Computers & Graphics (2023)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

AdaptMVSNet: Efficient Multi-View Stereo with Adaptive Convolution and Attention Fusion

Pengfei Jiang[a,b], Xiaoyan Yang[a], Yuanjie Chen[b], Wenjie Song[b], Yang Li[a,]

[a]*Department of Computer Science, East China Normal University, Shanghai, CN*
[b]*Nanhu Laboratory, Jiaxing, CN*

ARTICLE INFO

Article history:
Received July 13, 2023

Keywords: Multi-View Stereo, 3D Reconstruction, Computers Vision, Deep Learning

ABSTRACT

Multi-View Stereo (MVS) is a crucial technique for reconstructing the geometric structure of a scene, given the known camera parameters. Previous deep learning-based MVS methods have mainly focused on improving the reconstruction quality but overlooked the running efficiency during the actual algorithm deployment. For example, deformable convolutions have been introduced to improve the accuracy of the reconstruction results further, however, its inability for parallel optimization caused low inference speed. In this paper, we propose AdaptMVSNet which is device-friendly and reconstruction-efficient, while preserving the original results. To this end, adaptive convolution is introduced to significantly improve the efficiency in speed and metrics compared to current methods. In addition, an attention fusion module is proposed to blend features from adaptive convolution and the feature pyramid network. Our experiments demonstrate that our proposed approach achieves state-of-the-art performance and is almost 2× faster than the recent fastest MVS method. We will release our source code.

1. Introduction

Multi-view Stereo (MVS) is a fundamental problem in the computer vision field, which aims to reconstruct the 3D geometry of a scene from a set of images. Generating 3D models of the real world by using MVS approaches has a wide range of practical applications, including industrial production, consumer businesses, and art design. This makes MVS a highly regarded area of research from both academic and industrial perspectives.

Traditional MVS methods [1, 2, 3] depend on feature matching which is manually designed by incorporating expert knowledge and strong priors to construct the cost volume. However, these methods lack flexibility and may not be universally applicable or robust across diverse scenes. Data-driven approaches using convolutional neural networks (CNNs) exhibit significant potential in overcoming challenges faced by traditional methods. MVSNet [4] is a recent CNN-based approach that leverages deep networks for depth prediction. It uses a 3D CNN to process depth assumptions and optimize the cost volume with a re-projected loss. To improve model performance, subsequent works [5, 6, 7] utilize finer cascades with more iterations in 3D CNN. However, as the structure complexity increases, computation costs may increase, leading to a reduction in efficiency. To reduce memory overhead and speed up the inference time for depth estimation, recent 3D CNN-based works [5, 6, 8] adopt a two-stage mode to avoid redundant calculations for empty space. On the other hand, some approaches [9, 10, 11] aim to accelerate model inference by replacing heavy 3D CNNs with more lightweight architectures. EffiMVS [9], for example, uses a lightweight 3D CNN to generate the coarsest initial depth map that is crucial in initiating the GRU and ensures rapid convergence. MVS2D [10] removes the cost volume and introduces epipolar constraints to search for paired features, which reduces memory usage and slightly improves model inference speed.

Nevertheless, the aforementioned methods either lack a proper balance between efficiency and performance or have limited effectiveness in reducing memory usage and inference time. The essential property of 3D reconstruction makes it challenging to achieve high performance while using minimal resources.

Although these methods have improved the convolutional structure, they rely solely on traditional convolutional operations, which use filters of fixed sizes and have limited adaptability to objects of varying sizes and shapes in images. In contrast, deformable convolution [12] allows for flexible sampling of input features in MVS task, leading to better versatility in accurately capturing object boundaries and shapes. Following this insight, PatchMatchNet [13] uses slim 2D deformable convolutions for MVS. However, deformable convolution is not hardware-friendly as it requires non-parallelizable mesh-grid sampling for displaceable feature extraction.

In this paper, we present AdaptMVSNet, a novel lightweight 2D CNN-based MVS framework to address the hardware compatibility issue. The proposed framework is mainly composed of an adaptive convolution module and an attention fusion module. An adaptive convolution is introduced to enable the filter to perform a per-pixel diverse convolution using a set of learnable kernel weights, resulting in an equally flexible expression ability without altering the sampling interval. Furthermore, the adaptive convolution module integrates adaptive convolution in feature extraction, cost evaluation, and depth propagation. The estimated kernel weights serve not only as parameters for convolution but also as attention features for further use. The attention fusion module combines both the attention features and image features, resulting in an enriched representation and enhanced discriminability. Finally, the combined features are processed to generate a feature volume used for depth prediction. Our AdaptMVSNet reduces the inference time required for high-resolution MVS while boosting model performance. To the best of our knowledge, our method is the most efficient method compared to state-of-the-art algorithms. It is 3.3× faster than FastMVSNet [11], 9× faster than MVSNet [4], and almost 2× faster than the recent fastest method PatchMatchNet [13]. Extensive experiments demonstrate the outstanding reconstruction results of our method.

In summary, our contributions are as follows:

- AdaptMVSNet is introduced and achieves state-of-the-art performance in both reconstruction completeness and inference time through extensive experiments.
- A hardware-friendly adaptive convolution module is proposed for feature extraction, cost evaluation, and depth propagation in MVS, offering significant improvements in speed and metrics compared to current methods.
- A novel and efficient attention fusion module is introduced to blend features from adaptive convolution and the feature pyramid network.

2. Related work

Our proposed method is closely related to learning-based multi-view stereo methods. we provide a brief review of these works as follows.

2.1. Traditional MVS methods

Multi-view stereo methods can be categorized into four types, voxel-based [14, 15], surface evolution-based [16, 17], patch-based [18, 19], and depth map-based [20, 2, 21]. Among these methods, depth map-based techniques are considered the most practical due to their lightweight 2D representation. Galliani [20] presents Gipuma, a massively parallel multi-view Patchmatch extension. A propagation-like red-black scheme is proposed, which is well-suited for multi-GPU parallel computing. Schönberger [2] presents COLMAP, which combines the 3D reconstruction pipeline of SfM (Structure-from-Motion) and MVS. It estimates pixel-wise view selection, depth maps, and surface normal. PatchmatchSt [1] adopts the assumption of space surface continuity, binocular parallel relationships, and depth diffusion, which iteratively generates the final depth map. TACFT [22] used traditional filter to accelerated inference and improved accuracy. PHI-MVS [23] used a potential depth value hypothesis generating and selecting strategy using Markov Random Field inference is proposed, improving the reconstruction of textureless regions. Although traditional depth map-based methods can achieve impressive results, their performance is limited under challenging conditions due to the use of hand-crafted models and features.

2.2. Learning-based MVS methods

Recently, learning-based methods have dominated the multi-view stereo research field. Volumetric methods [24, 25] compute a cost volume from multiple images and then infer surface voxels after cost volume regularization with a 3D CNN. However, these methods are limited to smaller-scale reconstructions. More commonly, depth map-based methods [26, 27, 4] operate in a similar fashion. MVSNet [4] was the first to propose an end-to-end 3D cost pipeline, which mainly consists of four steps, image feature extraction by a 2D CNN, variance-based cost aggregation by homography warping, cost regularization through a 3D CNN, and depth regression. However, MVSNet can only handle low-resolution images due to memory restrictions and computational requirements. To generate high-resolution depth maps, some methods have been proposed. DeepStereoADVR [28], CascadeMVSnet [5], and CVPMVS [6] utilize recurrent networks or a coarse-to-fine strategy to replace 3D CNN regularization. These methods process the cost volumes along the depth dimension, reducing both memory consumption and running time. MVSformer [29] proposes a new feature extraction method and employs the ViT (Vision Transformer) [30] for enhanced reconstruction accuracy at the cost of additional computation and memory overhead. CBi-net [31] utilizes a binary tree search method for estimating depth maps during multiple cycles, whereas most methods use classification or regression for the same purpose. SP-MVS [32] introduces a geometry-aware regularization module to enhance the representative power of cost volume regularization. Geo-ConfNet [33] predicts the correctness of a depth hypothesis via a deep neural network that explores both spatial coherence

and cross-view consistency. Some researchers [34] adapt reinforcement learning to minimize a photometric loss, thereby overcoming the optimization challenges brought by the iterative process of Patchmatch. The above methods better overcome the limitations of traditional methods, but they require massive computation.

2.3. Learning-based efficient MVS methods

Recently, the efficiency of multi-view stereo (MVS) has garnered increasing attention. Fast-MVSNet [11] proposes a novel sparse-to-dense coarse-to-fine framework for fast and accurate depth estimation in MVS. PatchmatchNet [13] employs the traditional patchmatch algorithm and pixel-wise convolution to reduce inference time and memory consumption. EfficientMVS [9] utilizes a lightweight 3D CNN to estimate a coarse depth map as the initialization of a GRU. Additionally, a dynamic lightweight cost volume is proposed, which can be processed by 2D convolution-based GRU iterations to avoid the memory and time-consuming problem of a large-sized static cost volume. ADEP-MVSNet [35] proposes the lightweight pixelwise depth estimation network, which can estimate depth value for each selected location independently. MVS2D [10] integrates multi-view constraints into the single-view network via an attention mechanism, which only utilizes 2D convolutions, further boosting performance and efficiency simultaneously.

The above-mentioned methods primarily focus on model accuracy and overlook the challenges that models may face in real-world applications. During actual deployment, computation optimization must be performed first, during which loop operations that don't depend on front and back data should be parallelized, making it an efficient design method. Currently, the most efficient method [13] uses deformable convolution, which is not hardware-friendly. To address this problem, we propose a more hardware-friendly approach, which adopts attention-based adaptive convolution.

3. Methods

In this section, we introduce the structure of AdaptMVSNet, a hardware-friendly and efficient multi-view stereo method, as illustrated in Figure 1. Our goal is to design a method that is effective and efficient for depth map estimation. To this end, we propose an adaptive convolution-based coarse-to-fine framework for depth map estimation.

Firstly, we provide a brief introduction to the baseline PatchmatchNet [13] framework. AdaptMVSNet adopts a multi-scale depth estimation strategy. First, the model uses a feature pyramid network (FPN)[36] to extract multi-scale features. Then, using the Adaptive Convolution Module within the purple bounding box to extract adaptive convolution to process the feature map and depth map, as depicted in detail in Figure 2. Attention features are blended into adjacent frames by projecting attention feature using the Attention Fusion Module within the red bounding box, as shown in Figure 3. Finally, the depth refinement residual module optimizes the predicted depth estimates.

To optimize operations for parallelization, the deformable convolution is replaced and a device-friendly **Adaptive Convolution Module (ACM)** is designed. Additionally, a novel and innovative **Attention Fusion Module (AFM)** is introduced to integrate attention features learned within the Adaptive Convolution Module with multiscale features. The residual estimation approach and loss functions for refining the depth map are described at the end.

3.1. Preliminary

We refer to the framework of PatchmatchNet [13]. The Feature Pyramid Network (FPN) [36] extracts multi-scale feature maps. The depth hypotheses are uniformly initialized between the maximum and minimum depths, then random perturbation is added to the initialized depth space. The matching cost is computed using deformable convolution and depth updates at each stage also use deformable convolution in PatchmatchNet. To gather K_p depth hypotheses for pixel p in the reference image, PatchmatchNet learns additional 2D offsets $\{\Delta o_i(p)\}_{i=1}^{K_p}$ that are applied on the top of fixed 2D offsets $\{o_i(p)\}_{i=1}^{K_p}$, organized as a grid. It applies a 2D CNN on the reference feature map F_0 to learn additional 2D offsets for each pixel p and get the depth hypotheses $D_p(p)$ via bilinear interpolation as follows

$$D_p(p) = \{D(p + o_i + \Delta o_i(p))\}_{i=1}^{K_p} \tag{1}$$

The local hypotheses are then propagated to the following layers, and finally, a refinement module optimizes the depth estimation using RGB images. In our method, the depth offset is regressed in the form of residuals.

3.2. Adaptive Convolution Module

To flexibly select features for different pixels, we design a convolutional operation to estimate per-pixel adaptive convolutional weights, which are shared across adaptive convolutions on the feature maps, depth hypotheses, and depth maps at the same layer. We refer to this module as the Adaptive Convolution Module, which takes the multiscale features and feature volume as inputs and outputs the depth map and attention features used in the subsequent layer.

3.2.1. Attention Feature Estimation

The Adaptive Convolution Module has the advantage of allowing for flexible and adaptive feature selection on a per-pixel basis, resulting in more precise and accurate depth estimations. Inspired by this idea, we propose a hardware-friendly method for estimating adaptive convolution weights. As illustrated in the upper part of Figure 2, we first extract per-pixel weights, treated as attention features $\check{W}_d(u, v)$, from the reference feature map F^r using a convolution with kernel size $k \times k$. The formula for estimating the attention feature $\check{W}_d(u, v)$ for pixel (u, v) is described below

$$\begin{aligned} \check{W}_d(u, v) &= \sigma(Conv(F^r(u, v), W_d, D_d)) \\ \check{W}_s(u, v) &= \sigma(Conv(F^r(u, v), W_s, D_s)). \end{aligned} \tag{2}$$

Here, σ denotes the sigmoid activation function. $Conv$ refers to the 2D convolution module, and $F^r(u, v)$ represents the feature on pixel (u, v) of the reference feature map F^r. D_d and

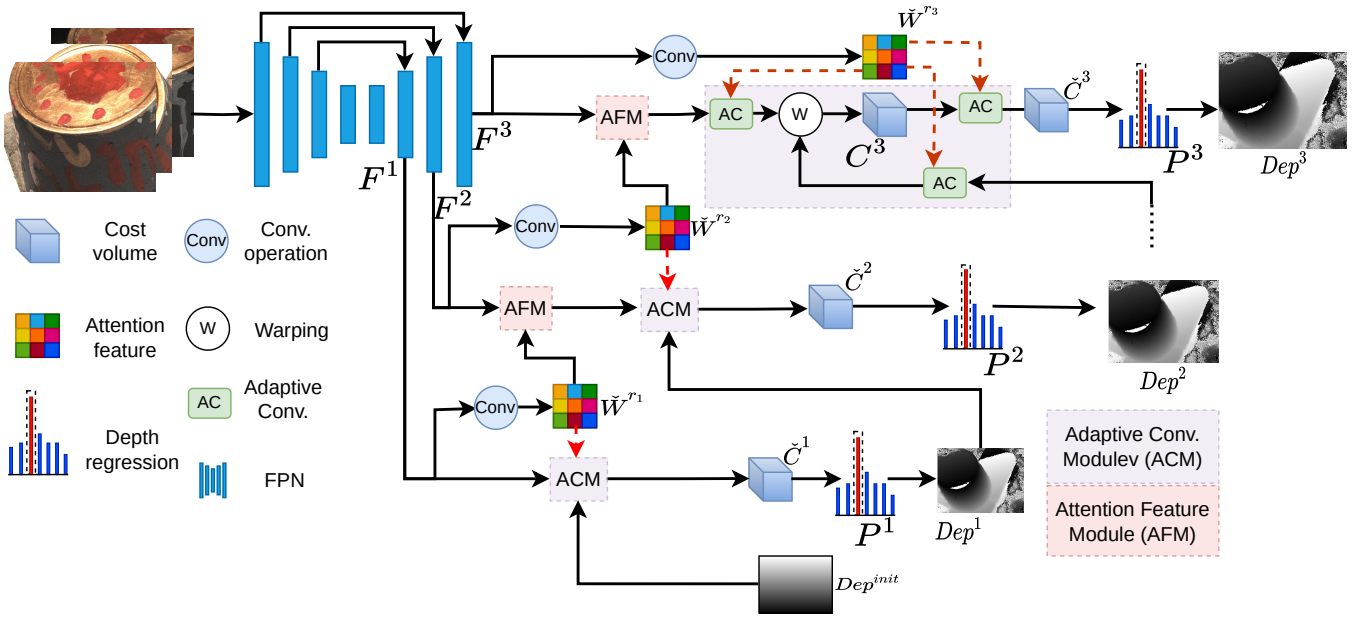


Fig. 1. Pipeline of AdaptMVSNet. AdaptMVSNet adopts a multi-scale depth estimation strategy. First, the model uses a feature pyramid network (FPN) to extract multi-scale features. Then, it uses the Adaptive Convolution Module within the purple bounding box to extract adaptive convolution to process the feature map, depth map and cost volume, as depicted in detail in Figure 2. Attention features are blended into adjacent frames by projecting attention feature using the Attention Fusion Module within the red bounding box, as shown in Figure 3. Finally, the depth refinement residual module optimizes the predicted depth estimates.

D_s represent different kernel dilations for dense and sparse convolutions, respectively, while W_d and W_s represent the corresponding kernel weights for the convolution networks.

Later, as shown in the lower part of Figure 2, we assign these attention features $\check{W}_d(u,v)$ to the kernel of size $k \times k$ in the adaptive convolution operation.

3.2.2. *Adaptive Convolution*

Sharing attention features across layers enables faster and more efficient computation. Different from deformable convolution as Eq.(1). Inspired by FewShifts [37], we designed the adaptive convolution, which has a similar simple structure and high efficiency at the same time. This shift pattern is illustrated in Figure 2. The main difference between our adaptive convolution and FewShifts [37] is that the former learns the weight of shift, while the latter learns the offset of shift. Adaptive convolution adopts a fixed offset method, making it faster on GPU devices. The same attention features support for adaptive convolution operations on different types of inputs within the same layer.

Since the adaptive convolutional weights are different for each pixel, traditional convolutional methods cannot be used for parallel processing. Instead, we implement a *expand* approach for parallel computing. We expand each pixel in the feature map to the same size as the convolution kernel and then perform element-wise multiplication with the per-pixel adaptive convolutional weights. Finally, we sum the nine values to obtain the output. In theory, this operation is consistent with shifting the kernel weights in eight directions and then doing each pixel's independent convolution operation. Thus, for an input value $X(u,v)$ on pixel (u,v) that needs to be processed, with the adaptive convolutional weights $\check{W}(u,v)$ and dilation D, the adaptive convolution operation is shown in the following equation

$$\begin{aligned} &\mathrm{AdapConv}(X(u,v), \check{W}(u,v), D) \\ &= \sum_s \sum_t X(u + s * D, v + t * D) \cdot W(u,v)[s,t], \\ &\mathrm{s\ and\ t} \in \{-1, 0, 1\} \end{aligned} \tag{3}$$

where s and t represent the offset units in the u and v directions, respectively, while D represents the dilation of the convolution. The offset is the product of the offset unit and dilation factor, resulting in the offset coordinate of $(u + sD, v + tD)$. For an adaptive convolutional weight W with nine values, the corresponding weight value $W(u,v)[s,t]$ is selected based on the offset unit.

Adaptive Convolution on Features Maps

To focus more on locally similar features, we utilize Adaptive Convolution when processing the feature map as follows

$$\begin{aligned} \check{F}^r_d(u,v) &= \mathrm{AdapConv}(F^r(u,v), \check{W}_d(u,v), D_d) \\ \check{F}^r_s(u,v) &= \mathrm{AdapConv}(F^r(u,v), \check{W}_s(u,v), D_s) \\ \check{F}^r(u,v) &= (\check{F}^r_d(u,v) + \check{F}^r_s(u,v))/2 \end{aligned} \tag{4}$$

The outputted adaptive feature maps $\check{F}^r_s(u,v)$ and 4D feature volumes are input to a group-wise point multiplication [26] along with dimension reduction by element-wise 3D CNN to obtain 3D feature volumes $C(u,v)$. Specific details and explanations will be provided in the next paragraph.

Adaptive Convolution on Feature volumes

Traditional MVS matching algorithms aggregate features by 3D convolution on the feature volume. Nonetheless, 3D convo-

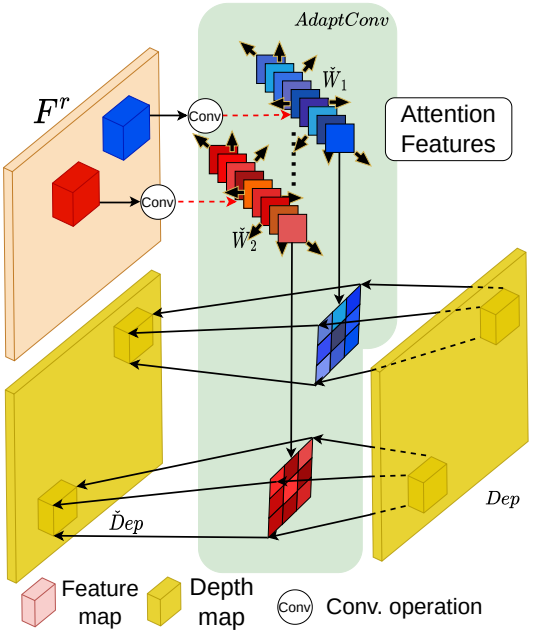


Fig. 2. Adaptive Convolution (AC). The input of the AC module can be a feature map, a 3D feature volume or a depth map. This figure only uses the depth map as an example. The convolution weight of each pixel is independently learned from the feature map of the corresponding pixel, which is called attention feature. The attention feature is shifted in all directions to form a different adaptive convolution operation for each pixel. In implementation, because the convolution kernel is different for different convolutions, the existing convolution operation cannot be used directly.

We expand each pixel in the feature map to the same size as the convolution kernel, and adopt The method of pixel point integral and summation realizes adaptive convolution operation.

lution on a 4D feature volume is computationally expensive and time-consuming. Inspired by SimpleRecon [38], we first use a 3D convolution operation that is equivalent to an MLP to reduce the 4D feature volume to a 3D feature volume. We then use the Adaptive Convolution Module that we propose to aggregate the 3D cost volume. While reducing the convolution kernel size to $1 \times 1 \times 1$ may impair the model's local perception ability, the Adaptive Convolution Module from the feature map has a local attention mechanism that complements this deficiency. The formula for the adaptive convolution on the 3D feature volume $C(u, v)$ using the cost volume $\check{C}_d(u, v)$ is as follows

$$\begin{aligned} \check{C}_d(u,v) &= \text{AdapConv}(C(u,v), \check{W}_d(u,v), D_d) \\ \check{C}_s(u,v) &= \text{AdapConv}(C(u,v), \check{W}_s(u,v), D_s) \\ \check{C}(u,v) &= (\check{C}_d(u,v) + \check{C}_s(u,v))/2 \end{aligned} \tag{5}$$

The depth probability $P^r(u, v)$ is obtained by applying a $softmax$ operation to the cost volume $\check{C}_d(u, v)$. The optimal depth map $Dep^r(u, v)$ is the expectation of the depth probability and depth hypotheses.

Adaptive Convolution on Depth Maps

The depth map serves as input to the adaptive convolution and outputs a new depth map, as shown in Figure 2. Finally, the outputs of the two adaptive convolutions are merged to obtain the final depth estimation

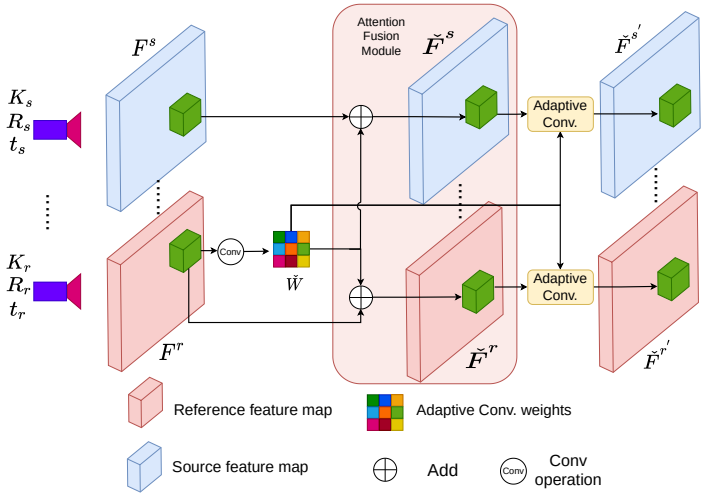


Fig. 3. Attention Fusion Module. As figure illustration, using traditional convolution to get respective attention feature of each pixel of reference feature map, then projecting the attention features of reference frames to all feature maps of source frames according to the known internal and external parameters of the camera.

$$\begin{aligned} \check{Dep}^r_d(u,v) &= \text{AdapConv}(Dep^r(u,v), \check{W}_d(u,v), D_d) \\ \check{Dep}^r_s(u,v) &= \text{AdapConv}(Dep^r(u,v), \check{W}_s(u,v), D_s) \\ \check{Dep}^r(u,v) &= (\check{Dep}^r_d(u,v) + \check{Dep}^r_s(u,v))/2 \end{aligned} \tag{6}$$

3.3. Attention Fusion Module

The learnable adaptive convolutional weights can be reused as attention features in volumetric settings, providing a richer and more discriminative representation. This is because the adaptive convolutional weights can capture the complex spatial relationships between features, allowing for more precise and accurate feature selection on a per-pixel basis. Moreover, it enables the model to focus on the most relevant and informative features, enhancing the overall algorithm performance. By utilizing these attention features, the model can better distinguish between objects in cluttered scenes and extract more detailed information, making it an effective tool for depth estimation and 3D reconstruction tasks in complex environments.

In this section, we first introduce the depth hypothesis prediction based on the feature map. We then provide a detailed description of the attention feature projection method, which projects the attention features learned from the reference frame onto the source frame at the pixel level.

3.3.1. Depth Hypotheses

Because the true depth values may lie in the vicinity of the predicted values, the loss volume constructed by projecting the features based on the depth values may be too sparse. Therefore, we construct K depth hypothesis candidates for each predicted depth value on each pixel.

Compared to directly predicting the depth candidate values in the depth map, predicting them in the feature space can improve the accuracy and robustness of depth estimation. In depth

estimation tasks, pixel values in the depth map are often noisy and have a wide range of distributions, making it challenging to predict the depth candidate values directly in the depth space. However, in the feature space, there is a deeper mapping relationship with the distribution of depth values. Pixels with similar features are likely to have similar depth values, and the estimated depth from the network should also have a similar uncertainty. Predicting depth hypothesis candidates in the feature space can reduce the uncertainty faced when estimating depth, as the different depth hypothesis values at different positions in the feature space often correspond to similar features, which are easier for the depth estimation model to learn. Moreover, features in the feature space can be shared across multiple depth values, further enhancing the model's robustness and generalization ability, thereby improving the accuracy of depth estimation.

For the depth map $\check{Dep}^r$ propagated by the adaptive convolution, we add perturbation from the feature map F^r to generate K depth hypothesis planes $\check{Dep}^{r,K}$, as shown in the following formula

$$\check{Dep}^{r,K} = \mathrm{Conv}(F^r) + \check{Dep}^r. \tag{7}$$

Here, K represents the number of perturbations. $Conv$ refers to a 2D convolution block.

3.3.2. Attention Feature Projection

For depth estimation tasks, the depth values corresponding to each position in the input image are synthesized from many local features. The use of attention mechanisms can help the model better utilize these local features and combine them, thereby improving the accuracy of depth estimation. One potential explanation is that attention convolution places more emphasis on more distinctive features, such as object edges, and in subsequent cost volume construction, these more distinctive features will provide more accurate feature matching. Therefore, sharing the attention features of the reference feature map among all views can enhance the features and compensate for the invalid matching caused by viewpoint changes.

As shown in Figure 3, attention features extracted from the reference feature map and all other current-layer multi-scale feature maps are fused. Specifically, an adaptive convolution operation is applied to the reference feature map to obtain an output that serves as the associated attention feature. Given the camera intrinsic K_r and extrinsic R_r, t_r of the reference view r, and the corresponding camera parameters K_s, R_s, t_s of the source view s, we can calculate the projected pixel position in the source image from the reference image as

$$\begin{bmatrix} u_s \\ v_s \\ d_s \\ 1 \end{bmatrix} = K_s * (R_s * (R_r^T K_r^{-1} \begin{bmatrix} u_r \\ v_r \\ d_r \\ 1 \end{bmatrix} - R_r^T t_r)) + t_s). \tag{8}$$

The different attention features from each pixel are projected onto the feature maps of other frames using sampling, following the above equation. The process of sampling features from frame r to frame s is represented by $P^{s,r}$.

3.3.3. Attention Feature Fusion

The benefit of fusing the features extracted from the pyramid and the attention features is that it can better utilize the information and features at different levels, thereby improving the accuracy and robustness of depth estimation. Pyramid extraction can capture information at different scales and granularities, and attention mechanisms can highlight key regions and features. Combining them can obtain a richer and more comprehensive feature representation, thereby improving the accuracy and robustness of depth estimation.

The attention feature is fused from two aspects of hybrid feature and adaptive convolution, as shown in Figure 3. In the first part, attention features $\check{W}d(u,v)$ and $\check{W}s(u,v)$ are estimated by a $k\times k$ operation, as mentioned in Sec. 3.2.1. Then, through a 1×1 convolution, the dimension of attention features is increased to match the dimension of the feature map. Dimension-increased attention features $\hat{W}d(u,v)$ and $\hat{W}s(u,v)$ are then projected onto all frames and added to their respective feature maps. The specific formula is as follows

$$\begin{aligned} \check{W}_d{}^s(u,v) &= P^{s,r}\, Conv(\check{W}_d(u,v)) \\ \check{W}_s{}^s(u,v) &= P^{s,r}\, Conv(\check{W}_s(u,v)) \\ \check{F}_s{}^s(u,v) &= (\check{W}_d{}^s(u,v) + \check{W}_2{}^s(u,v))/2 + F^s(u,v). \end{aligned} \tag{9}$$

Here, $P^{s,r}$ is the process of sampling features from frame r to frame s. $F^s(u,v)$ and $\check{F}^s(u,v)$ respectively represent the original source feature map and fused source feature map for pixel (u,v).

In the second part, the weights of each adaptive feature $\check{W}d(u,v)$ and $\check{W}s(u,v)$ are assigned to the corresponding adaptive convolutional kernel, as described in Sec. 3.2.1. These adaptive convolutions process the fused features $\check{F}^s(u,v)$ from the first part to generate the 4D feature volume for the next layer.

3.4. Loss Function

Using a $softmax$ operation to map all depth hypotheses $\check{C}(u,v)$ to a depth distribution $P(u,v)$, which is used for depth regression [13]. The depth prediction $\hat{Dep}(p)$ at pixel (u,v) during training is the expectation as

$$\hat{Dep}_{train}(u,v) = \sum_{i=0}^{K} (P_i(u,v) \cdot \check{Dep}_i^K(u,v)). \tag{10}$$

Smooth L1 loss L_d is used as the cost for depth estimation.

However, the depth loss only constrains the output geometry in the depth dimension, while smooth surfaces such as planes are common in scenes and objects. For such surfaces, the values in the depth space vary uniformly, while the values in the normal space remain constant. Low-frequency and simple information is easier for the network to learn. Therefore, we apply the normal loss function from SimpleRecon [38] to constrain the depth gradient. The ground-truth normal map $\bar{N}$ and estimated normal map N are calculated from the depth map.

The final loss function incorporates the results obtained from all the L-stages and the refined module as

$$L = L_d(Dep_{refine}, D\bar{e}p) + \sum_{l=1}^{L}\sum_{i=1}^{I_l}(L_n(N_l, \bar{N}) + L_d(Dep_{l,i}, D\bar{e}p)). \tag{11}$$

The true depth values $D\bar{e}p$ at each layer are resized to match the size of the predicted depth map. Each level in L estimates I_l depth maps.

During the testing phase, the depth value with the highest probability for each pixel is directly selected as the final depth value

$$D\hat{e}p_{test}(u,v) = D\check{e}p_i^K(u,v)[argmax(P(u,v))] \tag{12}$$

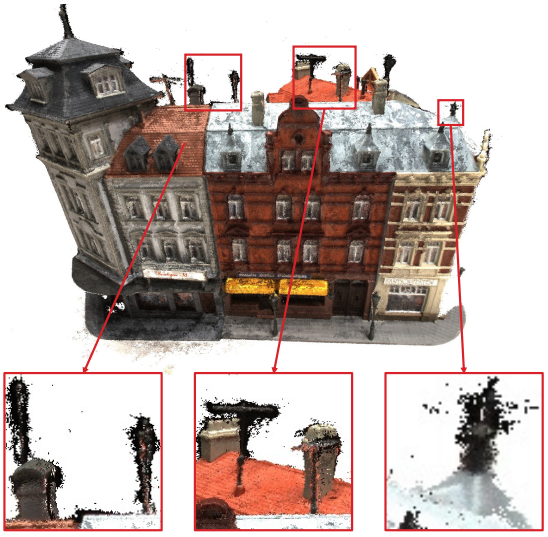
4. Experiments

To evaluate the effectiveness of our proposed method, we conducted extensive experiments on the depth estimation task on the DTU [40] and Tanks and Temples [41] datasets. In this section, we first introduce the experimental setup, including the implementation details and running settings. Then, we present the experimental results, comparing our method with several state-of-the-art methods and conducting ablation studies to analyze the contribution of each proposed module to the performance improvements. Finally, we provide qualitative visualizations to demonstrate the effectiveness of our method.

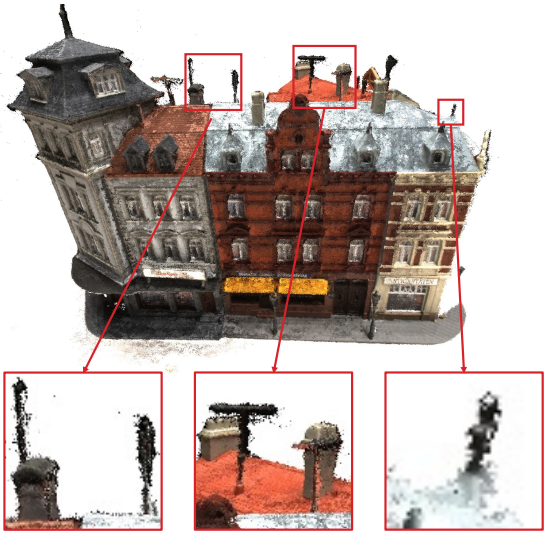
4.1. Experimental Setup

Implementation Details In the initial stage, we uniformly sample $K_0 = 48$ depth hypothesis planes in the normalized inverse depth for each pixel. For the following l layers, in addition to the K depth hypothesis planes estimated by the feature map, we also add K^l_{norm} depth hypothesis planes uniformly sampled in the normalized inverse depth plane for each pixel. The sampling interval is $r^l_{intr} * (Dep_{max} - Dep_{min})$. These two sets of sampled candidate depth hypotheses together form the final depth hypotheses. Unless otherwise specified, the values of K, K_{norm}, and r^l_{intr} for different layers l are $K = \{0, 9, 9\}$, $K_{norm} = \{8, 8, 16\}$, and $r^l_{intr} = \{0.005, 0.0125, 0.025\}$, respectively. The values of Dep_{max} and Dep_{min} are provided by the dataset and have different values for different scenes. To balance efficiency and accuracy, the size of the convolution kernel k used to extract attention features is set to 5.

Training setting Same as the training data generation method used in MVSNet [4], the point clouds provided by the dataset are used to reconstruct the mesh surfaces, which are then used to render the depth map for training. Our model is implemented using PyTorch [42]. The resolution of the input images for training is set to 640×512, and the number of views is set to 5. The selection strategy is the same as PVS-Net [43]. The Adam optimizer [44] is used with parameters of ($\beta_1 = 0.9, \beta_2 = 0.999$) and an initial learning rate of 0.001. The training is conducted on an NVIDIA A10 GPU device, with a batch size of 4.



(a) PatchmatchNet



(b) Ours

Fig. 4. Qualitative comparison of scan 9 in DTU. Compared with PatchmatchNet, our method predicts more accurate results of the small structures on the roof, especially the rod-shaped objects with clear outlines and less noise.

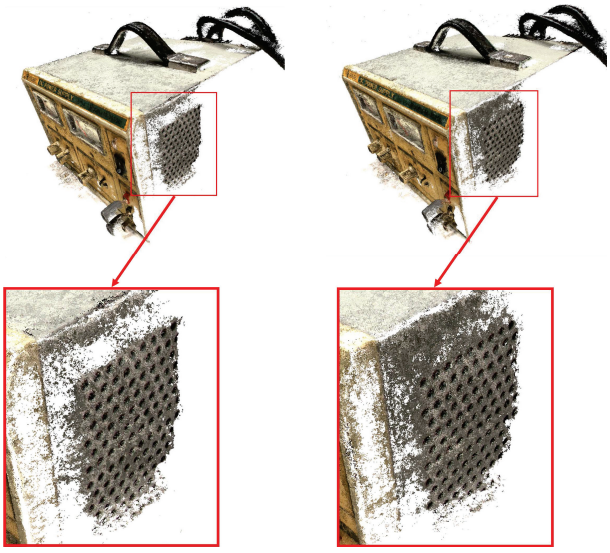
Testing setting During model testing, we deployed the model on an NVIDIA 2080ti GPU device and compiled it using TensorRT[1]

to optimize the model. To ensure a fair comparison, we use images with a resolution of 1600×1200 as input. Unlike most methods [13, 5, 6] that estimate depth maps using regression or classification during inference, AdapMVSNet compresses the forward runs of depth during testing. To balance training efficiency and robustness, the initial number of depth planes during testing remains at $D = 48$, but the same 9 planes are used in each iteration, and the estimated depth with the highest proba-

[1]TensorRt is designed to work in a complementary fashion with training frameworks such as TensorFlow, PyTorch, and MXNet.
"https://github.com/NVIDIA/TensorRT"

Table 1. Comparison results measured by the runtime, GPU memory requirements and reconstruction quality on the DTU evaluation (lower is better). Red stands for first place, green stands for second place, blue stands for third place.

Algorithm used	Inference time(s)	Memory(GB)	Overall(mm)	Acc(mm)	Comp(mm)
PatchmatchNet [13]	0.29	3.6	0.352	0.427	0.277
MVSNet [4]	1.05	10.8	0.551	0.456	0.646
Fast-MVSNet [11]	0.52	7.0	0.370	0.336	0.403
CVP-MVSNet [6]	1.51	8.8	0.351	**0.296**	0.406
UCS-Net [39]	0.54	6.6	**0.344**	0.338	0.349
CasMVSNet [5]	0.55	9.1	0.348	0.346	0.351
AdaptMVSNet (Ours)	**0.17**	**3.6**	0.351	0.445	**0.257**



(a) PatchmatchNet (b) Ours

Fig. 5. Qualitative comparison of scan 11 in DTU. Compared with our baseline, the proposed method recovers a more complete plane near the edge of the corner.

bility is selected as the output.

4.2. Comparison with SOTA

4.2.1. Results on the DTU dataset

The DTU dataset is a large-scale MVS dataset that contains 80 scenes with high diversity. Each scene was captured at 49 or 64 precise camera positions with 7 different lighting conditions. The dataset provides reference models acquired by a high precision structured light scanner and high-resolution RGB images. We used the same training, validation, and evaluation sets as other learning-based methods [13, 45, 46].

Table 1 shows a quantitative comparison of our reconstruction results with recently published learning-based MVS methods, such as PatchmatchNet, DynamicCVMVS, and FastMVS. Our reconstruction results are competitive, which verifies the effectiveness of our method. To evaluate the memory and runtime efficiency of different MVS methods, we also compared the performance of several MVS methods that focus on improving efficiency in terms of time and memory consumption. For each method, we recorded the memory usage and inference time for each scene in the DTU dataset. To guarantee an objective comparison, we compiled and deployed the models using

Table 2. Ablation Study of without Proposed Module on DTU (mm)

Method	Overall	Acc	Comp
Ours	0.351	0.445	0.257
w/o Adaptive Convolution Module	0.373	0.442	0.303
w/o Attention Fusion Module	0.383	0.469	0.296
w/o Normal Loss	0.360	0.469	0.252

Table 3. Ablation Study of Different Convolution on DTU (mm)

Method	adaptive propagation	adaptive evaluation	Overall	Acc	Comp
Ours	✓	✓	0.351	0.445	0.257
Ours-V1	×	✓	0.390	0.479	0.301
Ours-V2	✓	×	0.401	0.482	0.320

TensorRT on an NVIDIA GTX 2080Ti GPU device to compute the runtime. The results demonstrate that our method has the fastest inference time among all methods, reducing the time by 0.17 seconds and 0.12 seconds compared to PatchmatchNet, and has relatively lower memory utilization, approximately 3.6 GB. The completeness reaches 0.257mm, which approaches the state-of-the-art level. AdapMVSNet has lower memory utilization and the lowest inference time.

Figure 4 shows the qualitative results of our method and PatchmatchNet [13] at scan 9. AdapMVSNet can better capture the fine structures on the roof and provide more accurate boundaries than PatchmatchNet. The results for scan 11 are presented in Figure 5. Our method reconstructs a more complete point cloud, particularly at the locations of edge discontinuities. Figure 7 presents more visualization results on the DTU dataset. It can be observed that our method can recover the geometry and appearance of scenes well, even with various object structures and materials.

4.2.2. Results on Tanks & Temples

The DTU dataset is designed for single-object reconstruction. To demonstrate the effectiveness and robustness of our approach in complex and large-scale scenes with real-world data, we conducted experiments on the Tanks&Temples. We use the model trained on DTU without any fine-tuning. For evaluation, we set the input image size to 1920×1056 and the number of views N to 7. Qualitative results on intermediate dataset is shown on the Figure 6. The visualized point cloud from experiments can illustrate our method has considerable robustness.

Fig. 6. Generalization Prediction Results on the Untrained Tanks&Temples Dataset. AdapMVSNet predicts a relatively complete object structure on the real outdoor scene dataset.

Table 4. Ablation Study of Different Number of Views on DTU (mm)

Number	Inference time(s)	Memory (GB)	Overall	Acc	Comp
5	0.15	3.5	0.351	0.429	0.273
6	0.17	3.6	0.351	0.445	0.257
7	0.20	3.7	0.351	0.434	0.268

4.3. Ablation study

In this section, extensive ablation experiments to analyze the effects of the Adaptive Convolutional Module (ACM), Attention Fusion Module (AFM), the Number of views, and Normal Loss are provided. Then, combinations of deformable convolution and adaptive convolution at different stages are evaluated.

Adaptive Convolutional Module By comparing the first and second data in **Table 2** under different operations of adaptive convolutions and traditional convolutions, we found that our proposed ACM greatly influences the integrity and accuracy of the reconstruction results, reducing the integrity error by 15.2 percent. The improvement is attributed to the flexibility and learning ability of the ACM. The adaptive convolution is more flexible because the ACM can adaptively select the shape and size of each convolution kernel, enabling the model to better adapt to different features and structures in the data. This adaptivity enables the model to utilize more feature information, thereby improving the performance of depth estimation. In addition, the variable convolution parameters of the ACM have strong learning ability. The parameters of the ACM are learnable, which means that they can be optimized through back-propagation to improve the model's output results. This allows the ACM to better adapt to different data and tasks, thereby improving the performance of depth estimation.

Attention Fusion Module The AFM can greatly improve the integrity and accuracy of the reconstruction results, reducing the integrity error by 13.1 percent and the accuracy by 5.1 percent. The module can enable the model to focus more on important detailed features, optimize the importance of attention features in the depth estimation process, and to some extent alleviate the influence of local feature bias and noise in depth estimation, thereby improving the modeling ability and accuracy of complex scenes. In addition, the AFM can selectively fuse information across different viewpoints, enabling the model to capture richer detailed information from different viewpoints and improve the accuracy and robustness of depth estimation results.

Number of Views To figure out the effect of the number of views on reconstruction quality and storage, we compare the impact of 5,6 and 7 views on the results in **Table 4**. For fairness of comparison we use 6 views as training parameters, and testing with different parameters. Experimental results show that AdaptMVSNet requires fewer views for the same effect.

Normal Loss Since depth only constrains the output geometry in a single dimension, and smooth surfaces such as tabletops are common in indoor scenes, we verified the impact of adding the normal loss function on the results. It can be seen that under the use of normal loss, the accuracy error is reduced by 5.1 percent, confirming our hypothesis.

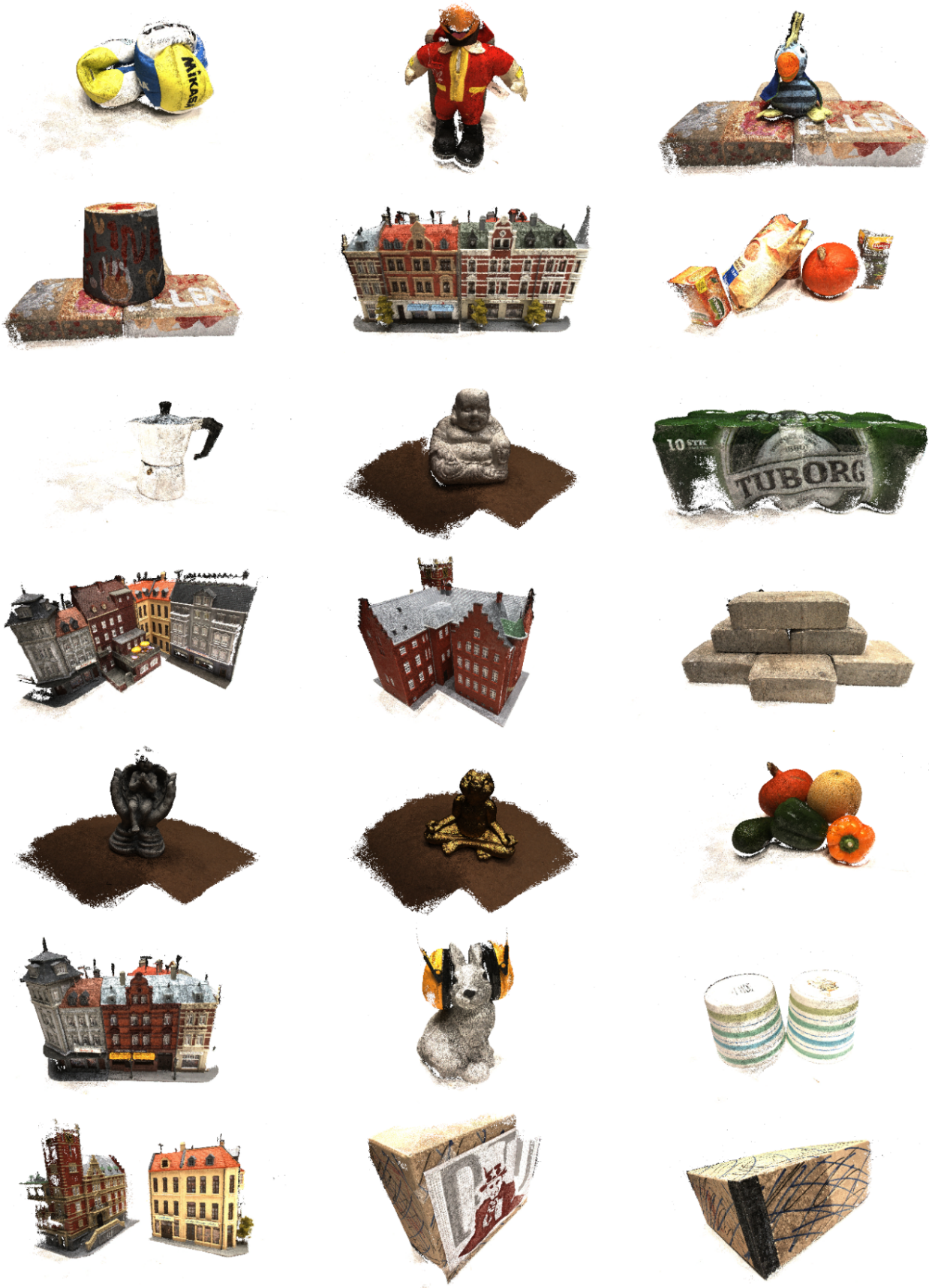


Fig. 7. Qualitative results of our method on DTU.

4.3.1. Adaptive Convolution vs Deformable Convolution

To further evaluate the effectiveness of the ACM, we compared the deformable convolution used in PatchMatchNet with our proposed ACM. We conducted experiments using different combinations of ACM and deformable convolutions on the depth map and loss volume. As shown in the experimental results in **Table 3**, using the ACM module outperforms using the deformable convolution module in terms of depth propagation and loss evaluation. The ACM module only adaptively adjusts the weight values of the convolution kernels based on the content of the feature map, which can improve the model's representation and generalization capabilities without increasing the model's computational cost, while the deformable convolution module requires non-parallelism in the convolution position change. Therefore, the ACM module has better processing capability and higher performance and is suitable for feature extraction and information fusion in depth estimation tasks.

5. Conclusion

This paper introduced an improved depth estimation model that uses adaptive convolution modules, attention fusion modules, and normal loss functions to improve the model's feature extraction ability and accuracy. In extensive experiments, we demonstrated the significant impact of these proposed modules on the depth estimation results. The flexibility and learning ability of adaptive convolution modules improved the model's adaptability and generalization ability. The attention fusion module reduced the influence of local feature deviation and noise, and improved the model's modeling ability and accuracy toward complex scenes, while the introduction of the normal loss function strengthens the model's constraint on geometric information. In addition, we compared the performance of adaptive convolution modules and deformable convolution modules, showing that adaptive convolution modules have better performance in depth propagation and loss evaluation.

For limitation, there are still many potential directions for improvement that are worth exploring, such as more effective model dynamic adaptation mechanisms, more reliable feature selection, and more accurate annotated datasets. We believe that this research direction still has enormous development potential.

References

[1] Bleyer, M, Rhemann, C, Rother, C. Patchmatch stereo-stereo matching with slanted support windows. In: Bmvc; vol. 11. 2011, p. 1–11.

[2] Schonberger, JL, Frahm, JM. Structure-from-motion revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, p. 4104–4113.

[3] Schops, T, Schonberger, JL, Galliani, S, Sattler, T, Schindler, K, Pollefeys, M, et al. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017, p. 3260–3269.

[4] Yao, Y, Luo, Z, Li, S, Fang, T, Quan, L. Mvsnet: Depth inference for unstructured multi-view stereo. In: Proceedings of the European conference on computer vision (ECCV). 2018, p. 767–783.

[5] Gu, X, Fan, Z, Zhu, S, Dai, Z, Tan, F, Tan, P. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, p. 2495–2504.

[6] Yang, J, Mao, W, Alvarez, JM, Liu, M. Cost volume pyramid based depth inference for multi-view stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 4877–4886.

[7] Mi, Z, Di, C, Xu, D. Generalized binary search network for highly-efficient multi-view stereo. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2022, p. 12981–12990. doi:10.1109/CVPR52688.2022.01265.

[8] Peng, R, Wang, R, Wang, Z, Lai, Y, Wang, R. Rethinking depth estimation for multi-view stereo: A unified representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 8645–8654.

[9] Wang, S, Li, B, Dai, Y. Efficient multi-view stereo by iterative dynamic cost volume. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2022, p. 8645–8654. doi:10.1109/CVPR52688.2022.00846.

[10] Yang, Z, Ren, Z, Shan, Q, Huang, Q. Mvs2d: Efficient multiview stereo via attention-driven 2d convolutions. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2022, p. 8564–8574. doi:10.1109/CVPR52688.2022.00838.

[11] Yu, Z, Gao, S. Fast-mvsnet: Sparse-to-dense multi-view stereo with learned propagation and gauss-newton refinement. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020, p. 1946–1955. doi:10.1109/CVPR42600.2020.00202.

[12] Dai, J, Qi, H, Xiong, Y, Li, Y, Zhang, G, Hu, H, et al. Deformable convolutional networks. In: 2017 IEEE International Conference on Computer Vision (ICCV). 2017, p. 764–773. doi:10.1109/ICCV.2017.89.

[13] Wang, F, Galliani, S, Vogel, C, Speciale, P, Pollefeys, M. Patchmatchnet: Learned multi-view patchmatch stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, p. 14194–14203.

[14] Sinha, SN, Mordohai, P, Pollefeys, M. Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. In: 2007 IEEE 11th International Conference on Computer Vision. IEEE; 2007, p. 1–8.

[15] Ulusoy, AO, Black, MJ, Geiger, A. Semantic multi-view stereo: Jointly estimating objects and voxels. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE; 2017, p. 4531–4540.

[16] Furukawa, Y, Ponce, J. Carved visual hulls for image-based modeling. In: Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9. Springer; 2006, p. 564–577.

[17] Zhaoxin Li, ea. Detail-preserving and content-aware variational multi-view stereo reconstruction. Transactions on Image Processing (TIP) 2016;69:026113.

[18] Furukawa, Y, Ponce, J. Accurate, dense, and robust multiview stereopsis. IEEE transactions on pattern analysis and machine intelligence 2009;32(8):1362–1376.

[19] Locher, A, Perdoch, M, Van Gool, L. Progressive prioritized multi-view stereo. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, p. 3244–3252.

[20] Galliani, S, Lasinger, K, Schindler, K. Massively parallel multiview stereopsis by surface normal diffusion. In: Proceedings of the IEEE International Conference on Computer Vision. 2015, p. 873–881.

[21] Xu, Q, Tao, W. Multi-scale geometric consistency guided multi-view stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, p. 5483–5492.

[22] Song, W, Li, Y, Zhu, J, Chen, C. Temporally-adjusted correlation filter-based tracking. Neurocomputing 2018;286:121–129. URL: https://www.sciencedirect.com/science/article/pii/S0925231218301000. doi:https://doi.org/10.1016/j.neucom.2018.01.067.

[23] Sun, S, Xu, D, Wu, H, Ying, H, Mou, Y. Multi-view stereo for large-scale scene reconstruction with mrf-based depth inference. Computers & Graphics 2022;106:248–258. URL: https://www.sciencedirect.com/science/article/pii/S009784932200111X. doi:https://doi.org/10.1016/j.cag.2022.06.009.

[24] Kar, A, Häne, C, Malik, J. Learning a multi-view stereo machine. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17; Red Hook, NY, USA: Curran Associates Inc. ISBN 9781510860964; 2017, p. 364–375.

[25] Ji, M, Gall, J, Zheng, H, Liu, Y, Fang, L. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In: 2017 IEEE International

Conference on Computer Vision (ICCV). 2017, p. 2326–2334. doi:10.1109/ICCV.2017.253.

[26] Xu, Q, Tao, W. Learning inverse depth regression for multi-view stereo with correlation cost volume. In: Proceedings of the AAAI Conference on Artificial Intelligence; vol. 34. 2020, p. 12508–12515.

[27] Yao, Y, Luo, Z, Li, S, Shen, T, Fang, T, Quan, L. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019, p. 5520–5529. doi:10.1109/CVPR.2019.00567.

[28] Cheng, S, Xu, Z, Zhu, S, Li, Z, Li, LE, Ramamoorthi, R, et al. Deep stereo using adaptive thin volume representation with uncertainty awareness. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 2524–2534.

[29] Cao, C, Ren, X, Fu, Y. Mvsformer: Multi-view stereo by learning robust image features and temperature-based depth. Transactions of Machine Learning Research 2022;.

[30] Dosovitskiy, A, Beyer, L, Kolesnikov, A, Weissenborn, D, Zhai, X, Unterthiner, T, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:201011929 2020;.

[31] Mi, Z, Di, C, Xu, D. Generalized binary search network for highly-efficient multi-view stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 12991–13000.

[32] Qi, Y, Su, W, Xu, Q, Tao, W. Sparse prior guided deep multi-view stereo. Computers & Graphics 2022;107:1–9. URL: https://www.sciencedirect.com/science/article/pii/S0097849322001157. doi:https://doi.org/10.1016/j.cag.2022.06.014.

[33] Li, Z, Zhang, X, Wang, K, Jiang, H, Wang, Z. High accuracy and geometry-consistent confidence prediction network for multi-view stereo. Computers & Graphics 2021;97:148–159. URL: https://www.sciencedirect.com/science/article/pii/S0097849321000625. doi:https://doi.org/10.1016/j.cag.2021.04.020.

[34] Lee, JY, DeGol, J, Zou, C, Hoiem, D. Patchmatch-rl: Deep mvs with pixelwise depth, normal, and visibility. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021, p. 6158–6167.

[35] Liao, J, Fu, Y, Yan, Q, Luo, F, Xiao, C. Adaptive depth estimation for pyramid multi-view stereo. Computers & Graphics 2021;97:268–278. URL: https://www.sciencedirect.com/science/article/pii/S0097849321000583. doi:https://doi.org/10.1016/j.cag.2021.04.016.

[36] Lin, TY, Dollár, P, Girshick, R, He, K, Hariharan, B, Belongie, S. Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 2117–2125.

[37] Chen, W, Xie, D, Zhang, Y, Pu, S. All you need is a few shifts: Designing efficient convolutional neural networks for image classification. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019, p. 7234–7243. doi:10.1109/CVPR.2019.00741.

[38] Sayed, M, Gibson, J, Watson, J, Prisacariu, V, Firman, M, Godard, C. Simplerecon: 3d reconstruction without 3d convolutions. In: Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII. Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-031-19826-7; 2022, p. 1–19. URL: https://doi.org/10.1007/978-3-031-19827-4_1. doi:10.1007/978-3-031-19827-4_1.

[39] Cheng, S, Xu, Z, Zhu, S, Li, Z, Li, LE, Ramamoorthi, R, et al. Deep stereo using adaptive thin volume representation with uncertainty awareness. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020, p. 2521–2531. doi:10.1109/CVPR42600.2020.00260.

[40] Yin, W, Liu, Y, Shen, C, Yan, Y. Enforcing geometric constraints of virtual normal for depth prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019, p. 5684–5693.

[41] Knapitsch, A, Park, J, Zhou, QY, Koltun, V. Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Trans Graph 2017;36(4). URL: https://doi.org/10.1145/3072959.3073599. doi:10.1145/3072959.3073599.

[42] Pytorch, ADI. Pytorch. 2018.

[43] Xu, Q, Tao, W. Pvsnet: Pixelwise visibility-aware multi-view stereo network. arXiv preprint arXiv:200707714 2020;.

[44] Kingma, DP, Ba, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980 2014;.

[45] Chen, R, Han, S, Xu, J, Su, H. Point-based multi-view stereo network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019, p. 1538–1547.

[46] Ji, M, Gall, J, Zheng, H, Liu, Y, Fang, L. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In: Proceedings of the IEEE International Conference on Computer Vision. 2017, p. 2307–2315.