End-to-End Learned Multi-View Stereo Reconstruction with Transformers

Dongyue Lu
Technical University of Munich
dongyue.lu@tum.de

Zhisheng Zheng
Technical University of Munich
finn.zheng@tum.de

Abstract

Multi-view stereopsis is a significant topic in computer vision. From an input monocular RGB video and camera parameters, we reconstruct the surface geometry directly by regressing a sparse TSDF volume. A Swin-Transformer backbone is used to extract the most informative features for downstream fusion tasks. Features of keyframes are back-projected to 3D space, constructing a local window for representing the local geometry. TSDF values are regressed using sparse convolution in a coarse-to-fine manner to ignore free space and increase computational efficiency. A transformer-based fusion module makes the local reconstruction to be globally consistent with the previously reconstructed windows. Our method is able to learn the local smoothness and global shape prior of 3D surfaces at interactive rates. Experiments on ScanNet show that the reconstruction quality and efficiency of our method are comparable to current state-of-the-art methods.

1. Introduction

Multi-view stereopsis (MVS) is a classic computer vision problem which aims to reconstruct a 3D scene given a set of images with corresponding camera parameters. This technology is one of the key components of autonomous driving, augmented reality and target motion analysis. Many pioneering works have been proposed in this field and one common method is to estimate the depth maps and fuse them into a Truncated Signed Distance Function (TSDF) volume from which the reconstructed mesh can be generated with Marching Cubes algorithm [1].

However, It is difficult for depth based method to consider the global priority even when many scenes overlap, resulting in noise artifacts and unsmooth local reconstruction. Recently, methods for directly predicting 3D geometries from images have made great progress. For instance, using a transformer network, TransformerFusion [2] fuses the observations into a volumetric feature grid representing the scene, which is later decoded to predict surface occupancy.

Inspired by these works and the great development of transformer application in computer vision, especially Swin-Transformer [3], we propose a novel framework integrated with transformer for real-time MVS reconstruction. Basically, features extracted by Swin-Transformer backbone are back projected to construct a feature volume as space representation. Sparse convolution and coarse to fine manner is used to refine features spatially. 3D reconstruction is preformed incrementally and a transformer fusion module is leveraged to guide the local reconstruction, resulting accurate reconstruction in real time.

2. Related Work

2.1. Depth Estimation

The boom in deep learning has brought great progress in learning-based depth estimation models. MVDepthNet [4] uses 2D CNN to process a cost volume constructed from multi-view image features and predicts depth in an encoder-decoder manner. DPSNet [5] leverages 3D CNN to process the concatenated features for the generation of cost volume. The estimated depth can be later used for RGB-D reconstruction. Unlike depth based method, SurfaceNet [6] back-projects a series of images to 3D space, directly uses 3D CNN to process them and obtains the 3D reconstruction results.

2.2. Transformer

Designed for sequence modeling, Transformer [7] has achieved tremendous success in NLP domain, which leads researcher to dig its potential on computer vision. Liu Z et.al [3] proposes a noval architecture Swin-Transformer, which can effectively learn the correlation between long-term features and can be applied as a general-purpose backbone for downstream tasks. This innovative architecture presents the huge potential of Swin-Transformer-based vision model.

3. Proposed Method

Given a set of N RGB images I_i of a scene with corresponding camera intrinsic parameters $\mathbf{K}_i$ and extrinsic poses $\mathbf{P}_i$, our goal is to reconstruct the scene geometry in real-time by predicting a global TSDF volume as the representation of this scene. Figure 1 shows the basic architecture of our proposed model.

3.1. View selection and window construction

Considering all frames as input to our model is very computationally challenging with large-scale scenes, so we need to sample all incoming frames, which is based on sufficient rotation R_{max} and translation T_{max} between the cameras of the two frames. Every 10 incoming keyframes I^{key} constitute a window. Within each window, we construct a cube containing the view frustum of all keyframes within a max depth range. In the early training, we will focus on the reconstruction of this local window.

3.2. Feature extraction and volume construction

We implement Swin-Transformer $\mathcal{S}$ as backbone for the multi-level feature extraction. The input image is first divided into small patches without overlapping, and then fed into the linear embedding layer for downsampling. The main Swin-Transformer block will implement the attention mechanism in the image for information exchange. In order to obtain features with different number of channels and size, we use N_l different Swin-Transformer blocks.

$$\mathcal{S} : I^{key} \mapsto (\Phi_w^1, ..., \Phi_w^{N_l})$$

Following other volumetric methods, after using image backbone to extract features, we construct the feature volume by back-projecting the features of keyframes along each ray into the feature volume using camera parameters $\mathbf{K}_i$ and $\mathbf{P}_i$. Features of different keyframes are aggregated by average operation.

$$\mathcal{B} : (\Phi_w^l, \mathbf{K}_i, \mathbf{P}_i) \mapsto F_w^l$$

3.3. Coarse to fine TSDF reconstruction

Given feature volume, we predict the TSDF and occupancy probability in each voxel in a coarse to fine manner. After combining features from last level, sparse 3D convolution $\mathcal{C}$ proposed in [8] is used to refine features spatially and also improve computation efficiency.

$$\mathcal{C} : (F_w^{l-1}, F_w^l) \mapsto \tilde{F}_w^l$$

At each level, the global feature volume F_g^l is built incrementally by a transformer module $\mathcal{T}$ that updates $\bar{F}_w^l$

$$\mathcal{T} : (F_g^l, \tilde{F}_w^l) \mapsto \bar{F}_w^l$$

The TSDF value and occupancy probability is predicted by two MLP $\mathcal{M}$ and features in occupied voxel will be upsampled and propagated to next level. With this coarse to fine manner, we can focus on the area near the surface and ignore the free space.

$$\mathcal{M} : \bar{F}_w^l \mapsto (TSDF \in [-1, 1], o \in [0, 1])$$

3.4. Local window fusion with attention-based feature propagation

The transformer module is used to fuse local state $\tilde{F}_w^l$ and hidden state F_g^l and avoid equal-weight treatment of each individual frame that may contain less useful information. Specifically, hidden state F_g^l is extracted in the same window volume as local state $\tilde{F}_w^l$ and then both of them are fed into the fusion module. The predicted features $\bar{F}_w^l$ are replaced in original position. Given that the computation of the commonly used scaled dot product attention in [7] of long sequences is very expensive, we design different modules and compare them in following experiments. First, we use a simplified attention mechanism in [9]. Basically, it shares the same idea as scaled dot product attention: compute the similarity of two inputs, use **softmax** to get attention and compute context vector and final update.

$$e = tanh(Linear[\tilde{F}_w^l, F_g^l])$$
$$atten = softmax(Linear(e))$$
$$\bar{F}_w^l = tanh(Linear[\tilde{F}_w^l, atten \odot F_g^l])$$

Second, we design a transformer block following [7], For simplicity, the transformer block consists of a multi head attention with 4 attention heads using above described attention and a feed forward layer. Both multi head attention and feed forward layer contain ReLU activation, residual connection and layer norm.

Third, in order to apply scaled dot product attention, we also design a UNet attention block. We apply two layers of Conv3d and ConvTranspose3d to voxels to achieve downsampling and upsampling respectively. At the bottom, we apply a scaled dot product attention.

$$\bar{F}_w^l = softmax(\frac{\tilde{F}_w^l {F_g^l}^T}{\sqrt{d_k}}) F_g^l$$

3.5. Loss function

There are two outputs in our network: occupancy probability and TSDF value in each voxel. For occupancy loss, we use Weighted Binary Cross Entropy Loss considering that most of the voxels in the space will be empty. For TSDF loss, we implement the logarithm mean absolute error following NeuralRecon [10].

3.6. Implemntation details

We perform the experiments on ScanNet(V2) [11] and implement our method in pytorch and torchsparse library.

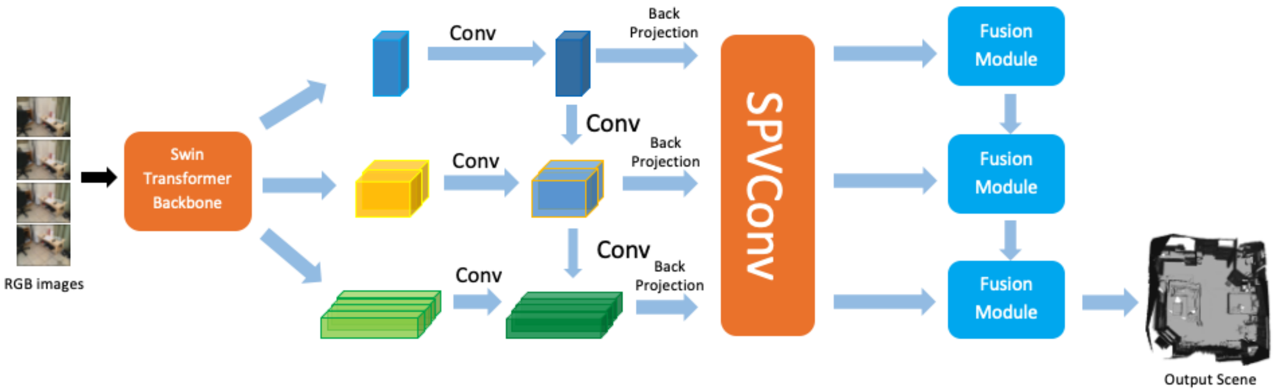

Figure 1. Proposed Architecture

For local window we set the voxel dimension $64 * 64 * 64$ the voxel size $0.04m$. R_{max} and T_{max} are set to be 0.1 and $0.15°$ respectively, so the entire training set of ScanNet is divided into 22650 local windows. Due to the limit of our hardware, we select 10000 windows for training, 2000 windows for validatoin and 2000 windows for testing. Training takes about one day using an Nvidia RTX 2080 GPU.

4. Experiments

4.1. Experiment Design

We evaluate the point-to-point errors between ground truth mesh and predicted mesh and compute precision and recall by calculating the ratio of point-to-point matches within small distances [2]. And we consider F-score as the most suitable metrics since both the accuracy and completeness of the reconstruction are considered.

We also compare our approach with state-of-the-art methods. Due to time constraints, we compare with our closest approach NeuralRecon and Atlas, as these methods build volumetric representation and predict 3D geometry directly. We retrain these two models on our dataset and use the same configurations. To compare efficiency, we also evaluate the time each method takes to reconstruct the scene, which is calculated as the time spent to reconstruct each scene divided by the number of frames in that scene.

4.2. Backbone Selection

To study the performance of Swin-Transformer as backbone, we first conduct a series of experiments under $1000/200$, $2000/500$, $4000/1000$ and $8000/1600$ training/validation windows for backbone selection. In order to eliminate the impact of other factors, we don't add fusion module to our network. Therefore, the results for comparison is between windows. We compare Swin-Transformer with pre-trained MnasNet [12]. We also conduct an ablation study, in which we back-project color directly.

Backbone	1000	2000	4000	8000
Non-Backbone	3.11	2.61	2.44	2.13
Pretrain-MnasNet	**1.97**	**1.92**	1.88	1.75
Swin-Transformer	2.11	1.98	**1.84**	**1.67**

Table 1. Backbone Selection: Results on different training windows. The value in this table represents the corresponding validation loss

Backbone	Precision	Recall	F-score
Non-Backbone	0.19	**0.65**	0.29
Pretraind-MnasNet	0.43	0.59	0.49
Swin-Transformer	**0.45**	0.60	**0.51**

Table 2. Backbone Selection: Final evaluation of different backbones

As shown in the Table 1, there is a huge gap between the validation loss of direct-projection and feature-projection, verifying the effectiveness of a feature extraction process. Comparing CNN backbone with Swin-Transformer, the generalization of Swin-Transformer based model has a steady, significant improvement, surpassing CNN backbone with increasing dataset size.

Finally, we evaluate the quality of generated mesh in the 8000 windows case. The result can be seen from Table 2. The inbalance of recall and precision in Non-Backbone case is probably because that the model just tries to over-complete reconstructions with noisy surface. This imbalance doesn't occur on the two backbone case, where the Swin-Transformer outperforms Pretrain-Mnasnet on all metrics.

4.3. Fusion Modules

We use three different attention mechanisms as fusion modules and compare their performance qualitatively and quantitatively. It can be seen from Table 3 and top row in Figure 2 that the performance of the most complex UNet at-

Method	Precision	Recall	F-score	Runtime(ms)
AttentionLayer	0.45	0.50	0.48	**40**
Transformer	**0.47**	**0.62**	**0.53**	47
UNet Attention	0.43	0.41	0.42	61

Table 3. Comparsion between different fusion modules

tention block is the worst. To analyze the reason, it's maybe because valuable information between two long sequences is lost after sampling, and complex model also increases the difficulty of training. The transformer uses multi head attention, and different attention heads focus on different aspects of the sequence, which achieves the best overall performance. In terms of runtime, single-layer attention and transformer are faster and can basically achieve real-time reconstruction, while the complex UNet attention block is the slowest. Qualitatively, transformer and attention linear achieve our expectation of combining the surrounding context, the reconstruction is smooth and consistent, while the UNet attention reconstruction is rough.

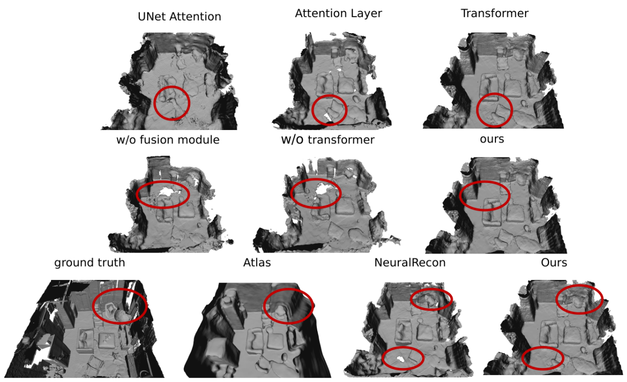

Figure 2. Comparison of different fusion methods

4.4. Overall Performance

From the quantitative and qualitative comparison results Table 4 and bottom row in Figure 2, our method has similar performance to NeuralRecon and even surpasses it in some scenes, but lower than atlas. As an offline method, Atlas provides very smooth reconstructions, and even oversmoothed geometries, resulting in inaccurate reconstruction, like the circled region in respective figure in Figure 2. Our method and NeuralRecon have generally good reconstruction quality with much higher speed compared to Atlas.

4.5. Ablation studies and failure cases

To verify the effectiveness of our transformer fusion module, we also do ablation experiments. In the first experiment, we completely remove the fusion module, that is, after reconstructing each local window, the incoming window is directly substituted in the corresponding location. In the second experiment, we use a simple linear layer for information exchange. From the middle row in Figure 2 and Table 5, we can see the performance of both are worse than our transformer module, especially that the direct substitute method results are rough, which reflects the ability of our transformer module to consider the surrounding conditions and context.

Method	Precision	Recall	F-score	Runtime(ms)
Atlas	0.49	**0.71**	**0.58**	280
NeuralRecon	**0.57**	0.52	0.54	**38**
Ours	0.47	0.62	0.53	47

Table 4. Comparsion with Atlas and NeuralRecon

Method	Precision	Recall	F-score
w/o fusion module	0.39	0.54	0.45
w/o transformer	0.45	0.51	0.47
Ours	**0.47**	**0.62**	**0.53**

Table 5. Ablation studies

As shown in Figure 3, in complex and occluded scenes, our reconstruction details are not enough, such as desks, seats, windows, which can be left for future exploration.

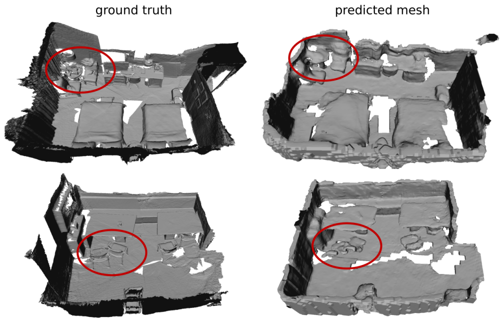

Figure 3. Failure cases

5. Conclusion

In this project, we present an end-to-end MVS method for real-time 3D reconstruction. We leverage the prevalent transformer on two parts of our model. The key idea is to jointly reconstruct and fuse sparse TSDF volumes for each window incrementally by 3D sparse convolutions and transformer fusion module. Experiments show the positive effectiveness of Swin-Transformer feature extraction module and Transformer fusion module. The overall performance of our model ties NeuralRecon. For further improvement for our model, using the whole ScanNet dataset, deeper Swin-Transformer architecture are both promising.

References

[1] William Lorensen and Harvey Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21:163–, 08 1987. 1

[2] Aljaz Bozic, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 3

[3] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 1

[4] Kaixuan Wang and Shaojie Shen. Mvdepthnet: Real-time multiview depth estimation neural network. In *2018 International conference on 3d vision (3DV)*, pages 248–257. IEEE, 2018. 1

[5] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. Dpsnet: End-to-end deep plane sweep stereo. *arXiv preprint arXiv:1905.00538*, 2019. 1

[6] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2307–2315, 2017. 1

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1, 2

[8] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *European conference on computer vision*, pages 685–702. Springer, 2020. 2

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 2

[10] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607, 2021. 2

[11] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 2

[12] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019. 3