$2^1/_2$D Scene Reconstruction of Indoor Scenes from Single RGB-D Images

Natalia Neverova, Damien Muselet, and Alain Trémeau

Laboratoire Hubert Curien – UMR CNRS 5516, University Jean Monnet,
42000 Saint-Étienne, France
natalia.neverova@etu.univ-st-etienne.fr,
{damien.muselet,alain.tremeau}@univ-st-etienne.fr

Abstract. Using the Manhattan world assumption we propose a new method for global $2^1/_2$D geometry estimation of indoor environments from single low quality RGB-D images. This method exploits both color and depth information at the same time and allows to obtain a full representation of an indoor scene from only a single shot of the Kinect sensor. The main novelty of our proposal is that it allows estimating geometry of a whole environment from a single Kinect RGB-D image and does not rely on complex optimization methods. This method performs robustly even in the conditions of low resolution, significant depth distortion, non-linearity of depth accuracy and presence of noise.

Keywords: 3D reconstruction, RGB-D Images, Manhattan World.

1 Introduction

The idea of extracting the information about the third dimension from images and videos has always attracted enormous attention of computer scientists, since depth estimation is involved, explicitly or not, in almost any 3D computer vision problem. For example, geometry estimation of observed scenes is an essential part of robot perception, especially for self-navigating and self-localizing systems. One of the most illustrative examples in outdoor environments is the concept of autonomously driving cars. Analogously, indoor scenes reconstruction is necessary for home robot navigation and assistance systems for visually impaired people as well as for some special kinds of video surveillance systems, for example, in the context of unusual events detection. Depending on the application, available data and general requirements of the system, different methods for depth extraction can be used. In the following paragraphs we will provide a brief overview of some classical approaches.

1.1 Geometry Reconstruction without Depth Information

Talking about depth extraction from still images, we can roughly divide all methods into three groups: bottom-up methods, deriving a 3D scene model using only integration of low-level features (such as edges, colors, vanishing points,

S. Tominaga, R. Schettini, and A. Trémeau (Eds.): CCIW 2013, LNCS 7786, pp. 281–295, 2013.

textures, etc.); salient-first approaches, which model semantic object classes to start with; top-down methods, which perform scene categorization as a prior for subsequent processes.

Bottom-up approaches extract primitive features, then progressively build their knowledge of scene depth. Some researchers have attempted to learn direct mappings from low-level features to scene geometry, bypassing the "region growing" procedure. The work of Torralba and Oliva [2] has been very influential in this regard. Based on the magnitude of the global Fourier transform and the local wavelet transforms, they get a sense of average scene depth. Saxena et al. [3] also presented a method to learn depth from single outdoor images based on low-level features in a Markov Random Field (MRF) model. Delage et al. [4] derived an algorithm for reconstructing indoor scenes from a single image, whereas Barinova et al. [5] achieve the same for urban outdoor scenes; they both learn the boundaries between the ground surface and the vertical structures.

Psychological evidences suggest that accurate depth perception is largely influenced by higher-level processes such as object class recognition. Several approaches attempt to learn models of commonly occurring semantic object classes, such as sky, water, mountain, and road. Others model more complex outdoor objects such as buildings and cars, or offices such as computer screens, desks and bookshelves. Hoiem et al. [6] model classes that depend on the orientation of a physical object with relation to the physical scene, they classify image regions into sky, horizontal ground, vertical left wall and vertical right wall.

An alternate approach involves recognition of scene category, regardless of individual surfaces or objects, such as categorization of the scene into indoor vs. outdoor, city/suburb versus landscape. They usually rely on particular discriminating features (e.g. cities have more vertical edge energy than flat landscapes). The scene descriptor used by Oliva and Torralba for estimation of scene depth has already proven useful in detecting objects and estimating their size [7].

Generally, the performance of all methods dedicated to depth estimation from still images is very low in practical sense. It is necessary to improve the quality of scene reconstruction, and to do that it seems logical to use more data and, first of all, more images of the same scene taken from different positions. Depending on the number of shots and camera trajectories, we can distinguish several methods of depth reconstruction, such as depth from stereo, depth from multiple views, depth from optical flow, or, generally, depth from video. Systems based on stereo or multiple views are usually object centered, while optical flow methods are more suitable for camera-centered systems such as moving robots. Multiple view methods are generally built upon the framework explained in [8]. There are many different models that approach this issue differently, but all of them can be mapped to the framework described above. One of the popular models that is widely used and researched is proposed by Pollefeys [9].

1.2 Geometry from Range Maps

The idea of using structured light cameras as depth sensors become more and more popular nowadays due to their speed and flexibility. The Kinect sensor is

the most well-known representative of these systems mostly because of its low price, good working range and reasonable resolution.

Using depth maps is very beneficial for geometry reconstruction, since it allows to skip one of the most challenging steps involved in multiple view reconstruction, such as feature detection and matching of frames, and move directly to the point cloud processing. As a result, depth-based methods perform robustly in presence, for example, of large uniform regions which are rather typical for indoor environments.

Although the ongoing research dedicated to different applications involving the Kinect sensors is very active, the field of geometry reconstruction with this kind of devices is not well developed yet. The research community still tends to treat "geometry from depth maps" or "geometry from point clouds" problem separately from color image processing. Moreover, in most cases known methods of scene reconstruction from point clouds are limited either to outdoor man-made scenes or to reconstruction of shapes of single objects. One of the most interesting works proposed by Microsoft research [11] allows handling indoor scenes using piece-wise planar representation, although requires a large amount of data and relies on computationally expensive Markov Random Fields formulation. Another work [12] is dedicated to rough room structure estimation from single-view point clouds, but they do not go further than separate plane estimation and exploit only depth information, while using Kinect sensors allows benefiting from a combination of color and depth information.

1.3 2½D Scene Reconstruction Using Depth Sensors

To sum up this introduction and establish the link between the state-of-the-art approaches and the method that we propose, we must note that:

- In our work we do not count on single image-based methods, since they generally require high resolution and high quality data, computationally expensive and are not robust enough in the practical sense, especially when it comes to indoor environments characterized by large variety of colors, textures and object categories.
- Although there are many methods that allow reconstructing geometry using point clouds generated from multiple views, in most cases they cannot be applied directly to the processing of a point cloud given by a depth sensor in a single shot. This data is different in the sense of its scarcity (due to low depth camera resolution) and presence of occlusions (due to single point of view). Moreover, being free of errors of frame matching and dependency on scene homogeneity, it has its own failure cases (such as light sources saturating the depth sensors, low albedo surfaces, object boundaries with high curvatures). In addition, as it was mentioned previously, most of the known methods of scene reconstruction from point clouds are computationally expensive and generally either limited to large-scale outdoor man-made scenes or to reconstruction of very local scenes and separate objects.

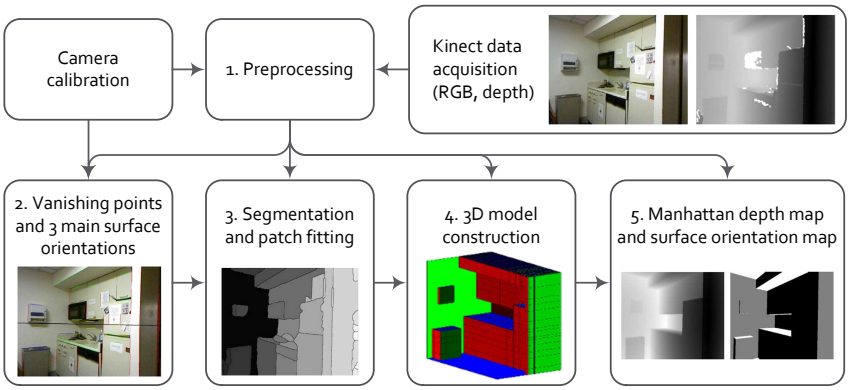

Fig. 1. The proposed workflow of the 3D modeling system

– Instead, in this paper we propose a simple and computationally efficient method for rough geometry reconstruction of the whole room using a single low resolution RGB-D image. We propose to use the famous Manhattan world assumption [1] which states that all object surfaces in the real world are aligned with one of three mutually perpendicular directions. This statement allows obtaining sufficient results for man-made environments.

The $2^1/_2$D modelling algorithm proposed here comprises several steps which are illustrated in Fig. 1. As an input, it takes RGB and depth images provided by the Kinect sensor, as well as its intrinsic parameters obtained as a result of calibration. As an output, it gives a piecewise planar Manhattan-based model of the scene (i.e. the Manhattan depth and surface orientation maps). This algorithm is based on five steps. The first step (see section 2.1) consists of data pre-processing and synchronization of RGB and D images in order to obtain a single RGB-D image. This step also exploits camera parameters in order to transform the depth image into a point cloud in the 3D space. In the second step (see section 2.2) we define the three main surface orientations of the Manhattan world. We will show that the normal vectors of three main planes are parallel to vectors coming from the camera coordinate system origin through 3 vanishing points on the image plane. The goal of this step is therefore to find these 3 vanishing points. In the third step (see section 2.3) we segment the RGB-D image and fit planar patches to each corresponding group of points in the point cloud each time choosing one of three possible orientations fitting the data in the best way. The fourth step (see section 2.4) consists in grouping similar patches and construction of a voxel-based occupancy grid of the studied scene by dividing the 3D space into blocks (with boundaries corresponding to positions of found patches) and checking which ones of them are occupied. On this step we perform final post-processing of the 3D model and fill holes produced by occlusions. In the last step (see section 2.5) we use the obtained 3D model to construct two new images: surface orientation map and Manhattan depth map.

Fig. 2. Hole filling in depth images: 1) RGB image; 2) corresponding raw depth data; 3) depth data after applying the bilateral filter proposed in [13]

2 2½D Scene Reconstruction Based on a Combination of Depth and Color Data

In the following sections we detail each step of the proposed workflow (Fig. 1).

2.1 Data Pre-processing

The main task of the data pre-processing step is to perform the following operations: (a) to project the depth map on the RGB camera point of view in order to establish correspondence between the color and depth information and obtain a single RGB-D image. This procedure is straightforward since intrinsic parameters of the two cameras and the transition matrix are known after the sensor calibration [14]. Regions on image boundaries where either depth of color information is missing (due to different fields of view of color and depth cameras) must be cropped. As a result, although each camera produces an image of 640×480 pixels, only 560×430 pixels are useful; (b) to fill holes in depth maps. Under some conditions, such as low albedo surfaces, bright light sources and specular reflections, transparent objects, etc. (e.g. see Fig. 2) the pair of IR emitter-sensor fails to provide depth information, therefore there are usually some holes in depth maps. To compensate for these negative effects, we smooth the depth map using the cross-bilateral filter of Paris [13] that exploits the edge information from the color image; (c) although this filter performs well for inner regions, it produces some negative effects on edges of depth maps. When the difference between depth values of two neighbouring pixels is significant due to object occlusions, this filter produces some pixels with intermediate depth values (see Fig. 3). These pixels will be considered in the following steps as outliers. To face this issue, we have introduced an additional refining procedure: we calculate the depth gradient distribution and filter out pixels where the gradient values are greater than a certain threshold γ_{bf}. These pixels will not be considered for further processing, (d) On this preliminary step we also need to transform the depth map to a point cloud which will be used for further processing. To do that, we exploit the classical pinhole camera model and use values of the camera parameters obtained during sensor calibration.

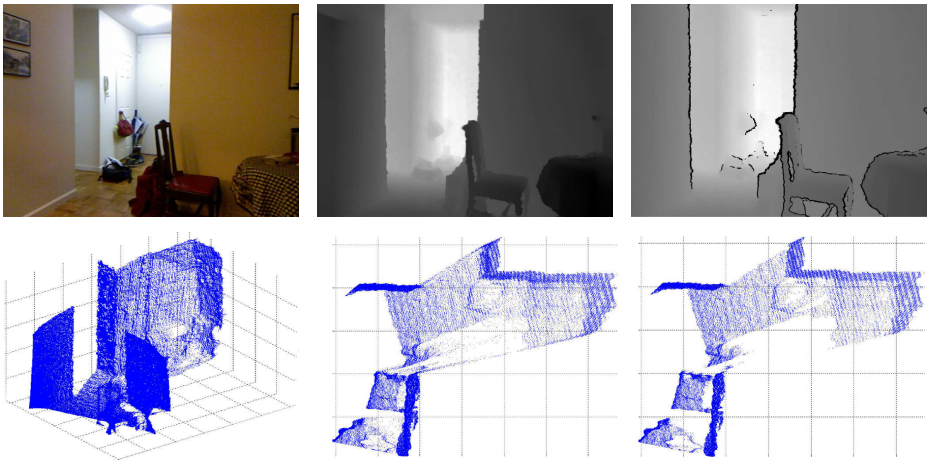

Fig. 3. Removal of erroneous pixels given by the bilateral filtering. 1st row: input color Kinect image, bilateral-filtered depth map before and after removing the pixels. 2nd row: the point cloud and two corresponding projections of the point cloud on the floor surface (view from above). We can observe that in the 1st case there are some "parasite" points laying on straight lines connecting the camera point with boundaries of occlusions (which are almost completely removed in the 2nd case).

2.2 Estimation of Main Surfaces Orientations

Estimation of three main surface orientations of the Manhattan world is the core of our framework. As it was mentioned before, on this step we rely on vanishing points detection since their positions contain all information about the world orientation. Fig. 4 shows that the vectors connecting the camera point with three main vanishing points in the image exactly correspond to the normal vectors of three main directions of the Manhattan world. There can be an infinite number of vanishing points in any image since any bunch of lines with an arbitrary orientation has its own point of convergence belonging to so-called vanishing lines. However, in our case we will consider only those three points that are formed by only "vertical" and "horizontal" lines parallel to main planes of our Manhattan world. Thus, the outline of the algorithm on this step is following: (a) edge extraction from an RGB image; (b) fitting of straight line segments to the obtained edges; (c) assigning of each line segment to one of three main directions and calculation of vanishing points and corresponding vectors normal to the three main planes, plus (d) optimization in order to improve the accuracy. **Step (a).** Edges from RGB images can be extracted using any methods, designed either for intensity or color images. All parameters must be tuned in advance in order to achieve the best performance, since the quality of images produced by Kinect color cameras is rather poor. Since in our application we are mostly interested in edges associated with scene geometry and not variations of illuminations, we use the method proposed in [15] to robustly extract

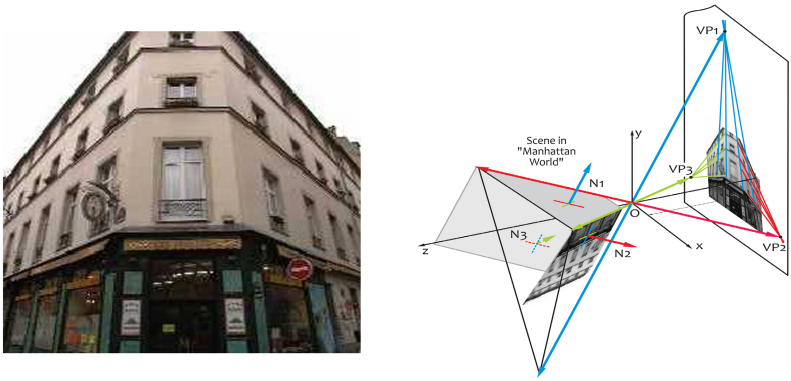

Fig. 4. Relation between vanishing points and main surface orientations. Each color of vectors corresponds to its own orientation.

photometric invariant features using color tensors. In order to reduce the influence of noise, images are filtered with a 9×9 median filter.

Step (b). As soon as edges are extracted, we replace them with straight lines (given minimum segment length, maximum deviation from straight line before a segment is broken in two, angle tolerance used then attempting to merge line segments and maximum distance between end points of line segments for segments to be eligible for linking). To do that, we use the Kovesi's method [16]. The parameters were set to: 5 for the minimum line length, 1 for the maximum deviation from the original edge pixels, 0.05 for the angular tolerance and 2 for the maximum distance between two lines needed to be connected.

Step (c). Each line segment is to be assigned to one of three main vanishing directions. As a basic method for vanishing point detection we have chosen the modification of Rother's method [17] described in [18]. Here we explicitly specify the value of the focal length and adapt the code to be able to work with an arbitrary position of the principal point of the RGB camera in the image coordinate system. The first task is to find the "vertical" vanishing point corresponding to orientation of horizontal surface in the Manhattan world. As an initial estimation, we assume that this point is located strictly above the principal point and at "infinite" distance from it. Setting angular tolerance (10°), we filter out all line segments pointing approximately to that direction and recalculate the vanishing point position more precisely. Then, exploiting the criterion of mutual orthogonality of the 3 vanishing points, we can greedily find the 2 others.

As soon as the 2D coordinates of the vanishing points in principal point-based coordinate system are defined, we transform them to 3-dimensional vectors adding negative focal length as a third coordinate and normalize their lengths. These vectors are normal vectors of three main surface orientations of the Manhattan world. For an example of how this algorithm performs with Kinect data, see Fig. 5. Red, green and blue colors correspond to the line segments assigned to the three vanishing points detected, while yellow color indicates edges we are not certain about. For the first image, there are many segments corresponding

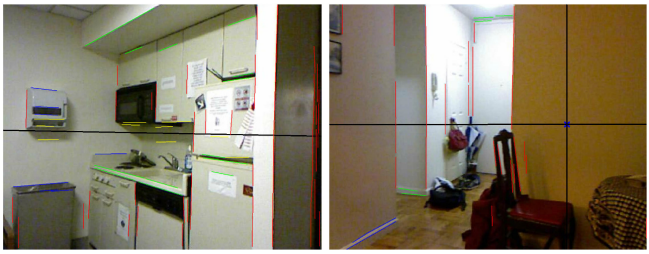

Fig. 5. Vanishing points estimation: red, green and blue line segments are assigned to three different vanishing points. Yellow lines are not classified.

to each color, that allows reaching high accuracy of vanishing point detection. At the same time, for the second image only two blue segments are detected, therefore the error of the vanishing points localization is higher. Generally, it depends on the quality and quantity of edges present in the image, how well they are aligned with three main directions and up to what degree the scene geometry meets the assumption of the Manhattan world. If there are many uniform and non-textured regions in the image or, conversely, many lines oriented in random directions, the accuracy of the vanishing points detection decreases drastically. In order to compensate for these negative effects, we have used an additional procedure of vanishing point position optimization.

Step (d). To optimize the directions of normal vectors we move from the image space to the 3D object space and deal with point cloud representation of the scene (the point cloud was obtained at the end of the pre-processing step). The optimization procedure is based on histogram processing and entropy minimization [12], [10]. The main idea is to analyze points coordinates distributions in the point cloud. For each "vanishing" direction, found just before, we build a corresponding histogram and calculate three values of histogram entropies:

$$E_i = -\sum_{j=1}^{J_{max}} p_{i,j} \log\left(p_{i,j}\right), \tag{1}$$

where E_i is the entropy of the histogram corresponding to i_{th} direction ($i \in [1 \dots 3]$), p_{ij} the probability of j_{th} bin and J_{max} the number of bins.

The idea behind this is illustrated in Fig. 6. There, we can see three projections of the point cloud (view from above) corresponding to the 2nd scene shown in Fig. 5. The 1st projection is in the original camera-based coordinate system. The 2nd image shows the projection of the point cloud transformed to the coordinate system aligned with the directions of the "vanishing" vectors found on the previous step. Below and on the left of each projection we put corresponding histograms of the coordinates distributions along corresponding directions. Obviously, in the case of the perfect alignment of all surfaces with coordinate axes we should have one narrow peak for each surface position. In this case the entropy of the histograms is minimized. Indeed, in comparison with the 1st

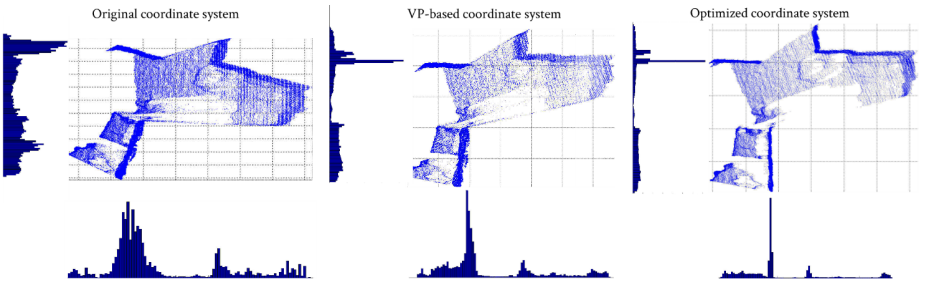

Fig. 6. Example of a point cloud orientation before and after the optimization procedure (see text for more explanations)

image, the peaks of the histograms for the 2nd projection are better pronounced but still not perfect. To further improve it, we are looking for such vector orientations (close to the "vanishing" directions) which would make those peaks as narrow as possible. The 3rd projection and its histograms in Fig. 6 illustrate the perfect alignment achieved using this optimization procedure. Let us note that here in order to illustrate the concept we have chosen an image where the vanishing point localization algorithm performed poorly due to reasons described previously. For most of the other images the difference between the "vanishing" vectors alignment and the optimized alignment is not that significant. Thus, using small variations in the vector directions we aim on finding such combination of three vectors that would minimize the following objective function:

$$\text{Obj.func.} = \sum_{i=1}^{3}\left(\sum_{j=1}^{J_{max}} p_{i,j}\log\left(p_{i,j}\right)\right)^2 + \alpha_e\sum_{i\neq k}^{3}|(N_i, N_k)| + \beta_e\sum_{i=1}^{3}||N_i - VP_i||, \quad (2)$$

$$\text{subject to } ||N_i|| = 1, \quad (3)$$

where N_1, N_2, N_3 are the normal vectors of three main surfaces of the Manhattan world and VP_1, VP_2, VP_3 are the vectors corresponding to three vanishing points. The 1st term is the sum of squares of the three entropies (along each direction) that should be minimized, the 2nd term expresses vectors orthogonality (notation $(\,,\,)$ stands for the dot product, since minimizing the dot product of two vectors we force them to be approximately orthogonal), the 3rd term is needed to keep the result close to the original estimation of the vectors directions. Coefficients α_e and β_e are set experimentally. The constraint (see (3)) keeps all vectors to be of a unit length. The objective here is to minimize the entropy of each component distributing the error equally between them (which is important for the following steps). To perform this, on each iteration we transform the point cloud to a new coordinate system aligned with current orientation of the normal vectors. As soon as the optimal directions of three main vectors are defined, we finally transform our point cloud into corresponding coordinate system and stay there for further processing.

Fig. 7. Kinect data segmentation. 1st row, from left to right, gradient: of the RGB image, of the depth image, of the gradient orientation map. 2nd row: combined gradient map, the result of watershed segmentation, the original RGB image (for comparison).

2.3 RGB-D Image Segmentation and Patch Fitting

Step (a). As soon as the main Manhattan world orientations are defined, the goal of the next step is to split the point cloud into multiple sub-clouds in such a way that each sub-cloud would correspond to one surface of one object. Since direct point cloud segmentation in 3D space is generally a non-trivial task, rather than doing that, we prefer to segment our data in the image space. We propose a method consisting of the following steps: (1) Compute the color gradient of the RGB image (as it is done in [19]); (2) Compute the gradient of the depth image, perform its gamma correction in order to filter out the edges due to data quantization; (3) Estimate the orientation of the depth gradient (i.e. the direction of the maximum rate of change) and, in turn, compute the gradient of this orientation map; (4) Compute a combined gradient map for each pixel taking the maximum value among the three gradient maps. As a result of these computations, we are able to highlight boundaries between surfaces regardless of their colors, depths and orientations. Then we apply a watershed algorithm to segment the combined gradient map (after using some pre-processing based on morphological operations and filling small holes based on region properties). An example of the output of the process is also shown in Fig. 7.

Step (b). Patch fitting is applied to each group of pixels computed after step (a) based on the following patch grouping strategy. For each image segment obtained just before, we consider a group of corresponding points in the point cloud and for each coordinate direction we calculate their mean and variance. Then, we represent each given group of points in the point cloud space with a rectangular patch perpendicular to the direction with minimum variance and intersecting

the corresponding axis at the coordinate corresponding to the calculated mean value. Patch boundaries are parallel to two another coordinate axes. For each patch we take M points from each side and fit 4 lines forming the rectangle boundaries. In our implementation $M = 50$. When the orientation and position of each patch are defined, we project all corresponding points on its surface, finding the intersection of a ray coming from the camera position (the origin of the coordinate system) through the given point and the patch plane. Next, we remove all points whose coordinates have changed greatly in comparison with their original position (they are considered as outliers):

$$(x - x_{pr})^2 + (y - y_{pr})^2 + (z - z_{pr})^2 \leq \alpha_{pr}, \tag{4}$$

where (x, y, z) denotes the current ray, (x_{pr}, y_{pr}, z_{pr}) its intersection with the surface. The threshold α_{pr} is set experimentally.

Then we recalculate the positions and boundaries of all patches again. It should be noted that while trying to fit patches to each group of points, we faced a problem with the Kinect data. According to the technical specifications of the device, it is supposed to work on the depth range no greater than 3.5 m. Although in practice it is possible to obtain depth data for distances up to $8-10\,m$, the depth resolution of the device in the far range decreases drastically. For example, for the distance of 2 m the depth is known with 10 mm precision, while for 9 m it is about 25 cm. Since in most of indoor environments the depth range is greater than 3 m, we have to deal with higher uncertainty in points positions involved. If we do not take into account this fact the algorithm tends to fit horizontal patches to the most distant group of points. To compensate for this negative effect, we have introduced an additional normalization step in order to decrease the probability of fitting with horizontal patches for great distances. This normalization is derived from the original normalization procedure used while converting raw non-linear Kinect output to actual depth map using the following mapping function:

$$d = \frac{\alpha_k}{\beta_k - D}, \tag{5}$$

where d is the real depth in meters and D is the Kinect raw data.

2.4 2½D Model Construction

Step (a). To reduce the number of patches and to combine sub-clouds that correspond to the same surface in the real world, we perform a clustering of patches along each direction taking into account their mutual location. We can explain the method used considering, for example, patches perpendicular to the X-axis (and, consequently, called X-patches). Such patches, apart from the orientation, have 5 parameters: one x-coordinate indicating its location and two y- and z-coordinates defining the patch boundaries. If two X-patches are combined, the 5 coordinates of the resulting X-patch are being calculated as follows:

$$x_\Sigma = \frac{x_1(y_{1,\max}-y_{1,\min})(z_{1,\max}-z_{1,\min})+x_2(y_{2,\max}-y_{2,\min})(z_{2,\max}-z_{2,\min})}{(y_{1,\max}-y_{1,\min})(z_{1,\max}-z_{1,\min})+(y_{2,\max}-y_{2,\min})(z_{2,\max}-z_{2,\min})}; \tag{6}$$

$$y_{\Sigma,\min} = \min(y_{1,\min}, y_{2,\min}), \quad y_{\Sigma,\max} = \max(y_{1,\max}, y_{2,\max}); \tag{7}$$

$$z_{\Sigma,\min} = \min(z_{1,\min}, z_{2,\min}), \quad z_{\Sigma,\max} = \max(z_{1,\max}, z_{2,\max}). \tag{8}$$

Two x-patches should be combined if:

$$\begin{cases} |x_1 - x_2| < \gamma, \\ \dfrac{(y_{\Sigma,\max}-y_{\Sigma,\min})(z_{\Sigma,\max}-z_{\Sigma,\min})}{(y_{1,\max}-y_{1,\min})(z_{1,\max}-z_{1,\min})+(y_{2,\max}-y_{2,\min})(z_{2,\max}-z_{2,\min})} < \mu. \end{cases} \tag{9}$$

The second line in the system of equations (9) expresses the idea that the area of the resulting patch should not be much greater than the sum of areas of two original patches. The thresholds γ and μ are set experimentally. For Y- and Z-patches the procedure is the same.

Step (b). To construct a 3D occupancy grid consisting of plenty of small cubes – voxels, all patch coordinates are used. Most of occupancy grid approaches use a grid with a fixed step size along each direction. From one side, this step is too small for large objects, from another one, it means that the position of all surfaces are determined with an accuracy of at most half of the step, which is not always sufficient. Instead, we propose to tackle these two problems using image-based grids. It means that instead of the fixed step, we build our grid on the coordinates of patches using their positions and boundaries. It allows to obtain a small number of voxels, but enables to describe the scene in a more precise way. For each voxel we set a flag: either "occupied" or not, depending on whether or not (and how many in relation to the voxel side areas) there are points on its sides visible from the camera position.

2.5 Manhattan Depth and Orientation Maps

After obtaining the final 2½D model, for further processing it is convenient to come back to projection (image) based representation and calculate a new ("Manhattan") depth map together with a surface orientation map. This first map gives a recalculated position of each point corresponding to each image pixel in the 3D space, while the orientation map indicates to what kind of patch (in terms of direction) this point belongs. For an illustration see Fig. 8.

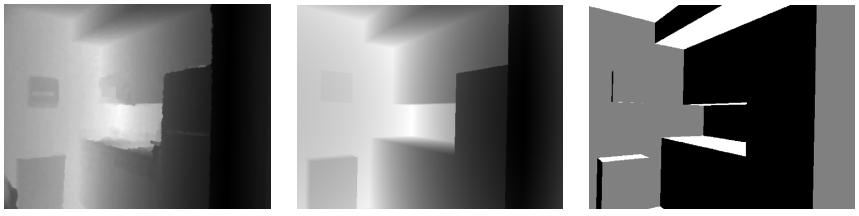

Fig. 8. Original depth map, the Manhattan depth and the orientation map

3 Results and Dicussion

We have conducted several series of experiments in order to find optimal parameters for all steps of the 2 1/2D modelling algorithm. Optimal values of all coefficients and thresholds found experimentally are listed in Table 1.

In most of these experiments we have used the Kinect data from the New York University Depth datasets [14]. The motivation behind this choice is that these datasets contain a large variety of indoor scenes and, consequently, provide plenty of illustrations of all possible difficulties and limitations which can affect the process of geometry reconstruction. Moreover, with such standard dataset, we have access to camera calibration parameters.

The 2 1/2D models obtained for several indoor scenes are shown in Fig. 9. It can be observed that the algorithm performs well for scenes really close to their Manhattan representation, while for objects of arbitrary shapes and orientations (such as a chair and a table in the third image) is may produce noisy results. On this step of the project the only way to evaluate the results is visual observation. There are no metrics which would allow to quantitatively evaluate the quality of the reconstructed geometry, since from the beginning we aim on approximate

Table 1. Optimal values of parameters used in geometry estimation

Parameter	Value
γ_{bf}, the threshold for pixel filtering (Section 2.1)	0.06
J_{max}, the number of bins (formula 1)	100
α_e (formula 3)	8
β_e (formula 3)	2
M, number of points for patch fitting (Section 2.3)	50
α_{pr} (formula 4)	0.15
γ (formula 9)	0.05
μ (formula 9)	1.1

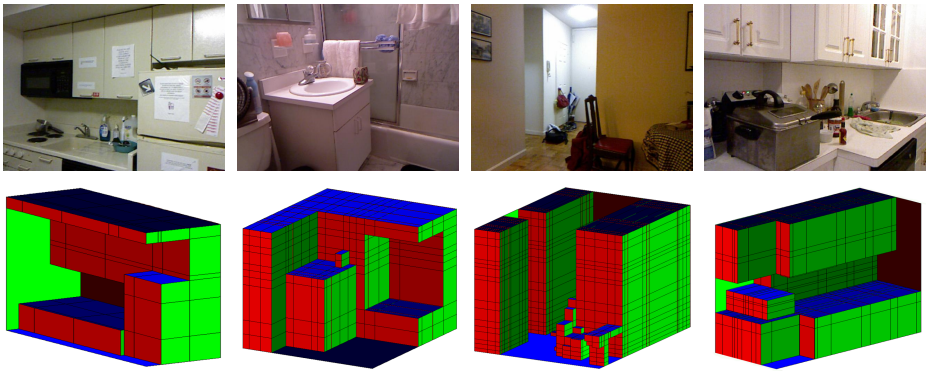

Fig. 9. Some examples of 3D models

"Manhattan" world reconstruction which is not precise by definition. Nevertheless, if consider some reference surfaces and estimate the difference between their real and estimated positions, the result depends, first, on the Kinect accuracy of depth estimation varying with the distance from the object to the camera and also on the thresholds set while constructing the voxel-based 3D model. The lower the thresholds for patch combining, the more complex the model, the more precisely the surface positions can be defined. At the same time, setting very low thresholds leads to higher errors in the discriminating between occupied and non-occupied voxels due to sparcity of the point cloud. In this sense, the values of the parameters specified in the method description seem to be an "optimal" tradeoff. Thus, in our implementation the estimated maximum error of the surface position estimation is $5\,cm + 1/2\times$(resolution of the depth sensor for the given distance between the surface and the camera). Here the value 5 cm corresponds to the threshold for patching grouping.

4 Conclusion

Using the Manhattan world assumption we have proposed a new framework for global $2^1/_2$D geometry estimation of indoor environments from single low quality RGB-D images. This framework is based on original algorithms for: - RGB-D data segmentation in image space; - planar patch fitting to corresponding subclouds of points; -following procedure of patch grouping and - 3D model construction built on image-based voxel occupancy grid. This framework performs robustly even in the conditions of low resolution, significant depth distortion, nonlinearity of depth accuracy and presence of noise.

One source of error that we are planning to fight in future work, is depth map distortion. In our experiments we found out that in all cases the representation of a flat surface given by the Kinect had rather significant curvature. Therefore the depth scale should be normalized not only depending on the distance between the camera and the object but also on the distance between the given point and the optical axis. In addition, taking into account the working principle of the Kinect sensor, we expect this curvature to be also dependent on the absolute distance between the camera and the object, which further complicates the possible procedure of the camera calibration. To our knowledge, there is no work reporting on that, and this kind of calibration is not yet developed.

It would be also interesting to incorporate accelerometer data provided by the Kinect sensor in vanishing points detection. Having the direction of gravity, it is possible to directly reconstruct the orientation of the horizon line and, consequently, estimate the reliability of the vanishing points localization and increase the robustness of the system. Unfortunately, due to significant oscillations of the accelerometer output, the accuracy of a single measurement is not sufficient for this purpose. This problem probably can be solved by averaging the data over a series of measurements, but since in this paper we limit ourselves to a single Kinect shot, we leave this idea for future work.

References

1. Coughlan, J.M., Yuille, A.L.: Manhattan world: Compass direction from a single image by bayesian inference. In: IEEE International Conference on Computer Vision (ICCV), pp. 1–10. IEEE Press, New York (1999)
2. Oliva, A., Torralba, A.: Depth estimation from image structure. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 1226–1238 (2002)
3. Saxena, A., Chung, S., Ng, A.Y.: Learning depth from single monocular images. In: Neural Information Processing Systems (NIPS), vol. 18. MIT Press (2005)
4. Delage, E., Lee, H., Ng, A.Y.: A dynamic bayesian network model for autonomous 3D reconstruction from a single indoor image. In: CVPR Conference (2006)
5. Barinova, O., Konushin, V., Yakubenko, A., Lee, K., Lim, H., Konushin, A.: Fast Automatic Single-View 3-d Reconstruction of Urban Scenes. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 100–113. Springer, Heidelberg (2008)
6. Hoiem, D., Efros, A.A., Hebert, M.: Automatic photo pop-up. In: SIGGRAPH Conference (2005)
7. Torralba, A., Sinha, P.: Statistical context priming for object detection. In: ICCV Conference (2001)
8. Kien, D.: A review of 3d reconstruction from video sequences. University of Amsterdam ISIS Technical Report Series (2005)
9. Pollefeys, M.: Visual 3d modelling from images. Tutorial notes. Technical report (2007)
10. Gallup, D., Frahm, J., Mordohai, P., Yang, Q., Pollefeys, M.: Real-time plane-sweeping stereo with multiple sweeping directions. In: CVPR Conference (2007)
11. Sinha, S.N., Steedly, D., Szelinski, R.: Piecewise planar stereo for image-based rendering. In: ICCV Conference, pp. 1881–1888 (2009)
12. Olufs, S., Vincze, M.: Robust Room-Structure estimation in Manhattan-like Environments from dense 2.5 range data. In: WACV Workshop, pp. 118–124 (2011)
13. Paris, S., Durand, F.: A Fast Approximation of the Bilateral Filter Using a Signal Processing Approach. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 568–580. Springer, Heidelberg (2006)
14. Silberman, N., Fergus, R.: Indoor scene segmentation using a structured light sensor. In: ICCV Workshop, pp. 601–608 (2011)
15. Van de Weijer, J., Gevers, T., Smeulders, A.W.M.: Robust photometric invariant features from the color tensor. IEEE Transactions on Image Processing 15(1), 118–127 (2006)
16. Kovesi, P.D.: MATLAB and Octave Functions for Computer Vision and Image Processing
17. Rother, C.: A new approach to vanishing point detection in architectural environments. Image and Vision Computing 20(9-10), 647–655 (2002)
18. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the spatial layout of cluttered rooms. In: ICCV Conference, pp. 1849–1856 (2009)
19. Neverova, N., Konik, H.: Edge-based method for sharp region extraction from low depth of field images. In: VCIP Conference (2012)