Proceedings of the 2014 IEEE
International Conference on Robotics and Biomimetics
December 5-10, 2014, Bali, Indonesia

Support-plane Estimation for Floor Detection to Understand Regions' Spatial Organization

Lei Wang, Zhimin Zhou, Jun Wu, Yuncai Liu, Xu Zhao

***Abstract*— Plane fitting plays an important role in image processing and computer vision. It is challenging because of the outliers that do not follow the plane pattern. In this work, we address the problem of support-plane fitting for room floor detection from point clouds that are generated from depth image. Based on the geometric layout of data, an optimization problem is derived to estimate the support-plane. Algorithms are also proposed to deal with data noise. The floor detection is achieved by support-plane fitting, and is employed as a reference to analyze the spatial organization of room scene. A projection method is presented to form the organization map. Experiments demonstrate the proposed method is more robust, and it achieves remarkable performance in understanding the spatial organization.**

I. INTRODUCTION

Plane fitting (or plane estimation) is a fundamental problem in image processing and computer vision areas. The task is to construct a plane that has the best fit to a set of 3D points. It is essential in many applications. For example, the floor plane detection plays an important role in understanding the spatial organization of objects in the environment [1]. However, fitting a plane is challenging due to the large number of outliers [2] which do not follow the pattern of the other observations.

Recently, indoor mapping can be achieved in a more convenient way, and the 3D point cloud can be obtained by an RGBD sensor (e.g. Microsoft Kinect). The indoor environment usually comprises a lot of planar surfaces (e.g. floor, wall and table surface) which can be used as reference for object segmentation and spatial relation modeling. In this work, we address the problem of plane fitting for floor estimation from RGBD images, and use the floor as a reference to analyze the spatial relations of objects for indoor scene.

Plane fitting has been developed for decades. Various methods were proposed to handle the problem. The classical approach of total least squares (TLS) [3] is a technique to solve a system of equation $AX \approx B$ for X, where $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times d}$ are the given data. However, TLS can't deal with data that contain many outliers. The method of iteratively re-weighted least squares (IRLS) [4] is an iterative approach in which each step involves solving a weighted least squares problem, and is capable of dealing with outliers in robust regression. The RANSAC algorithm [5] can iteratively fit a model for data containing a lot of outliers. Schnabel et al. [6] employed RANSAC for plane extraction efficiently and precisely, but the parameters, such as outlier ratios and neighbor point relations, have to be set properly. Poppinga et al. [7] proposed an approach to detect planes by combining region growing and plane fitting for noise 3D range images. Borrmann et al. [8] used 3D Hough Transform to detect planes in 3D point cloud, and they also presented a novel approach to design the accumulator. The method of Hough Transform only works well if there are enough points lying exactly in a certain plane, and the computational cost is high.

This work has been partially supported by the National Natural Science Foundation of China (No. 61273285, 61375019) and 973 Program (No. 2011CB302203). The authors are with the Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, P.R. China {wltongxing, 115236, zhugetian, whomliu, zhaoxu}@sjtu.edu.cn

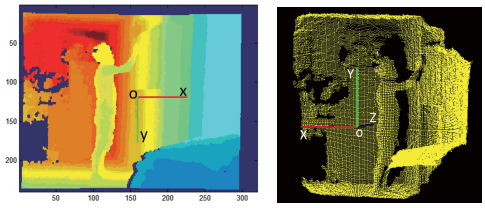
(a) (b) (c)

Fig. 1. Floor detection for object segmentation and spatial relation analysis. (a) shows an RGB image of an indoor scene. In (b) the support-plane is estimated to detect the floor and segment objects in the corresponding depth (range) image, and the green region indicates the floor, while the white area is due to the missing data from sensor. The objects are projected onto the floor plane to form an organization map, as shown in (c).

Although most of the previous methods can estimate plane on data with noise robustly, they tend to fail when the outlier points are dominant in the whole given data set. For example, the floor plane to be detected only occupy a small proportion of the whole image, and most of the obtained data from sensor are outliers for the estimation of floor plane. In this case, most of the previous approaches are ineffective. Another problem is the huge amount of data. A depth image with resolution of 640×480 contains over 300,000 points, which are not acceptable for most of the methods.

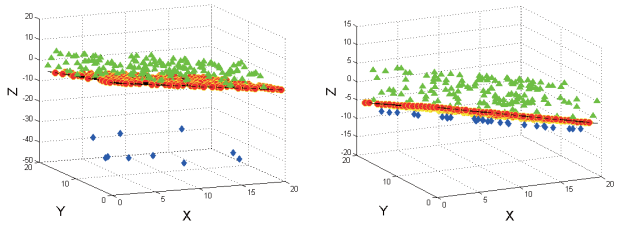
Most of the acquired data points lie above the floor in room scene, except for some data noise below. In this work a method of support-plane fitting is presented, and the technique of random sampling is applied to reduce the input data without decreasing performance of plane estimation. Support-plane means that all the objects are on one side of the plane and little noise data on the other side. For example, given a 3D point cloud of indoor scene, the table, the person and other objects are above the floor, and no data below the floor can be obtained by the sensor. The plane where the floor lies is a support-plane. Fig. 1(a) shows an RGB image of a room from Cornell Activity Dataset [9]. Based on the corresponding depth image, a support-plane can be

978-1-4799-7397-2/14/$31.00 © 2014 IEEE



(a) (b)

Fig. 2. Coordinate systems of depth map and 3D points cloud.



(a) (b)

Fig. 3. Two types of data noise in support-plane fitting. The red circles indicate the points lying in the support-plane, the green triangles (points above the plane) are the first type of outliers, and the blue diamonds are the second type of outliers, which is the data noise. (a) shows one type of data noise with a few unstable noise points, while (b) shows another type with many stable noises.

constructed, and the floor is detected in the meantime. In Fig. 1(b), the green region is made up of points that fall into the floor plane. The objects in the room are then projected to the floor plane along the normal direction, as shown in Fig. 1(c), which makes the spatial layout clear.

In order to analyze the indoor scene, one can adjust the RGBD camera to ensure that part of the floor region is visible, and then the floor plane is detected by support-plane estimation. The floor is used as a reference for segmenting objects and modeling the spatial relations of those objects in the room. We assume that the acquired data from one object have similar spatial distributions. The method of region growing is applied for object segmentation. Then the data points from segmented objects are projected onto the floor plane to form a map representing the objects' spatial organization.

The details of support-plane estimation are given in Section II. The projection from 3D objects to 2D map is illustrated in Section III. Section IV shows the experimental results.

II. SUPPORT-PLANE FITTING

In this section, we develop a method for support-plane estimation by solving a linear optimization problem. The depth maps obtained from an RGBD device can be converted to a 3D point cloud. Generally, the point cloud is noisy because of the inaccuracy of the capture device. This section deals with two types of capture errors. One type has a few unstable noise points, while the other type has many stable noises, as shown in Fig. 3.

A. From Depth Map to Point Cloud

A depth map is a single channel image with the same resolution as the corresponding RGB image. Each pixel value of the depth map denotes a distance between the object and the RGBD sensor. Suppose the value at pixel (u, v) is d, the center of the map is (u_0, v_0), then the 3D coordinate (X, Y, Z) of the object on this pixel can be calculated by the equations:

$$X = -(u - u_0) * d/c, \quad (1)$$

$$Y = -(v - v_0) * d/c, \quad (2)$$

$$Z = d, \quad (3)$$

where c is the sensor constant that can be acquired by calibration. The coordinate systems of the depth map and 3D point cloud are shown in Fig. 2.

B. Outliers

Many previous work have explored robust plane fitting with outliers. However, there is no clear definition of outliers. This work defines two different kinds of outliers for support-plane estimation. The first type denotes the data that lie above the support-plane, and most of the input data belong to this type. For example, in a room the support-plane is the floor plane, but the majority of the acquired point clouds come from other objects (e.g. wall, furniture). The second type is the noise data due to the inaccuracy or instability of the sensor. There are many types of distribution of noise data, and we deal with two of them in this paper, as shown in Fig. 3. In the first case the number of noisy points is small but the errors are large; while in the second case the number of noisy points is large but the errors are small.

C. Algorithm

We first propose the algorithm to deal with the outliers above the support plane, and then present algorithms to deal with the noise.

A plane has the form:

$$Ax + By + Cz + D = 0, \quad (4)$$

where A, B, C and D are plane parameters that need to be estimated. The coordinate system of the point cloud is shown in Fig. 2(b). Generally, the support-plane is unlikely to be vertical, so B is nonzero. Equation (4) can be rewritten as:

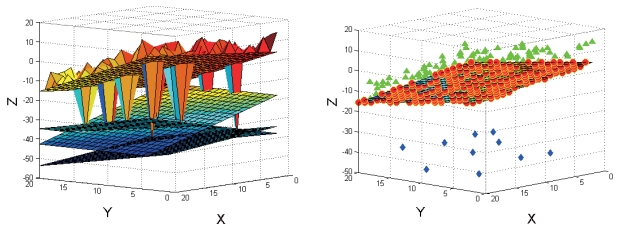
$$y = A' \cdot x + C' \cdot z + D'. \quad (5)$$

Suppose the point cloud contains m samples $(x_i, y_i, z_i)_{i=1}^{m}$. Because the noise is not considered so far, all those points are in or above the support-plane, which means:

$$y_i \geq A' \cdot x_i + C' \cdot z_i + D', \quad i = 1, ..., m. \quad (6)$$

Because of the geometrical relations and constraints, the sum of the vertical distances between all the points and the best fit support-plane must be minimum:

$$\text{minimize} \quad \sum_{i=1}^{m} (y_i - A' \cdot x_i - C' \cdot z_i - D'). \quad (7)$$



(a) (b)

Fig. 4. Support-plane estimation by data with the first type noise. (a) shows that different false support-planes can be fitted due to the noise. (b) shows the stable support-plane estimated by our method.

In order to solve this problem, an objective function $f(t)$ and constraints $g_i(t)$ are defined as:

$$f(t) = \sum_{i=1}^{m} y_i - \sum_{i=1}^{m} -x_i \cdot t_1 - \sum_{i=1}^{m} z_i \cdot t_2 - m \cdot t_3, \quad (8)$$

$$g_i(t) = y_i - x_i \cdot t_1 - z_i \cdot t_2 - t_3, \quad i = 1, ..., m, \quad (9)$$

where $t = (t_1, t_2, t_3)$ is the optimization variables. The convex minimization problem is written as:

$$\begin{aligned} &\underset{t}{\text{minimize}} \quad f(t) \\ &\text{subject to} \quad g_i(t) \geq 0, \quad i = 1, ..., m. \end{aligned} \quad (10)$$

The solution $t^* = (t_1^*, t_2^*, t_3^*)$ determines the support-plane:

$$y = t_1^* \cdot x + t_2^* \cdot z + t_3^*. \quad (11)$$

Then we handle the data with noise of the first type shown in Fig. 3(a). Due to the discrete noise, the estimated support-plane by (10) is unstable. If different data noises are observed, different support-planes will be fitted, as shown in Fig. 4(a). Inspired by this, a procedure of repeating fitting is proposed. After a support-plane is estimated, omit part of points that fall into the plane with a ratio λ and fit the support-plane again. The procedure is kept on until the difference between the two successive planes are negligible, and a stable support-plane is estimated, as shown in Fig. 4(b).

The angle θ between the normals of the two successive planes is used to denote the difference. The normals are written as n_1 and n_2. The angle between them is:

$$\theta = \arccos \frac{n_1 \cdot n_2}{|n_1| \cdot |n_2|}, \quad (12)$$

where $|n_1|$ and $|n_2|$ indicate the vector norm.

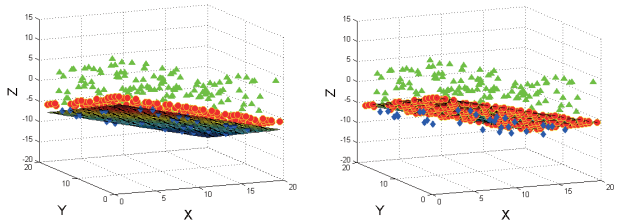
In general, the number of points that fall into a support-plane are larger than the number of noise points. A ratio r between the plane points and all data points is defined as:

$$r = m_p / m, \quad (13)$$

where m_p denotes the number of points that fall into the estimated plane, and m is the number of all data points. The value r of ground-truth support-plane is larger than that of false support-plane estimated by noise data, since the number of the first type of noise is small.

Algorithm 1 Support-plane stabilization

1: **procedure** INIT
2: $m \leftarrow$ number of *data points*
3: $th1 \leftarrow$ threshold of *angle between plane normals*
4: $th2 \leftarrow$ threshold of *plane point ratio*
5: $\lambda \leftarrow$ ratio of *omitted plane points*
6: $\theta_0 \leftarrow \inf$
7: $r_0 \leftarrow 0$
8: $j \leftarrow 1$
9: **procedure** LOOP
10: **repeat**
11: *Support-plane* fitted by (10)
12: $\theta_j \leftarrow$ update by (12)
13: $m_p \leftarrow$ from *Support-plane*
14: $r_j \leftarrow$ update by (13)
15: $m \leftarrow m - \lambda \cdot m_p$
16: $j \leftarrow j + 1$
17: **until** $(\theta_j - \theta_{j-1} < th1$ && $r_j > th2)$



(a) (b)

Fig. 5. Support-plane estimation by data with the second type noise. (a) shows the false support-planes fitted by the noise. (b) shows the support-plane estimated by adjusting.

The approach to deal with the first type of noise is illustrated in Algorithm 1.

When it comes to the second type of noise shown in Fig. 3(b), Algorithm 1 fails. Because the support-plane estimated with this type of noise satisfied the constraints in Line 17 of Algorithm 1. This case is shown in Fig. 5(a). Because the errors of the noise points are stable, the distribution patters is well formed. Usually a parallel plane with a displacement to the ground-truth support-plane can be estimated by Algorithm 1. The displacement indicates the error that is determined by the capture device. The error is given in advance, and is used to adjust the support-plane.

From (11), the estimated support-plane is also written as:

$$y - t_1^* \cdot x - t_2^* \cdot z - t_3^* = 0. \quad (14)$$

The ground-truth support-plane is parallel, and is written as:

$$y - t_1^* \cdot x - t_2^* \cdot z - t_3^o = 0. \quad (15)$$

The error ϵ determines the perpendicular distance of those two plane:

$$\epsilon = \frac{|t_3^* - t_3^o|}{\sqrt{1 + (t_1^*)^2 + (t_2^*)^2}}. \quad (16)$$

Then the adjusted support-plane is given as:

$$y = t_1^* \cdot x + t_2^* \cdot z + t_3^* + \epsilon \cdot \sqrt{1 + (t_1^*)^2 + (t_2^*)^2}. \quad (17)$$

This procedure is summarized in Algorithm 2.

Algorithm 2 Support-plane adjustment

1: **procedure** STABILIZATION
2: *Support-plane* estimated by *Algorithm 1*
3: **procedure** ADJUSTMENT
4: $\epsilon \leftarrow$ error of *data noise*
5: *Final support-plane* estimated by (17)

D. Downsampling and Floor Detection

It is unnecessary and ineffective to utilize the whole point cloud of a room scene for support-plane fitting, because the plane only takes a small portion, and most points in the cloud are outliers. Most points of the acquired data are counted as outliers, and reasonable downsampling contributes to outlier reduction and reduces the computation time. The common methods of image downsampling are evenly sampling on a grid or random sampling. We adopted the later in our work.

The proposed method is based on the geometrical relationship of data, so three random points in the support-plane are enough for the fitting task. This feature also enables us to employ downsampling to reduce the computational cost. During the random sampling, prior knowledge can be employed. For example, in the task of room floor detection, the floor is more likely to be in the bottom of the depth map, and we can sample more points at the bottom than at the top.

Two approaches for floor detection are designed based on random sampling. One is interactive floor detection. A random image region containing part of the floor is given manually, and is randomly sampled. With the randomly sampled points, a support-plane is fitted to detect the floor by Algorithm 2. Another approach is an automatic floor detection by human detection, with the assumption that the floor is underneath the person. The human detector of Deformable Part Models (DPM) [10] is applied to detect the person in the RGB image. The detected bounding box serves the similar role to the manually labeled region in the first method. Note that we consider both the points lying exactly in the support plane and the points near the plane as floor pixels. This treatment is robust to data noise in the experiments.

III. SPATIAL ORGANIZATION ESTIMATION

The method of support-plane estimation, proposed in Section II-C, is suitable for floor detection. Floor plays an important role in room scene segmentation, and is also widely used as a reference to analyze the object relations. After detecting the floor, the further semantic understanding of the room environment can be achieved.

In this section, a traditional method of region growing [11] is employed for object segmentation. Then those segmented objects are projected to the floor plane, forming a spatial organization map.

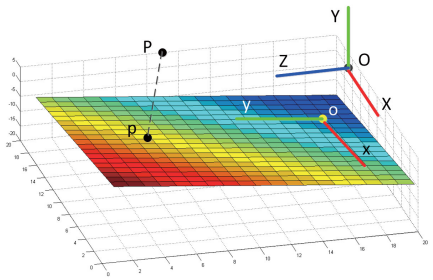


Fig. 6. 3D point is projected to the floor plane. P is a 3D point, and p is the projected 2D point on the floor plane.

A. Segmentation

Region growing is a simple region-based method for image segmentation. This approach examines the features of neighboring pixels and determines whether the pixel should be appended to the region. For a color image, the color feature is usually adopted. By region growing, the whole image is partitioned into several regions, and the pixels of each region have similar colors.

We adopt the region growing method on depth image. By examining the depth of nearby pixels, the image can be segmented into regions. Each region has pixels of similar depth values. One factor that may affect the procedure is the connection between objects and floor. Since most of the room objects lie on the floor, and the pixels at the joint locations have similar values, no mater the pixels are from the floor or from the objects. We remove the floor points and use the remaining for segmentation by region growing.

Another problem is the selection of seed points. The general criterion is to select them randomly, but the points near the view center and those near the camera have priority.

B. Projection

2D map of the objects is effective to demonstrate the organization (e.g. distance of object to the camera, distance between an object and another). An example is displayed in Fig. 1(c). In order to generate a 2D map, the typical method is to project the 3D objects into a reference plane which is the floor plane in this work.

The key issue of projection is the coordinate transformation. The coordinate systems of 3D point cloud are shown in Fig. 2(b) and Fig. 6. By the coordinate transformation, all points are mapped to 2D coordinate system xoy. To simplify the transformation, Z-axis is directly mapped to y-axis, and Y-axis is omitted, since we pay close attention to horizontal organization (XOZ) instead of vertical organization. The details of projection is described in Algorithm 3.

IV. EXPERIMENTAL RESULTS

The experiments were conducted on two datasets to evaluate the proposed method. One is the Cornell Activity Dataset [9]. There are 60 RGB-D videos, and each video comes with RGB images, depth image and the skeletons. It involves 12 activities in 5 different scenarios. Another one is our action dataset. It includes 6 actions that carried out by 3 persons, and each action has several video instances.

Algorithm 3 Projection to plane

Require: 3D points $(X_i, Y_i, Z_i)_{i=1}^{m}$

procedure COORDINATE TRANSFORMATION
  $n \leftarrow$ normal of *plane*
  $x_b \leftarrow (1, 0, 0)$
  $y_b \leftarrow n$
  $z_b \leftarrow x_b \times y_b$
  $x_b \leftarrow y_b \times z_b$
  $x_b \leftarrow x_b/|x_b| \quad y_b \leftarrow y_b/|y_b| \quad z_b \leftarrow z_b/|z_b|$
procedure CALCULATE 2D COORDINATE
  for i=1:m **do**
    $x_i \leftarrow (X_i, Y_i, Z_i) \cdot x_b$
    $y_i \leftarrow (X_i, Y_i, Z_i) \cdot z_b$

A. Experimental Setting and Evaluation Criterion

The Matlab toolbox of SeDuMi [12] is used to solve the constrained optimization problem in (10). Since there are few noise points of the first type, the parameters that control the loop in Algorithm 1 take loose values. Specifically, the threshold $th1$ in this algorithm is set to 0.01, and $th2$ is set to 0.01. The ϵ in Algorithm 2 denotes the noise error, and depends on the distance to the camera. In the experiments, it is set to 15. The points within the distance of 50 to the estimated support-plane are considered as floor pixels, because of the noise in depth image captured from the sensor, and the noise error becomes larger when the object is far from the camera. All the distance mentioned in this work is in millimeter (mm).

To describe the performance of the floor detection, a definition of overlap between the detected floor and the ground-truth floor is written as:

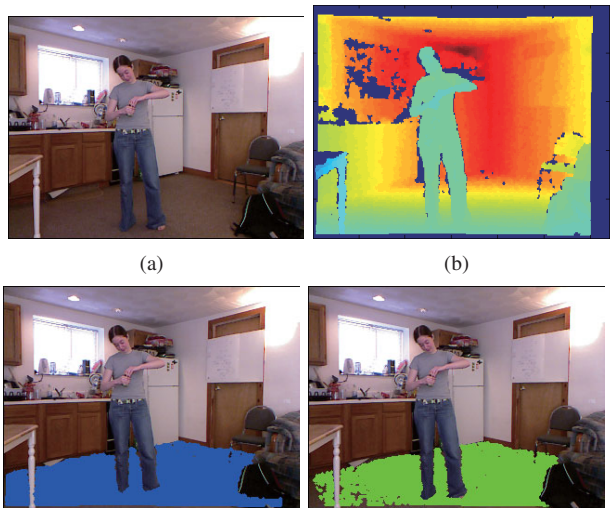
$$o = \frac{\text{area}(D \cap G)}{\text{area}(D \cup G)}, \tag{18}$$

where $D \cap G$ denotes the intersection of the regions covered by the detected floor and the ground-truth floor, while $D \cup G$ is the region union. Large detection overlap means high accuracy.

B. Results of Floor Detection

We compared the proposed method with the RANSAC method on images of resolution 320×240. The input points that are generated from a depth image are over 60,000 (76800 in total), with the invalid points (zero values) being removed. The average running time was 2.93s for RANSAC and 3.46s for our method. The results are shown in Fig. 7, and the proposed method achieves better performance. The estimated floor by RANSAC method is higher than the ground-truth due to the outliers.

Then, points were randomly selected from the depth image to evaluate the performance. The location and number of points are both random, while the floor prior (floor is at bottom of image) is applied. The accuracy of detection is determined by the overlap criteria (18).



(a) (b) (c) (d)

Fig. 7. Results of floor detection by RANSAC and our method. All the points from the depth map are used. (a) and (b) are original RGB and depth image. (c) shows the detected floor by RANSAC, and (d) shows our result.

TABLE I
COMPARISON OF RANSAC AND OUR METHOD.

Random points	Time (s)		Overlap	
	RANSAC	Ours	RANSAC	Ours
512	0.01	0.31	0.6807	0.7084
1391	0.03	0.42	0.0202	0.6981
2537	0.05	0.60	0.6901	0.7402
3392	0.05	0.69	0.0097	0.5624
4168	0.09	0.71	0.0085	0.7159
5879	0.20	1.33	0.6440	0.7228

The computational cost and detection accuracy are shown in Table I. The numbers of test points were randomly selected. More points demand more running time, but the accuracy does not necessarily improve because of the outliers. The proposed method consumes more time than RANSAC, but is more stable and more accurate than RANSAC.

The proposed method is based on the geometric constrains of the floor, and is robust to outliers. RANSAC fits the plane with the least error, and tends to be affected by the room objects. Fig. 8 shows the results of floor detection. In Fig. 8(a), the plane fitted by RANSAC is along the wall, because in the sampling there are more points from the wall than those from the floor. In Fig. 8(c), the estimated plane deviates from the floor due to the outliers. Fig. 8(b) and Fig. 8(d) show that our method is free of those problems.

C. Results of Spatial Organization

The whole procedure of the proposed method is shown in Fig. 9. All the work was done on the depth images (as shown in Fig. 9(b) and 9(d)), while the RGB images (as shown in Fig. 9(a) and 9(c)) are only shown for better demonstration.

Based on the detected floor, the region growing method was employed for segmentation. The seed points were selected randomly from the image center to image border. The pixels with zeros values are disqualified for the seed. The results of segmentation are shown in Fig. 9(d).

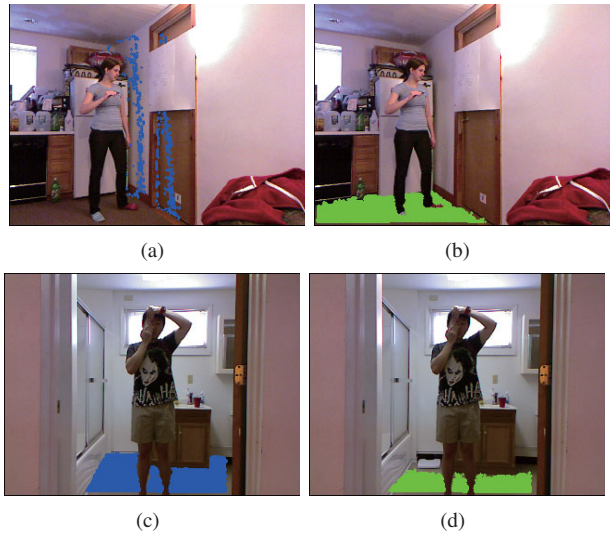


Fig. 8. Results of floor detection by RANSAC and our method. (a) and (c) are estimated by RANSAC, while (b) and (d) are fitted by our method.

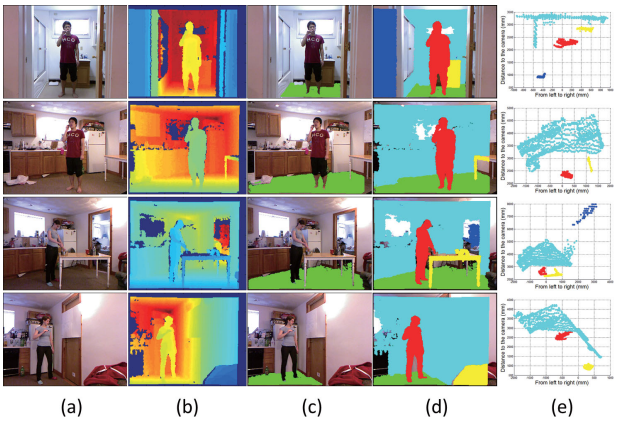


Fig. 9. Results on Cornell Activity Dataset. (a) shows the original RGB images, and (b) shows the depth images. The detected floors by support-plane estimation are painted green in (c). The segmented regions are demonstrated in different colors in (d). The regions from (d) are projected to the floor plane, forming the 2D spatial map in (e).

Then the spatial organization map is formed by the projection process that is presented in Section III-B as shown in Fig. 9(e). The color points in the map are corresponding to those in Fig. 9(d). The map can be deemed as the top view of the indoor scene. The location of the camera is at the bottom center of the organization map, with coordinate of $(0, 0)$.

D. Analysis

Limitations. In support-plane estimation, we assumed that all the objects are on or above the plane, except for some noise points. This results in an easier convex minimization problem. In the experiments, the effectiveness of the proposed method is validated. However, this method is only valid for support-plane. For other planes, such as the plane with objects both above and below it, the proposed method is no longer applicable. This may limit the applications of this method.

Computational cost. The main computational cost of the presented method is to solve the convex minimization problem (10). The quantity of the constraints in the optimization equals the number of the data points. Random sampling can alleviate the computational cost, but the speed is not competitive to RANSAC.

V. CONCLUSIONS

In this paper, we present a method for support-plane estimation. It is based on solving a convex minimization optimization problem that is derived from the geometric structure of the input data. The support-plane is suitable for floor detection. It is robust to outliers and more stable than RANSAC method. The detected floor with our method is used as a reference to analyze the spatial relations of indoor objects. The segmented object points are mapped into the floor plane, forming a spatial organization map. Our proposed method achieved remarkable accuracy. However, the computational cost is high. In the future work, we will explore efficient approaches to further optimize our method.

REFERENCES

[1] Heinrich H Bülthoff and HAHC van Veen, "Vision and action in virtual environments: Modern psychophysics in spatial cognition research," *Vision and attention*, pp. 233–252, 2001.

[2] Victoria Hodge and Jim Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[3] Ivan Markovsky and Sabine Van Huffel, "Overview of total least-squares methods," *Signal processing*, vol. 87, no. 10, pp. 2283–2302, 2007.

[4] PW Holland and RE Welsch, "Robust regression using iteratively re-weighted least-squares," *Communications in statistics part A-theory and methods*, vol. 6, no. 9, pp. 813–827, 1977.

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

[6] Ruwen Schnabel, Roland Wahl, and Reinhard Klein, "Efficient ransac for point-cloud shape detection," in *Computer Graphics Forum*. Wiley Online Library, 2007, vol. 26, pp. 214–226.

[7] Jann Poppinga, Narunas Vaskevicius, Andreas Birk, and Kaustubh Pathak, "Fast plane detection and polygonalization in noisy 3d range images," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3378–3383.

[8] Dorit Borrmann, Jan Elseberg, Kai Lingemann, and Andreas Nchter, "The 3d hough transform for plane detection in point clouds: A review and a new accumulator design," *3D Research*, vol. 2, pp. 1–13, 2011.

[9] Jaeyong Sung, C. Ponce, B. Selman, and A Saxena, "Unstructured human activity detection from rgbd images," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012, pp. 842–849.

[10] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 9, pp. 1627–1645, 2010.

[11] R. C. Gonzalez and R.E. Woods, *Digital Image Processing 2nd Edition*, Prentice Hall, New Jersey, 2002.

[12] J.F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11–12, pp. 625–653, 1999.