LayART: Generating indoor layout using ARCore Transformations

Shreya Goyal
IIT Jodhpur

Naimul Khan
Ryerson University

Chiranjoy Chattopadhyay
IIT Jodhpur

Gaurav Bhatnagar
IIT Jodhpur

***Abstract*—Reconstructing an indoor scene and generating a layout/floor plan in 3D or 2D is a widely known problem. Quite a few algorithms have been proposed in the literature recently. However, most of the existing methods either use RGB-D images, thus requiring a depth camera, or depend on panoramic photos with the assumption that there is little to no occlusion in the rooms. In this work, we proposed generation of layout using an RGB image captured using a simple mobile phone camera. We take advantage of Simultaneous Localization and Mapping (SLAM) to assess the 3D transformations required for layout generation. SLAM technology is built-in in recent mobile libraries such as ARCore by Google. Hence, the proposed method is fast and efficient, while giving the user freedom to generate layout by simply taking a few conventional photos, rather than relying on specialized depth hardware or occlusion-free panoramic photos.**

***Keywords*-Layout estimation, ARcore, floor plans**

I. Introduction

The generation of 2D layout from indoor images is an essential task with several applications. However, determining an actual plan is a challenging job as it requires accuracy in terms of depth data from the RGB images, and point cloud with real-world scaling. The state-of-the-art (see Sec. II), and the commercial applications for layout estimation, uses RGB-D images or panorama images to capture the layout. Notwithstanding good accuracy, these methods require specialized hardware (depth camera) or a particular mode of photo capture (panorama with little to no occlusion), which restricts widespread adoption. In this work, we proposed LayART (Layout estimation using ARcore Transformation), a framework to estimate the 2D layout of an indoor scene using RGB images captured through a mobile camera. Figure 1 depicts the potential partial input RGB images of indoor scenes and their corresponding layout of the entire room. ARcore uses SLAM in the background and capable of localizing the device in real-world coordinates. In LayART, the user needs to take 4 pictures of the indoor scenes using our developed application. The depth map of the scenes are extracted from these pictures adapting the method in [1] followed by edge map extraction using [2]. The RGB images and depth maps are used for 3D reconstruction. Then the point clouds and edge maps are mapped in 2D for the final layout using a novel regularization technique. LayART takes advantage of Google's ARCore library for pose estimation, thus saving computational time.

In the proposed framework (see Fig. 2), we have made the following contributions. Firstly, LayART performs 2D estimation from RGB monocular images, without using panorama or 360° spherical images. Secondly, it utilizes ARcore pose data for 3D reconstruction of scene. Finally, it generates a near accurate layout estimation with fewer number of images captured through camera, as compared to the existing techniques. Currently we are assuming only Manhattan and weak Manhattan scenes. LayART (see Sec. III) compares well with respect to the state-of-the-art (see Sec. IV) for generating layout and measuring the dimensions. Also, it works well in case of occlusion in the indoor scene where existing methods fail.

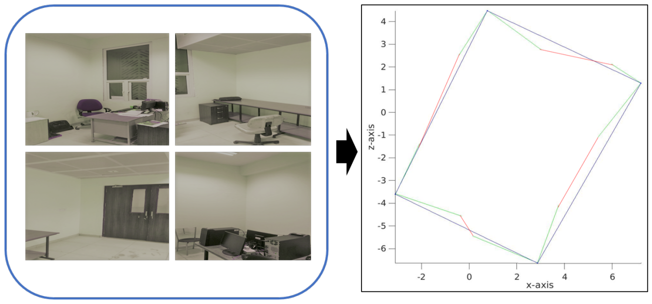

Figure 1. An illustration of LayART. (Left) A set of room images captured using a mobile camera is the input. (Right) The resultant room layout.

II. Related Work

Layout estimation from RGB-D and panorama images has been a widely explored problem. The authors in [3], [4] have reconstructed the indoor scene in 3D using monocular images and estimated layout using vanishing points and depth features. In [5], authors have estimated layouts on RGB images by using an encoder-decoder framework to jointly learn the edge maps and semantic labels of each image. In [6], room layout is generated from images taken from multiple views and reconstructed using SfM and region classification. In [7], layout estimation was performed in cluttered indoor scene by identifying label for pixel from RGB images, using deep FCNN and refined using geometrical techniques. Since monocular images can not capture the entire scene, layout estimation from panorama images to increase the field of views has been explored in the literature. In recent work, Zou et. al [2] has proposed an encoder-decoder network which predicts boundary and corner maps and optimized over constrained geometrical properties of the room. Sun et al. [8] estimated room layout by regressing over boundaries and classifying the corner

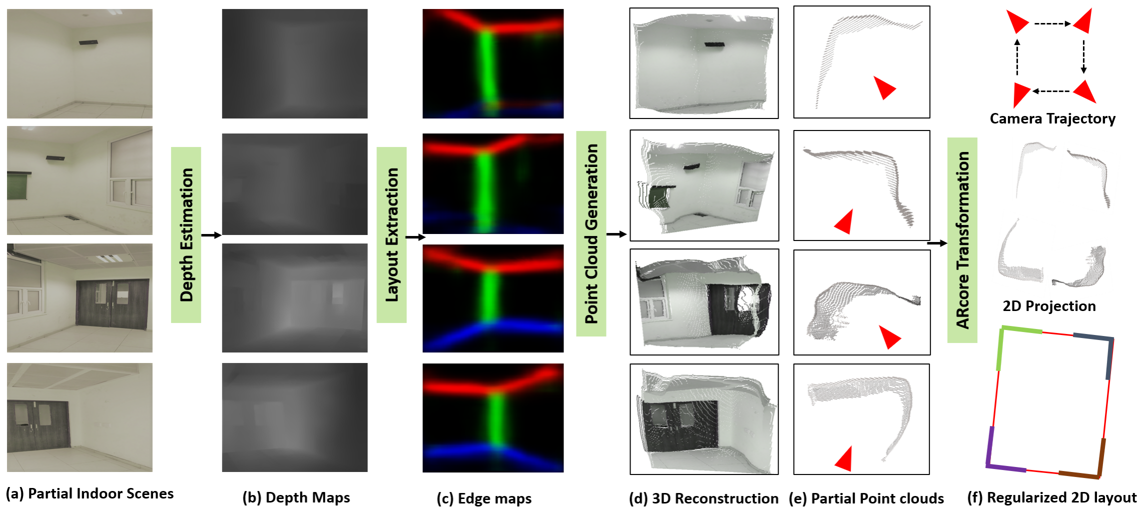

Figure 2. Pictorial depiction of the intermediate results of the various stages of the processing pipeline of LayART.

for each column representation of image. Also in [9], 3D layout was generated for 360^o panorama image by reasoning between geometry and edge maps returned by a deep neural network. An approach to generate indoor scene layout by learning the encoded layout representation in row vectors, was proposed in [10]. In [11] [12], layout is obtained from panorama image by estimating object locations and pose in the room. Indoor layout has also been generated from monocular video sequences in [13], where SLAM/SfM techniques are used for 3D reconstruction and layout is generated by fitting the planes. In another work [14], authors have generated 2D layout of the room by using depth data and 3D reconstruction using SLAM. In the context of floor plan generation, authors of [15] has reconstructed the layout from the input panorama images using SfM and from that generated a 2D floor plan by posing it as a shortest path problem. Also, Lin et. al [16] predicts the global room layout and transformations using partial reconstructions of indoor scenes using RGB-D images without making use of feature matching between partial scans.

III. Description of the Framework

Related work shows that 3D reconstruction of a room and layout estimation requires ample amount of hardware such as depth cameras, Kinect cameras, LiDAR. Additionally, the existing techniques require human intervention at several stages. Although some methods exist for layout generation from monocular images [2], they rely on occlusion-free panoramic photos, which are almost impossible to take for regular spaces such as office or home. Our method attempts to address this issue by taking advantage of recent mobile Augmented Reality libraries for pose estimation. ARcore [17] is such a library by Google, which uses the phone's IMU sensor's data along with image feature points for tracking the motion (pose and orientation) of the camera. In LayART, to track the motion of the camera, an android application using ARcore was developed in Unity3D environment for capturing RGB images along with real world location of the device. For our experiments we used Google Pixel 2 XL as the device for capturing images and tracking camera motion. As shown in Fig. 2(a), a set of RGB images are taken for each partial scene. Depth estimation is done individually for each RGB image and layout is estimated by adapting the network proposed in [2] for perspective images. Mapping the depth maps with RGB images and edge maps of layouts, the point clouds are generated and layout is extracted from the point clouds. While RGB images were clicked, ARcore stored the location of the camera in the background and this camera trajectory is utilized for transforming each partial scene point cloud after projecting them into a 2D plane. Each partial point cloud is regularized locally and globally to generate the layout.

A. Depth Estimation

Figure 2 depicts overall framework of LayART. For depth perception from RGB images, a set of images of the same scene from multiple views with a calibrated camera is required. An easier alternative is to use machine learning. Depth for RGB images can be learnt from ground truth depth-maps and a trained model can be used for estimating depth for new images. We have adapted the method proposed in [1], where an encoder-decoder architecture is used after extracting image features with DenseNet-169, resulting in high-resolution depth-maps. In Fig. 2(b), the depth maps generated for each partial scene RGB image is shown.

B. Layout Estimation

Edge and boundary maps of a scene is a requirement for layout estimation. In our work, we have adapted the technique proposed in [2] to identify the edge/boundary maps of the scene. Figure 2(c) shows the edge maps generated for each partial scene. These edge maps are then used to identify the boundary pixels in the point clouds in later stages.

C. 3D Reconstruction

To generate a layout of the entire scene, 3D reconstruction of each partial scene is required. In LayART, we have tried to suppress the requirement of additional hardware for this task. We mapped every pixel of an RGB image with depth maps generated in the previous steps to generate a point cloud for the scene. This step is preceded by a camera calibration step to identify intrinsic parameters of the phone's camera, such as focal length (f), center coordinates (C_x, C_y). Here, coordinates in the point cloud are calculated by:

$$Z = \frac{D_{u,v}}{S} \tag{1}$$

$$X = \frac{(u - C_x) * Z}{f} \tag{2}$$

$$Y = \frac{(v - C_y) * Z}{f} \tag{3}$$

where, X, Y, Z are coordinates of the real world and Z is the depth value. $D_{u,v}$ is the depth value corresponding to the (u, v) pixel in the image in the depth map. S is the scaling factor of the scene (obtained empirically) and comparing dimensions of real world objects and point clouds. f, C_x, C_y are the intrinsic parameters of the camera, generated by calibration. This step yields the 3D reconstruction of each partial scene in the real world scale (Fig. 2(d)).

D. Point cloud transformation and regularization

The generated point cloud in previous step are then mapped with the edge maps generated to identify the boundary pixels in the point cloud (Fig. 2(e)). These point clouds are scattered 3D points of the layout and required to be regularized in order to reduce the error in the geometry of the generated 2D layout. Each red marker is the location of the camera for each scene, tracked by ARcore. Camera trajectory for the entire scene is tracked by ARcore as shown in the top panel of Fig. 2(f).

Algorithm 1 regularizes the local point cloud of each partial scene image. Here, P_i is the point cloud of each i-th scene, n is the total number of point clouds. Boundary points for each P_i is extracted in $P_i(K)$. Using the k-means algorithm, clusters of point set is made for $k = 3$ on the basis of the Euclidean distance between them, where m_1, m_2, m_3 are the cluster means (line 5). Since we are assuming Manhattan world for the scene, the lines joining means are re-adjusted to have a right angle between them (line 9). Each regularized local point cloud (RP_i) is transformed (TP_i) using rotation angle $\theta_x, \theta_y, \theta_z$, along each x, y, z axis and translation coordinates $[t_x, t_y]$ returned by ARcore (line 12). For global regularization, using each transformed point

Algorithm 1 Regularize point clouds (PC)

```
1: for $i = 1 : n$ do                                          ▷ $n$: no of PC
2:     $P_i = 2DPointClouds$
3:     $K = boundary(P_i)$
4:     $C(c_1, c_2, ..., c_k) = kmeans(P_i(K))$                  ▷ $C$: Clusters
5:     $m_1, m_2, m_3 = mean(c_1), mean(c_2), mean(c_3)$
6:     $line_1 = line(m_1, m_2)$
7:     $line_2 = line(m_2, m_3)$
8:     while $angle(line_1, line_2) <= 90$ do
9:         $Rotate(line_2)$
10:    end while
11:    $RP_i = (line_1, line_2)$                                ▷ $RP_i$: local PC
12:    $TP_i = (Rot(\theta_x, \theta_y, \theta_z) * Tr(t_x, t_y)) * RP_i$
13: end for
14: $FP = polygon(TP_1, TP_2, ..., TP_n)$                       ▷ $FP$: Final PC
15: for $i = 1 : p$ do                          ▷ $p$: no of sides of polygon
16:     $\phi = angle(s_i, s_{i+1})$                    ▷ $s$: sides of polygon
17:     if $\phi > 90$ or $\phi < 90$ then
18:         $\phi(s_i, s_{i+1}) = 0$
19:     end if
20: end for
```

cloud, polygon (FP) is formed (line 14), with p number of sides (s). For each pair of sides, the angle between them (ϕ) is checked and if they are not perpendicular, they are made colinear (line 18) assuming the world to be Manhattan.

Figure 3 depicts the process of transformation of partial point clouds and regularization of global layout. Figure. 3 (a) shows the coordinate system in real world (X_W, Y_W, Z_W) and in ARcore with the mobile device (X_A, Y_A, Z_A). To align the coordinates in both coordinate systems, ARcore transformations have to be rotated about Z_A axis. Each partial 2D point set is then rotated and translated with the transformation given by ARcore (Fig. 3 (b)). Figure 3(c) shows the globally regularized 2D point set for the partial point clouds and it agrees with the real world dimensions.

IV. Experiments

For all the experiments Google Pixel 2 XL was used as a mobile phone to deploy the app and capture a set of images. Table I depicts the quantitative evaluation for the estimated

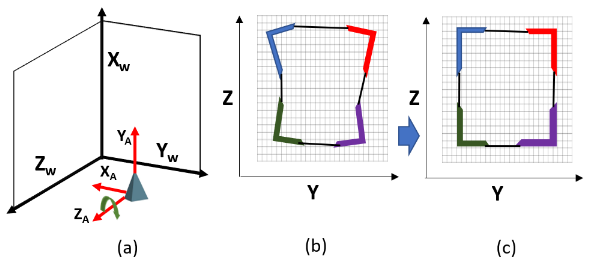

Figure 3. An illustration of translation and global regularization.

Table I
QUANTITATIVE EVALUATION OF THE ESTIMATED LAYOUTS FOR DIFFERENT SCENES(S) AND METHODS(M)

S / M	**Lab_1**				**Lab_2**				**ClassRoom**				**MeetingRoom**			
GT	$86.48m^2$		1.48		$68.37m^2$		1.87		$72.54m^2$		1.31		$55.92m^2$		1.12	
	Area	E (%)	Asp	E (%)	Area	E (%)	Asp	E (%)	Area	E (%)	Asp	E (%)	Area	E (%)	Asp	E (%)
Ours	80.60	6.8	1.45	2.0	63.75	6.7	1.94	3.7	**68.09**	**6.1**	**1.35**	**3.0**	**54.64**	**2.3**	1.15	2.6
MP	**86.08**	**0.4**	**1.49**	**0.7**	73.03	6.8	1.73	7.5	64.80	10.6	1.26	3.8	54.09	3.3	1.17	4.5
TM	81.40	5.9	1.49	0.7	64.90	5.1	**1.86**	**0.5**	61.64	15.0	1.37	4.5	53.13	4.9	**1.11**	**0.9**
iPhM	81.94	5.2	1.46	1.3	69.03	0.9	2.08	11.2	60.84	16.1	1.35	3.0	52.95	5.3	1.13	0.9
AR3d	86.03	0.5	1.49	0.7	**68.80**	**0.6**	1.81	3.2	64.62	10.9	1.37	4.5	54.65	2.3	1.13	0.9

layout for the different scenes and different applications of Android and iOS. The experiments are done on 4 indoor scene, where all the scenes are in the Manhattan world. In this experimental setup, Lab_1 and Lab_2 are scenes with corners occluded and heavily cluttered. $MeetingRoom$ had two corners occluded and medium cluttered, while $ClassRoom$ has all the corners visible and with low clutter environment. Comparative study was performed with applications such as Magic Plan (MP) [18], Tape Measure (TM) [19], iPhone Measure (iPhM) [20], and AR Plan3D (AR3d) [21] with the given ground truth (GT) measurements. The two metrics used to measure the accuracy were area and aspect ratio (Asp). It can be seen that LayART is performing best in terms of percentage error ($E = (GT - X)/GT \times 100$) in the area for the Classroom and Meeting Room. Here X is the estimated value for a given method. In terms of aspect ratio, it is performing best for the Classroom. For Lab_1, Magic Plan is performing best in area and aspect ratio, and for Lab_2, AR Plan3D is performing the best. LayART is performing worst in case of Lab_1 due to a heavily cluttered environment, but still, $E < 7\%$ for all the indoor scenes.

Table II depicts the qualitative comparison between the proposed method and other applications. Here, the number of user interactions and the amount of manual intervention required were considered as the basis of comparison. In terms of the number of user interactions, our method requires only 4 interactions, i.e., images of 4 corners of a room. In contrast, other state-of-the-art methods require a continuous scan and movement in the entire room. In terms of manual intervention, LayART does not require any, after clicking the pictures. Whereas, the other applications require manually adding the corners and height of the room. Also, the only requirement for LayART is to click images, while other applications considered for comparison take some time and manual calibration to understand the environment and features. Due to this continuous scanning and more manual intervention, techniques like [18] yields a more accurate result than ours. However, in the existing apps, manual correction of the corners are required for occlusion. User error in the existing methods can profoundly affect the accuracy of the resultant layout. The accuracy of the existing apps also suffers in case of limited salient features in different frames of the scene while scanning.

Some existing methods in the literature, such as [2], are also used for estimating the entire floor plan for the room. However, [2] works on the panoramic image of the scene. As compared to LayART, [2] require more information to generate a layout. LayART works on non-overlapping images of the room which do not have any shared features. Due to the lack of any available dataset for partial scene images, we could not test our method on any publicly available datasets. We tested LayART on our dataset and performed experiments by comparing it with a few existing state-of-the-art applications under the same experimental condition to ensure fairness. LayART can generate reasonably accurate layout in terms of the error in area, aspect ratio, while requiring far less user interaction and intervention.

Table II
RESULTS OF QUALITATIVE COMPARISON

Method	**User Interaction**	**Manual Intervention**
Ours	4 Nos.	Not required
MP	Continuous Scan	Add corners
TM	Continuous Scan	Add corners
iPhM	Continuous Scan	Add corners
AR3d	Continuous Scan	Add corners, height

V. CONCLUSION

In this paper, we propose LayART to generate 2D room layout using RGB images taken from mobile phone's camera and camera pose data given by Google ARcore. A mobile application is developed to facilitate data acquisition. The estimated layout agrees well in terms of real world dimensions. LayART works well for cluttered indoor setting and occluded walls and corners. LayART require lesser number of user interactions and no manual intervention, as compared to the state-of-the-art. In future, we are planning to estimate the layout for the entire floor by relaxing the Manhattan world assumption and for more generalised scene.

ACKNOWLEDGEMENT

This work is partially supported by Science and Engineering Research Board, India under the project id ECR/2016/000953 and the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number CRDPJ530666-18.

References

[1] Ibraheem Alhashim and Peter Wonka, "High quality monocular depth estimation via transfer learning," *arXiv preprint arXiv:1812.11941*, 2018.

[2] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem, "Layoutnet: Reconstructing the 3d room layout from a single rgb image," in *CVPR*, 2018, pp. 2051–2059.

[3] Chenxi Liu, Alexander G Schwing, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler, "Rent3d: Floor-plan priors for monocular layout estimation," in *CVPR*, 2015, pp. 3413–3421.

[4] Jian Zhang, Chen Kan, Alexander G Schwing, and Raquel Urtasun, "Estimating the 3d layout of indoor scenes and its clutter from depth sensors," in *ICCV*, 2013, pp. 1273–1280.

[5] Weidong Zhang, Wei Zhang, and Jason Gu, "Edge-semantic learning strategy for layout estimation in indoor environment," *IEEE-T on cybernetics*, 2019.

[6] Sid Yingze Bao, Axel Furlan, Li Fei-Fei, and Silvio Savarese, "Understanding the 3d layout of a cluttered room from multiple images," in *WACV*. IEEE, 2014, pp. 690–697.

[7] Saumitro Dasgupta, Kuan Fang, Kevin Chen, and Silvio Savarese, "Delay: Robust spatial layout estimation for cluttered indoor scenes," in *CVPR*, 2016, pp. 616–624.

[8] Cheng Sun, Chi-Wei Hsiao, Min Sun, and Hwann-Tzong Chen, "Horizonnet: Learning room layout with 1d representation and pano stretch data augmentation," in *CVPR*, 2019, pp. 1047–1056.

[9] Clara Fernandez-Labrador, Alejandro Perez-Yus, Gonzalo Lopez-Nicolas, and Jose J Guerrero, "Layouts from panoramic images with geometry and deep learning," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3153–3160, 2018.

[10] Chi-Wei Hsiao, Cheng Sun, Min Sun, and Hwann-Tzong Chen, "Flat2layout: Flat representation for estimating layout of general room types," *arXiv preprint arXiv:1905.12571*, 2019.

[11] Jiu Xu, Björn Stenger, Tommi Kerola, and Tony Tung, "Pano2cad: Room layout from a single panorama image," in *WACV*. IEEE, 2017, pp. 354–362.

[12] Yinda Zhang, Shuran Song, Ping Tan, and Jianxiong Xiao, "Panocontext: A whole-room 3d context model for panoramic scene understanding," in *ECCV*. Springer, 2014, pp. 668–686.

[13] Axel Furlan, Stephen D Miller, Domenico G Sorrenti, Fei-Fei Li, and Silvio Savarese, "Free your camera: 3d indoor scene understanding from arbitrary camera motion.," in *BMVC*, 2013.

[14] Vincent Angladon, *Room layout estimation on mobile devices*, Ph.D. thesis, 2018.

[15] Ricardo Cabral and Yasutaka Furukawa, "Piecewise planar and compact floorplan reconstruction from images," in *CVPR*. IEEE, 2014, pp. 628–635.

[16] Cheng Lin, Changjian Li, Yasutaka Furukawa, and Wenping Wang, "Floorplan priors for joint camera pose and room layout estimation," *arXiv preprint arXiv:1812.06677*, 2018.

[17] "ARcore, fundamental concepts," https://developers.google.com/ar/discover/concepts.

[18] "Sensopia. MagicPlan: Create a floor plan in just a few minutes," https://www.magicplan.app/magicplan/.

[19] "Occipital. The fastest way to measure (iOS application)," https://tapmeasure.io/.

[20] "Use the Measure app on your iPhone, iPad, or iPod touch," https://support.apple.com/en-us/HT208924.

[21] "AR Plan 3D Ruler – Camera to Plan, Floorplanner," https://arplan-3d.en.aptoide.com/.