Sequential Fusion of Multi-view Video Frames for 3D Scene Generation

Weilin Sun, Xiangxian Li, Manyi Li, Yuqing Wang, Yuze Zheng, Xiangxu Meng, and Lei Meng*

Shandong University, Jinan, Shandong, China
swl_lynn2000@163.com xiangxian_lee@mail.sdu.edu.cn manyili@sdu.edu.cn
wang_yuqing@mail.sdu.edu.cn zhengyuze@mail.sdu.edu.cn mxx@sdu.edu.cn
lmeng@sdu.edu.cn

Abstract. 3D scene understanding and generation are to reconstruct the layout of the scene and each object from an RGB image, estimate its semantic type in 3D space and generate a 3D scene. At present, the 3D scene generation algorithm based on deep learning mainly recovers the 3D scene from a single image. Due to the complexity of the real environment, the information provided by a single image is limited, and there are problems such as the lack of single-view information and the occlusion of objects in the scene. In response to the above problems, we propose a 3D scene generation framework SGMT, which realizes multi-view position information fusion and reconstructs the 3D scene from multi-view video time series data to compensate for the missing object position in existing methods. We demonstrated the effectiveness of multi-view scene generation of SGMT on the UrbanScene3D and SUNRGBD dataset and studied the influence of SGCN and joint fine-tuning. In addition, we further explored the transfer ability of the SGMT between datasets and discussed future improvements.

Keywords: 3D scene generation · Multi-view fusion · Multi-view time series data.

1 INTRODUCTION

3D scene generation is an important task in computer vision, which has a great impact on many fields like augmented reality and virtual reality. The main idea of the traditional 3D scene construction method is to manually process and fuse the visual information, and reconstruct a 3D scene by scene rendering which has high time and labor costs. To alleviate the above problems, end-to-end deep learning methods are introduced into 3D scene generation, which avoids complex manual processing through a data-driven manner.

The methods based on deep learning mainly divide the 3D scene generation task into three sub-tasks: layout estimation, object detection, and shape recovery. Early works completed the three sub tasks separately[15, 19, 1]. Total3d[17]

* Corresponding author

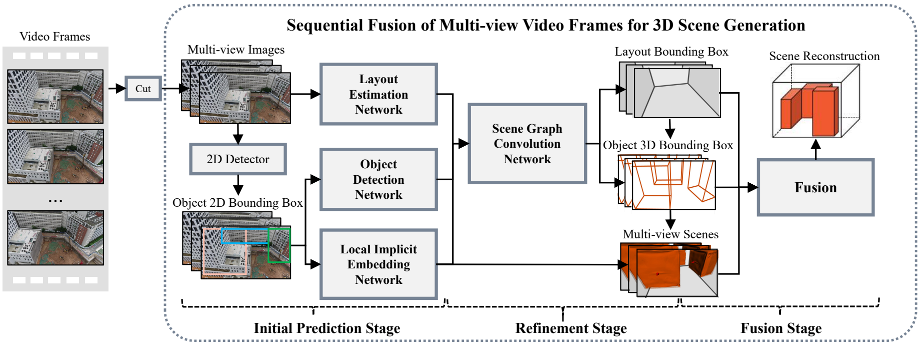


Fig. 1. The schematic diagram of SGMT, obtaining multi-view time series data from video frames as input, and finally realizing 3D scene recovery.

bridged the gap between these three tasks and restored the 3D scene from the perspective of the overall scene. On this basis, follow-up studies have proposed solutions to improve the accuracy of overall 3D scene restoration[23, 24]. However, most of the existing methods recover 3D scenes from a single image. Due to the complexity of the real environment, the information provided by a single image is limited, and there are some problems such as the lack of single-view information and the occlusion of objects in the scene.

To alleviate the aforementioned problems, we study to decouple and reorganize the existing deep learning-based 3D scene generation methods and explore the key factors affecting the performance. On this basis, we proposed a 3D scene generation framework SGMT, which recovers the overall 3D scene and compensates for the multi-view scene generation by fusing multi-angle position information. The overall framework of the model is shown in Fig. 1, which is mainly divided into three stages: the initial prediction stage, the refinement stage and the fusion stage. In the initial prediction stage, the geometric information in the visual input is extracted through the layout estimation network(LEN), the object detection network(ODN) and the local implicit embedding network(LIEN), and the initial prediction of the layout box, object box and object grid is realized.In the refinement stage, the scene graph convolution network(SGCN) is used to update the layout and object features and the refinement of the initial results is completed. In the fusion stage, the translation, rotation and fusion of the results from different perspectives are realized, so that the position information of the object can be adjusted and supplemented.

In order to explore the influence of the refinement stage in the proposed framework, we design comparative experiments to demonstrate its effectiveness in improving the generation result from both qualitative and quantitative perspectives. Further, we compare the result in the scene dataset SUNRGBD and UrbanScene3D multi-view video data, analyze it from the aspects of geometry and appearance, and discuss the transfer ability of 3D scene generation model and the problems in the transfer process in depth.

In summary, the main contributions of this paper include:

- On the basis of cutting-edge work in deep learning-based 3D scene reconstruction, we propose a multi-view 3D scene generation framework SGMT, which realizes the conversion from multi-view 2D video data to 3D scenes.
- The effectiveness of SGCN and joint fine-tuning in improving model performance is analyzed and verified, and the transfer ability of the model and the key problems in the process of model transfer are discussed.

2 RELATED WORKS

Layout estimation, object detection, and shape recovery are important components of 3D scene generation algorithms. **Layout Estimation.** Layout estimation can be divided into two types. One is to obtain the feature map of the layout based on the neural network, and then generate the parametric representation[15, 19, 2]. The other is a deep learning end-to-end method[11, 12, 5, 6], which treats the layout estimation task as a regression of keypoints or a classification of spatial layout types, improving the accuracy of layout estimation. **Object Detection.** Object detection includes 2D object detection and 3D object detection. 2D object detection is to detect 2D bounding boxes and category information of objects in 2D images, such as Faster-RCNN[18] and YOLO series algorithms. 3D object detection often predicts the 3D bounding box based on the 2D bounding box[7], so as to obtain the information such as the length, width, height, offset angle and 3D space position of the object in the real 3D scene. **Shape Recovery.** Previous works of shape recovery have attempted to use point clouds and voxels to represent the 3D target object[1, 10], or used retrieval methods to search for similar-looking models from the dataset[8]. The reconstruction results of these method have lower resolution and consume more memory. In order to alleviate the above problems, more methods begin to exploit the prior knowledge of shape, express the shape of an object as a feature vector or an implicit function, and finally recover its shape[16, 4, 3].

The above methods only consider independent geometries. In order to understand and reconstruct the scene from an overall perspective, a method of fusing the contextual information of the scene has emerged[17]. At the same time, the graph convolutional neural network is added to refine the model[23], and the structural implicit network is further used to improve the shape estimation of the object[24], which has become the most advanced method at present.

3 METHOD

The overall algorithm flow of SGMT is shown in Fig. 2, which includes five modules. We divide them into three stages, namely the initial prediction stage, the refinement stage, and the fusion stage. Their details are described below.

3.1 Initial Prediction Stage

The initial prediction stage adopts LEN, ODN, and LIEN in [17, 24].

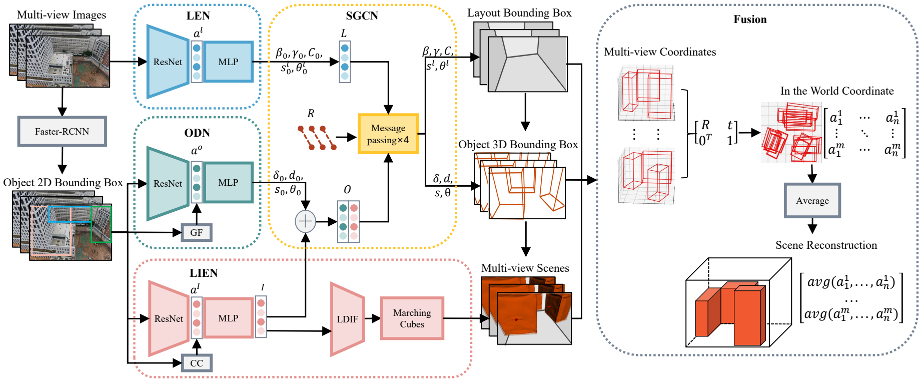

Multi-view Images
LEN
ResNet
a^l
MLP
SGCN
$\beta_0, \gamma_0, C_0, s_0^l, \theta_0^l$
L
$\beta, \gamma, C, s^l, \theta^l$
Layout Bounding Box
Faster-RCNN
R
Message passing×4
ODN
a^o
ResNet
MLP
Object 3D Bounding Box
Object 2D Bounding Box
$\delta_0, d_0, s_0, \theta_0$
O
δ, d, s, θ
GF
LIEN
a^l
ResNet
MLP
I
LDIF
Marching Cubes
Multi-view Scenes
CC
Fusion
Multi-view Coordinates
$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$
In the World Coordinate
$\begin{bmatrix} a_1^1 & \cdots & a_n^1 \\ \vdots & \ddots & \vdots \\ a_1^m & \cdots & a_n^m \end{bmatrix}$
Average
Scene Reconstruction
$\begin{bmatrix} avg(a_1^1, \ldots, a_n^1) \\ \cdots \\ avg(a_1^m, \ldots, a_n^m) \end{bmatrix}$

Fig. 2. Illustration of the algorithm flow of SGMT, enabling 3D scene recovery from multi-view time series data.

Layout Estimation Network LEN is used for the initial prediction of the layout 3D bounding box $X_0^L \in \mathbb{R}^3$ and camera pose $R(\beta_0, \gamma_0) \in \mathbb{R}^2$, which is further parameterized by the method of [17, 24] as,

$$X_0^L = h(C_0, s_0^l, \theta_0^l) \tag{1}$$

where $C_0 \in \mathbb{R}^3$ is the center of the layout box, $s_0^l \in \mathbb{R}^3$ is the space size, $\theta_0^l \in (-\pi, \pi)$ is the direction angle, and $h(\cdot)$ is the function that composes the 3D bounding box. The algorithm flow of LEN is shown in Fig. 2. First, the ResNet is used to extract the appearance features a_l, and then the two-layer MLP is used to predict $(\beta_0, \gamma_0, C_0, s_0^l, \theta_0^l)$.

Object Detection Network ODN can predict 3D bounding boxes $X_0^O \in \mathbb{R}^3$ from 2D bounding boxes of objects. Using the method of [17, 24], it is further parameterized as,

$$X_0^O = h(\delta_0, d_0, s_0, \theta_0) \tag{2}$$

where $\delta_0 \in \mathbb{R}^2$ is the offset between the center of the 2D bounding box and the 2D projection center of the 3D bounding box, $d_0 \in \mathbb{R}$ is the distance from the 2D projection center of the 3D bounding box to the center of the camera, $s_0 \in \mathbb{R}^3$ is the space size of the object, $\theta_0 \in (-\pi, \pi)$ is the orientation angle of the object. The algorithm flow of ODN is shown in Fig. 2. The appearance feature a_o is extracted from the 2D bounding box using ResNet, at the same time, the size and relative position of each object 2D bounding box are encoded as geometric features GF. GF and a_o are input into a two-layer MLP to predict $(\delta_0, d_0, s_0, \theta_0)$ of the object.

Local Implicit Embedding Network LIEN is used to to recover the shape and pose of objects. The algorithm flow of this module is shown in Fig. 2. First, we input the 2D bounding box of the object to ResNet to extract the appearance feature a_I. In order to effectively learn the implicit shape features, the class code

CC of the object is concatenated with its appearance feature a_I, and then the latent code I is obtained using a three-layer MLP. I is input into LDIF to obtain a 3D latent shape representation. Finally, we use the Marching Cubes[14] to get the point and surface information of the object.

3.2 Refinement Stage

In the refinement stage, SGCN is added to update the layout, object and relation nodes in the scene graph through the process of four message passing[24], as shown in the Fig. 2. We use the results of the initial stage to extract the feature vectors of layout and object nodes and then process them into feature matrices $M^o \in \mathbb{R}^{d\times(N+1)}$. The relation nodes are divided into two categories, one represents the relationship between the layout and the object, which is initialized with a constant value, and the other represents the relationship between the objects, which is initialized using the GT and bounding box coordinates, and then they are processed as feature matrices $M^r \in \mathbb{R}^{d\times(N+1)^2}$. The process of message passing can be expressed as

$$M^{o'} = \sigma(M^o + W^{sd}M^o + W^{rs}M^rA^{rs} + W^{rd}M^rA^{rd}) \tag{3}$$

$$M^{r'} = \sigma(M^r + W^{sr}M^oA^{sr} + W^{dr}M^oA^{dr}) \tag{4}$$

where s is the source object/layout node; d is the target object/layout node; r is the relation node; W and A are the linear transformation and adjacency matrix from the source node to the target node.

3.3 Fusion Stage

Multi-view scene fusion refers to the process of fusing objects from different perspectives into one scene. This process can be regarded as the transformation of the coordinate system of objects in the scene. Compared with generating a 3D scene from a single image, multi-view scene generation involves not only the rotation of the camera, but also but also the translation of the camera. For a point a on the object, the coordinate before transformation is $(a_1, a_2, a_3)^T$, the coordinate after transformation is $(a_1^{'}, a_2^{'}, a_3^{'})^T$, and the rigid body transformation formula is

$$\begin{bmatrix} a_1^{'} \\ a_2^{'} \\ a_3^{'} \end{bmatrix} = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{5}$$

where $R \in \mathbb{R}^{3\times3}$ is the rotation matrix of the camera, which can be predicted by LEN, and $t \in \mathbb{R}^{3\times1}$ is the translation matrix, which can be obtained by the aerial photography trajectory of the camera. We use the first scene graph as the world coordinate system, transform the 3D coordinates of objects from other perspectives into it, and then average the eight corner coordinates of the same object from different perspectives. Finally, the fusion of object position information is realized. The whole fusion process is shown in Fig. 2.

3.4 Loss Function

We refer to the method of [17, 24] to define the loss function of the model in modules. When training LEN and ODN, we use classification and regression loss for every output parameter of LEN and ODN. When training LIEN, we weight shape element center loss L_c[3] and point sample loss L_p[24] to sum. In the initial prediction stage, the loss function is mainly optimized for the 3D bounding box parameters of the object and the layout, but not for the final prediction result. Therefore, when training SGCN, the cooperation loss L_{co}[7] is added, and the formula is as follows:

$$L_{co} = \lambda_{phy} L_{phy} + \lambda_{bdb2D} L_{bdb2D} + \lambda_{corner} L_{corner} \tag{6}$$

where L_{phy} is the mean square error loss, which is used to reduce the intersection between the layout and the object bounding box; L_{bdb2D} and L_{corner} are SmoothL1Loss, which is used to reduce the error of the 3D bounding box of the object and its 2D projection. In addition to L_{co}, $L_{ldifphy}$[24] is also added in the joint fine-tuning to reduce the crossover between objects. The formula is as follows:

$$L_{ldifphy} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|\mathbb{S}_i|} \sum_{x \in \mathbb{P}_i} ||relu(0.5 - sigmoid(\alpha LDIF_i(x)))|| \tag{7}$$

where N is the number of objects in the scene; $\mathbb{P}_i$ is the sampling point from each object; $\alpha LDIF_i(x)$ is the value of the obtained point on the LDIF decoder, and has been scaled by α. After the sigmoid and relu activation functions, the loss function only considers the points that intersect inside the object, that is, the points where $\alpha LDIF_i(x)$ is negative. Finally, we can get the loss function of the whole model, the formula is as follows:

$$L_{joint} = L_{LEN} + L_{ODN} + L_{co} + L_{ldifphy} \tag{8}$$

4 EXPERIMENTS

4.1 Experiments Setup

Datasets We use SUNRGBD[21, 20, 9, 22] and UrbanScene3D[13] for model training, testing and transfer. The SUNRGBD dataset contains 10,335 RGBD images of indoor scenes, of which the 1-5050th images of the dataset are used for validation and testing; the 5051-10,335th images are used for training. The UrbanScene3D dataset contains 5 reconstructed real scenes. We intercepted the multi-view pictures of the scene from the aerial video of the Sci-Art, and got a total of 341 pictures. At the same time, annotations are established for each image, including the coordinates of the 2D bounding box of the object, the object category, the camera internal parameters, and the image ID. The processed UrbanScene3D dataset is used as multi-view time series data to complete the model testing and transfer.

Evaluation Measures We use the mean of IoU to evaluate the accuracy of layout and object bounding box, the mean of camera radian error to evaluate the accuracy of camera pose, and Lg [17] to evaluate the accuracy of object triangular mesh. The formula is as follows:

$$IoU(G, E) = \frac{|G \cup E|}{|G \cap E|} \tag{9}$$

where G is the actual layout/object bounding box and E is the predicted layout/object bounding box;

$$Cam_{Err} = |\theta_g - \theta_e| \times \frac{180}{\pi} \tag{10}$$

where θ_g is the actual camera rotation angle, and θ_e is the predicted camera rotation angle;

$$Lg = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|\mathbb{S}_i|} \sum_{q \in \mathbb{S}_i, p \in \mathbb{M}_i} min||p - q||_2^2 \tag{11}$$

where N is the number of objects in the scene, q is a point on the ground-truth surface $\mathbb{S}_i$, p is a point on the predicted object grid $\mathbb{M}_i$, and $||p-q||_2^2$ represents the distance between the two points. To sum up, we set six evaluation metrics: $LayoutIoU$, $CamPitchErr$, $CamRollErr$, $Box3DIoU$, $Box2DIoU$, and Lg.

Implementation Details For LEN, ODN and LIEN, we use pre-trained weight parameters[17, 24]. SGCN is trained on the SUNRGBD dataset, using 30 epochs in total and Adam optimizer with a batch size of 32 and learning rate decaying from 2e-4 (scaled by 0.5 when the epoch reaches 18, 23, 28). When training SGCN individually, we use L_{joint} without $L_{ldifphy}$, and put it into the full model with pre-trained weights of other modules. Joint fine-tuning is similar to the training setup of SGCN, except that the batch size is 4, the learning rate decays from 1e-4 and L_{joint} with $L_{ldifphy}$ is used.

4.2 Comparative experiment

In this section, we are going to analyse the effects of SGCN and joint fine-tuning in the refinement stage on improving SGMT performance from both quantitative and qualitative perspectives.

Quantitative analysis Six evaluation metrics are used to evaluate the performance change of the model before and after refinement and fine-tuning. As is shown in Table 1, from initial prediction to SGCN and then to joint fine-tuning, $LayoutIoU$, $Box3DIoU$ and $Box2DIoU$ are all improved, while $CamPitchErr$, $CamRollErr$ and Lg all decreases. This indicates that the performance of 3D scene generation improves, and possible reasons are as follows:

Table 1. Comparison of the SGMT performance before and after SGCN and joint fine-tuning in the SUNRGBD dataset. IP means Initial Prediction.

Evaluation Metrics	IP	IP+SGCN	IP+SGCN+Joint Fine-tuning
$LayoutIoU$	0.61854	0.63649	0.67800
$CamPitchErr$	3.98966	3.04301	2.49251
$CamRollErr$	2.71317	2.18722	2.13512
$Box3DIoU$	0.13635	0.18991	0.29596
$Box2DIoU$	0.63158	0.67418	0.75534
Lg	1.20858	1.13047	1.10760

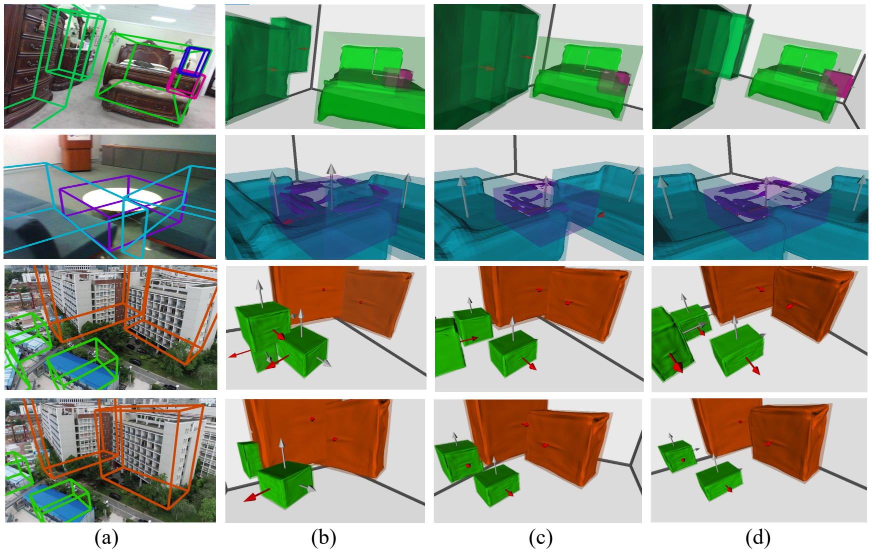


Fig. 3. Comparison of model results before and after SGCN and joint fine-tuning: (a) input image and recognized objects; (b) initial prediction results; (c) results after SGCN; (d) results after joint fine-tuning.

(1) SGCN integrates scene context information, acquires important scene knowledge, and updates the features of objects and layouts, making the results more accurate. At the same time, L_{co} is added when training SGCN, which maintains the consistency between the 2D and 3D bounding boxes, and improves the accuracy of the model.

(2) During joint fine-tuning, some models that are frozen during SGCN training are unfrozen, and the weight parameters of the overall model are updated. At the same time, $L_{ldifphy}$ is added to reduce the intersection between objects, which further improves the accuracy of the model.

Qualitative analysis According to the horizontal comparison of the results in Fig. 3, it can be found that: 1) The 3D reconstruction ability of initial prediction is relatively poor. As is shown in Fig. 3(b), although the approximate position and shape of objects in the scene can be predicted, there are lots of problems. For example, there are improper intersections between objects, between objects

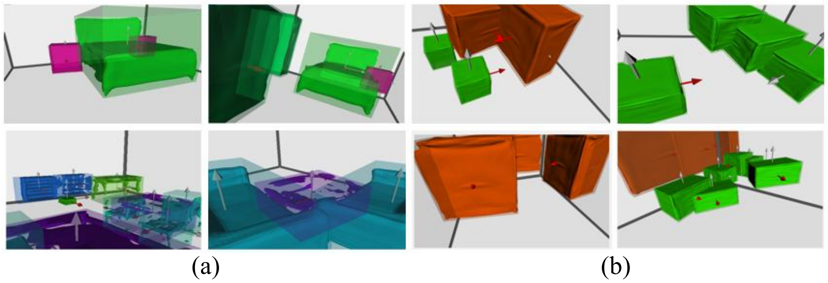


Fig. 4. Generation results of different datasets (a) SUNRGBD dataset; (b) UrbanScene3D dataset.

and layout boxes, and some objects floats in the air. Furthermore, some objects' orientation is different from the real one. 2) SGCN can solve some problems above. As is shown in Fig. 3(c), objects in the scene do not suspend in the air anymore and the position is more accurate. This indicates that after refinement by SGCN, the model can accurately predict the position and size of the object's 3D boundary box. 3) Joint fine-tuning can further improve the performance of the model. As is shown in Fig. 3(d), improper intersection between objects and between objects and layout boxes is reduced, and the orientation of objects is more accurate. 4) SGCN and joint fine-tuning can improve the transfer ability of the model. Comparing only the results on the UrbanScene3D dataset in Fig. 3, it can be found that the results are improved to same extent after adding SGCN and joint fine-tuning, but there are still many problems, which will be discussed in the following.

4.3 Deep Analysis of Model Transferability

As is shown in Fig. 4, we will show the generation effect of the model on the SUNRGBD and UrbanScene3D datasets, and the transfer ability of the model and the problems will be discussed.

Fig. 4(b) shows that the model can accurately predict the location of objects in UrbanScene3D. However, comparing with Fig. 4(a) in SUNRGBD, there are still many shortcomings. For example, the reconstruction quality of details, shapes and textures of objects is poor, and the deviation of object's angle still exists. In addition, the category of reconstructed objects is relatively simple. Two main reasons are as follows:

(1) Different shooting methods. Most of SUNRGBD are head-up shots, while UrbanScene3D are mostly aerial shots and the camera position is not fixed.

(2) Differences in object classes. SUNRGBD is meant for indoor scenes, and the objects are mostly furniture objects. However, UrbanScene3D is meant for outdoor scenes, and the objects are more complex and diverse.

4.4 Visualization and fusion of multi-view scenes

We input multi-view time series data from UrbanScene3D to the model, and the results include 3D boundary boxes of layouts and objects, triangular mesh and

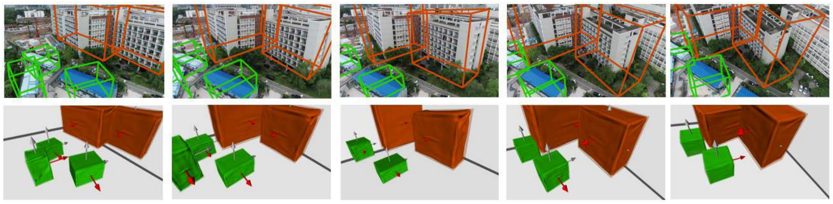
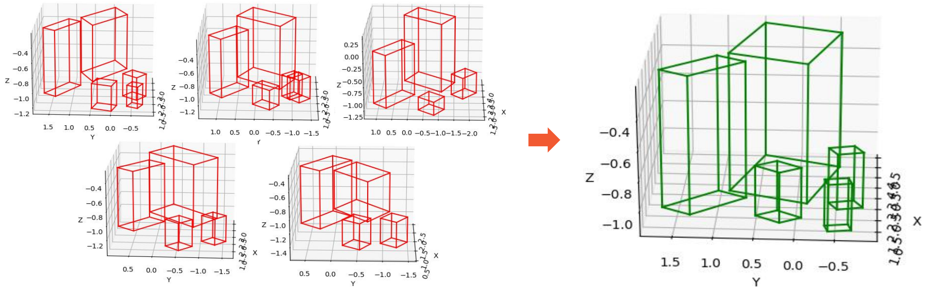


Fig. 5. 3D scene generation visualization results from multi-view time series data.



(a)

(b)

Fig. 6. Multi-view 3D scene fusion results. The left side is the 3D bounding box of the object generated by each frame of pictures before fusion, and the right side is the 3D bounding box of the fused object.

rotation matrix. Based on those results, the geometries are created and rendered. The visualization results of two groups are shown in Fig. 5. It can be seen that the model can reconstruct 3D scenes from each frame of time series data. In this way, each frame has corresponding reconstruction result.

In order to fuse the boundary boxes of objects from different perspectives, the method in Section 3.3 is used. The results before and after the fusion of two groups are shown in Fig. 6. Comparing before and after the fusion, it can be found that the fusion makes up for the location information loss by the same object due to different shooting angles. In conclusion, the model can effectively recover the overall 3D scene from multi-view time series data and realize the fusion of multi-view location information.

5 CONCLUSION

In this paper, we design a framework SGMT, which can can recover the overall 3D scene from multi-view video data. However, there are still some problems remain to be further discussed. Firstly, what the framework can reconstruct is very dependent on the original dataset. When it was transferred to other datasets, the prediction of object categories, shape details and object position orientation is not accurate enough. Secondly, the method of multi-view scene fusion does not involve the fusion of object shape and texture features. Therefore, the next research will continue to train and optimize the model, improve the method of multi-perspective scene fusion, pay more attention to the fusion of object shape and texture features, and restore 3D scenes more completely.

Acknowledgments

This work is supported in part by the Excellent Youth Scholars Program of Shandong Province (Grant no. 2022HWYQ-048) and the Oversea Innovation Team Project of the ”20 Regulations for New Universities” funding program of Jinan (Grant no. 2021GXRC073)

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds (2017)
2. Dasgupta, S., Fang, K., Chen, K., Savarese, S.: Delay: Robust spatial layout estimation for cluttered indoor scenes. In: Computer Vision Pattern Recognition (2016)
3. Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.: Local deep implicit functions for 3d shape (2019)
4. Gkioxari, G., Malik, J., Johnson, J.: Mesh r-cnn (2019)
5. Hirzer, M., Roth, P.M., Lepetit, V.: Smart hypothesis generation for efficient and robust room layout estimation (2019)
6. Hsiao, C.W., Sun, C., Sun, M., Chen, H.T.: Flat2layout: Flat representation for estimating layout of general room types (2019)
7. Huang, S.: Cooperative holistic scene understanding: Unifying 3d object, layout, and camera pose estimation (2018)
8. Huang, S., Qi, S., Zhu, Y., Xiao, Y., Xu, Y., Zhu, S.C.: Holistic 3d scene parsing and reconstruction from a single rgb image. In: European Conference on Computer Vision (2018)
9. Janoch, A., Karayev, S., Jia, Y., Barron, J.T., Darrell, T.: A category-level 3d object dataset: Putting the kinect to work. In: IEEE International Conference on Computer Vision Workshops (2013)
10. Kulkarni, N., Misra, I., Tulsiani, S., Gupta, A.: 3d-relnet: Joint object and relational network for 3d prediction. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV) (2019)
11. Lee, C.Y., Badrinarayanan, V., Malisiewicz, T., Rabinovich, A.: Roomnet: End-to-end room layout estimation. In: 2017 IEEE International Conference on Computer Vision (ICCV) (2017)
12. Lin, H.J., Huang, S.W., Lai, S.H., Chiang, C.K.: Indoor scene layout estimation from a single image. In: 2018 24th International Conference on Pattern Recognition (ICPR) (2018)
13. Liu, Y., Xue, F., Huang, H.: Urbanscene3d: A large scale urban scene dataset and simulator (2021)
14. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM SIGGRAPH Computer Graphics pp. 163–169 (1987)
15. Mallya, A., Lazebnik, S.: Learning informative edge maps for indoor scene layout prediction. In: 2015 IEEE International Conference on Computer Vision (ICCV) (2015)
16. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

17. Nie, Y., Han, X., Guo, S., Zheng, Y., Zhang, J.J.: Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
18. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis Machine Intelligence **39**(6), 1137–1149 (2017)
19. Ren, Y., Li, S., Chen, C., Kuo, C.C.J.: A coarse-to-fine indoor layout estimation (cfile) method. Springer, Cham (2016)
20. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: R.: Indoor segmentation and support inference from rgbd images. in: Eccv (2012)
21. Song, S., Lichtenberg, S.P., Xiao, J.: Sun rgb-d: A rgb-d scene understanding benchmark suite. In: IEEE Conference on Computer Vision Pattern Recognition. pp. 567–576 (2015)
22. Xiao, J., Owens, A.H., Torralba, A.: Sun3d: A database of big spaces reconstructed using sfm and object labels. In: 2013 IEEE International Conference on Computer Vision (ICCV) (2013)
23. Xiao, J., Wang, R., Chen, X.: Holistic pose graph: Modeling geometric structure among objects in a scene using graph inference for 3d object prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 12717–12726 (October 2021)
24. Zhang, C., Cui, Z., Zhang, Y., Zeng, B., Liu, S.: Holistic 3d scene understanding from a single image with implicit representation (2021)