	95.01	24.74	14.52	63.73	76.61
Manual	66.55	94.63	40.67	15.59	63.36	79.96
Both	64.91	95.01	24.74	14.59	63.69	76.92
	Overall Results					
	67.94	95.77	6.16	16.39	61.70	77.60

Table I: Results of the evaluation metric, IOU, for planar segmentation accuracy of the generated data using our automatic pipeline.

Appendix II - Detailed results of the ortogonal planes metric for the generated data

	Ours				Ralho			
	avg (º)	max (º)	min (º)	std (º)	avg (º)	max (º)	min (º)	std (º)
	Sequence 1 - Santa Clara - front view							
Propagated	1.827	5.333	0.035	1.516	2.800	20.594	0.036	4.556
Manual	1.829	4.559	0.122	1.296	2.056	5.163	0.028	2.074
Both	1.827	5.333	0.035	1.516	2.691	20.594	0.028	4.476
	Sequence 2 - Santa Clara - side view							
Propagated	1.882	5.305	0.018	1.289	3.920	82.573	0.005	14.099
Manual	1.702	7.028	0.024	1.447	2.8522	12.494	0.056	4.172
Both	1.871	7.028	0.018	1.257	3.725	82.573	0.005	12.7615
	Sequence 3 - Seminário							
Propagated	2.601	25.992	0.003	4.431	4.715	39.466	0.271	8.643
Manual	2.247	5.114	0.402	1.382	3.494	9.287	0.162	3.226
Both	2.575	25.992	0.003	4.391	4.782	39.466	0.162	8.349
	Sequence 4 - Sé Velha							
Propagated	0.813	3.879	0.084	0.823	2.430	16.920	0.029	3.746
Manual	0.831	2.946	0.073	0.718	2.767	11.737	0.236	3.367
Both	0.813	3.879	0.073	0.823	2.312	16.920	0.029	3.602

Table II: Orthogonal planes metric, for plane detection accuracy of the generated data using our automatic pipeline.