Adaptive Patch Deformation for Textureless-Resilient Multi-View Stereo

Yuesong Wang Zhaojie Zeng* Tao Guan Wei Yang Zhuo Chen Wenkai Liu
Luoyuan Xu
School of Computer Science & Technology, Huazhong University of Science & Technology
{yuesongwang,zhaojiezeng,qd_gt,weiyangcs,cz_007,wenkai_liu,xu_luoyuan}@hust.edu.cn

Yawei Luo
School of Computer Science & Technology, Zhejiang University
yaweiluo329@gmail.com

Abstract

In recent years, deep learning-based approaches have shown great strength in multi-view stereo because of their outstanding ability to extract robust visual features. However, most learning-based methods need to build the cost volume and increase the receptive field enormously to get a satisfactory result when dealing with large-scale textureless regions, consequently leading to prohibitive memory consumption. To ensure both memory-friendly and textureless-resilient, we innovatively transplant the spirit of deformable convolution from deep learning into the traditional PatchMatch-based method. Specifically, for each pixel with matching ambiguity (termed ***unreliable*** *pixel), we adaptively deform the patch centered on it to extend the receptive field until covering enough correlative* ***reliable*** *pixels (without matching ambiguity) that serve as anchors. When performing PatchMatch, constrained by the anchor pixels, the matching cost of an unreliable pixel is guaranteed to reach the global minimum at the correct depth and therefore increases the robustness of multi-view stereo significantly. To detect more anchor pixels to ensure better adaptive patch deformation, we propose to evaluate the matching ambiguity of a certain pixel by checking the convergence of the estimated depth as optimization proceeds. As a result, our method achieves state-of-the-art performance on ETH3D and Tanks and Temples while preserving low memory consumption.*

1. Introduction

Multi-view stereo (MVS) is one of the core tasks in computer vision which aims to recover the 3D geometry of a scene using images captured from different viewpoints. It has been playing an essential role in many downstream tasks, such as automatic drive and virtual reality. Plentiful ideas stem from this vein [5, 13, 20, 31, 33] and continuously boost the reconstruction performances to a new level. These prior arts can be roughly divided into traditional and deep learning-based methods.

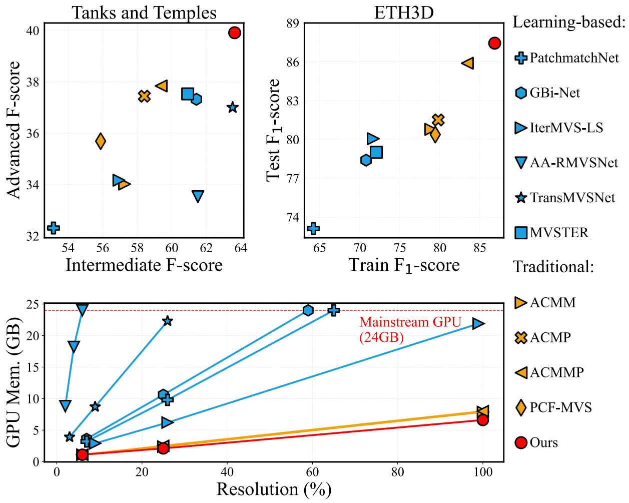


Figure 1. Comparison with the latest learning-based methods [6, 19, 27–29, 32] and traditional methods [12, 36, 37, 39] on Tanks and Temples [10] and ETH3D [23]. When comparing memory cost, we set the number of source images to 10 for all methods and the image size $6,221 \times 4,146$ (ETH3D) as 100% resolution (8.04% corresponds to Tanks and Temples). Note that learning-based methods train their models on DTU [1] or BlendedMVS [44] and only regard the train set of ETH3D as one of their test sets.

Many existing newly-proposed traditional MVS methods [22, 37, 39, 47] are extended versions of the PatchMatch [2] (PM), which calculate the matching cost between

[*] Corresponding author. Contact him at whoiszzj@outlook.com or zhaojiezeng@hust.edu.cn. Code: https://github.com/whoiszzj/APD-MVS.

a fixed-size reference patch and patches in source images according to a plane hypothesis. These PM-based methods avoid the construction of cost volume as they employ a propagation and local refinement strategy to find proper matches and hence require little memory. Nevertheless, according to [14], when a patch lies in a textureless region, the matching cost will lose credibility since there is no useful feature information in the receptive field. To mitigate this problem, attempts [14, 30, 37, 40] either downsample images or use multiple window sizes to increase the receptive field. However, they can only handle small areas of textureless regions well. To better cope with large-scale textureless regions, methods such as ACMP [39], PCF-MVS [12], and ACMMP [36] introduce a coarse fitting plane hypothesis to the textureless region. Nevertheless, such an approach suffers from gradual deviation from the provided plane hypothesis, leading to inaccurate depth estimation.

On the contrary, deep learning-based methods [15–17, 34, 38] usually suffer less from the above issue. Benefiting from the prevalent application of convolution operation, the receptive field of these methods is much larger than traditional ones. AA-RMVSNet [32] and TransMVSNet [6] further expand the receptive field by introducing deformable convolution [4]. As the receptive field increases, unreliable pixels can obtain adequate geometrical information from surrounding reliable pixels, which results in better depth estimation. Nevertheless, as shown in Fig. 1, a larger receptive field results in more memory consumption, making them hard to handle datasets with large-scale textureless regions or high-resolution images using mainstream GPU devices. Although several recent works have endeavored to reduce the memory consumption [19, 27, 28], the results are still not satisfactory compared with traditional methods [36, 39].

To develop a memory-friendly solution that can well handle large-scale textureless regions at the same time, in this paper, we transplant the spirit of deformable convolution to a traditional PM-based MVS pipeline. Specifically, for each unreliable pixel, we adaptively deform its corresponding patch to extend the receptive field until covering enough reliable pixels, as shown in Fig. 2. We then use RANSAC to filter out unrelated reliable pixels (belonging to different geometry hyperthesis or gathered due to occlusion). The residual reliable pixels serve as anchor pixels for the deformable patch. Then we conduct PM based on the widely-used normalized cross-correlation (NCC) metric within this deformable patch. As demonstrated in Fig. 2, the profile of matching cost using our deformable PM reaches a salient single valley at the ground-truth depth and hence guides the unreliable pixels to find a correct match.

One remaining and non-trivial issue that affects the success of our adaptive patch deformation is how to evaluate pixel reliability. Many existing approaches [14, 21, 40] rely solely on the pixel's intensity, which is unreliable when facing repetitive texture or drastic changes in illumination that can also cause matching ambiguity. Others [36, 39] simply set a threshold for the pixel's matching cost to evaluate reliability. However, as mentioned before, the matching cost is unreliable in textureless regions, making it hard to set a proper threshold. Instead, we propose to evaluate the reliability of pixels by checking the convergence of estimated depth as optimization proceeds. Specifically, in each iteration, we use conventional PM to calculate the matching cost of each pixel within a neighboring window of the current depth and form a matching cost profile. Then we evaluate pixel reliability by analyzing the geometric features of the profile, including local and global minima. Our evaluation approach can help to find more anchor pixels while maintaining their credibility, bringing better adaptive patch deformation.

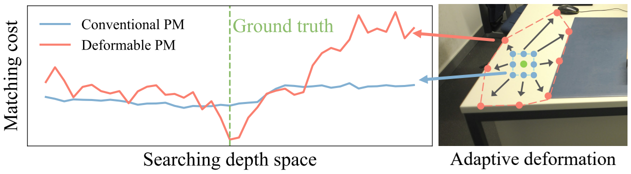


Figure 2. The right image is a demo of adaptive patch deformation. The green point represents the center pixel, the blue points around it represent the conventional patch, and the red points form the receptive field of our deformable patch. The left profile shows matching costs around the ground truth (green dashed line). Compared with conventional PM, our deformable PM has significant convergence performance around the ground truth for the unreliable pixel.

In summary, our contributions are as follows:

- For PM-based MVS, we propose to adaptively deform the patch of an unreliable pixel when computing the matching cost, which increases the receptive field when facing textureless regions to ensure robust matching.
- We propose to detect reliable pixels by checking the convergence of matching cost profiles, maintaining the accuracy of detection while being able to find more anchor pixels, which ensures better adaptive patch deformation.
- We realize a PM-based MVS method, APD-MVS, which adopts our adaptive patch deformation and an NCC-based matching metric. Our method achieves state-of-the-art results on ETH3D dataset and Tanks and Temples dataset with lower memory consumption.

2. Related Work

Traditional MVS Methods. According to the definition of [24], traditional MVS algorithms can be roughly cate-

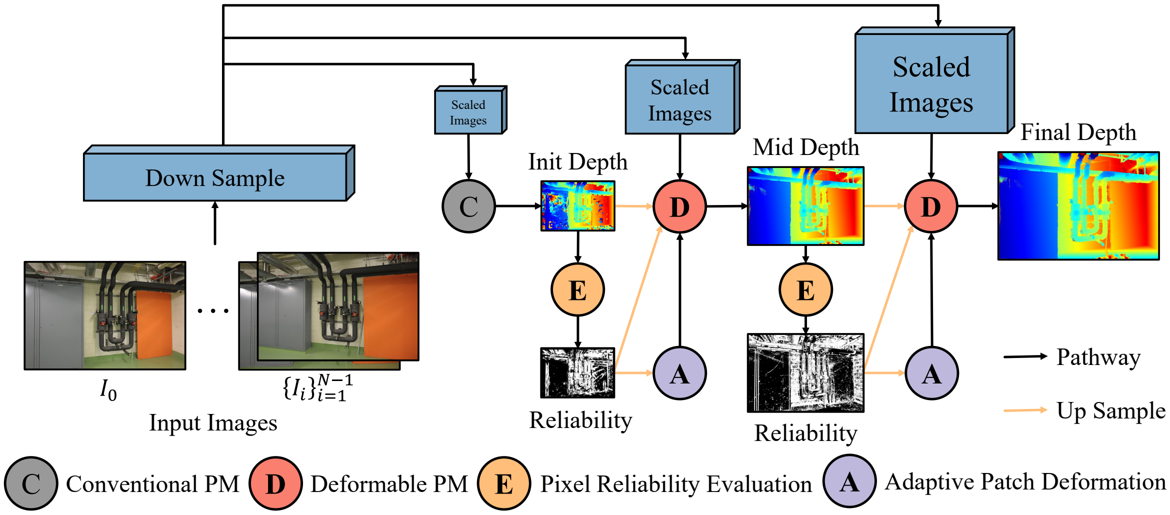


Figure 3. **Pipeline of APD-MVS**. This is a typical pyramid architecture. Conventional PM [36, 37] is applied at the coarsest layer to obtain the initial depth map. Given a current depth estimate, each pixel's reliability is evaluated (Sec. 3.4), which is used in adaptive patch deformation (Sec. 3.3). Based on the depth map and the deformable patches from the former layer, deformable PM (Sec. 3.2) is performed to get the depth estimates at a finer layer.

gorized into four classes: voxel-based [26], surface iterative evolution-based [3], depth map-based [2], and patch-based [7] methods. Among them, the depth map-based approach is favored for its wide applicability. In recent years, [8, 11, 14, 37] have raised the reconstruction effectiveness of depth map-based methods to a new level. ACMM [37] applies pyramid structure and geometric consistency into MVS, which makes it possible to efficiently process rich textured regions and small textureless regions while still cannot estimate the depth for large-scale textureless regions very well. Subsequently, [36, 39] provide a coarse fitting plane hypothesis for large-scale textureless regions by performing the Delaunay Triangulation of reliable pixels. [12, 21] segment images into superpixels and compute a fit plane for each superpixel. But as the optimization proceeds, the depth estimation gradually drifts away from the fitting plane hypothesis. MAR-MVS [40] calculates the gradient of intensity along the epipolar line to determine the patch size, which can adaptively increase the feature receptive field. However, the preset patch size cannot cope with various application scenarios.

Learning-based MVS Method. After [43] applied deep learning to depth map estimation, methods based on deep learning have sprung up. Researchers are dedicated to proposing a more reasonable way of expanding the receptive field to improve the results. By introducing a pyramid structure, methods such as [9, 35, 42, 45] use the depth range obtained from the previous layer to provide the initial value of depth for the next layer, further expanding the receptive field while reducing the size of cost volume. AA-RMVSNet [32] proposes an adaptive aggregation module implemented by deformable convolution to achieve a more robust feature extraction. [6, 29] go a step further by introducing the transformer structure into the MVS task to obtain global feature information. As opposed to traditional methods, which are always limited by the size of the receptive field, deep learning-based methods are more feature-aware. As a result, their reconstruction results often outperform traditional methods. However, even if numerous works [19, 27, 28] try to reduce GPU memory consumption, they still can not well handle large-scale scenes with high-resolution images such as ETH3D [23].

3. Method

3.1. Overview

Given a set of images $\{I_i\}_{i=1}^{M}$ and the corresponding camera parameters $\{\mathbf{P}_i\}_{i=1}^{M}$, our algorithm needs to estimate the depth map of each image. The algorithm pipeline is shown in Fig. 3.

For each reference image I_0 with its source images $\{I_i\}_{i=1}^{N-1}(N \leq M)$, we obtain a pyramid structure by scaling at different layers $\{L_i\}_{i=1}^{K}$, where the 1-st layer corresponds to the raw image. At the K-th layer, we adopt conventional PM to get the initial depth map, which is subsequently used to evaluate the reliability of each pixel (Sec. 3.4). For each unreliable pixel, we adaptively deform its corresponding patch to cover enough anchor pixels with high reliability (Sec. 3.3). At the middle layer, depths are inherited from the previous layer by upsampling. Depending on the reliability of the pixel, different matching strategies are applied. For reliable pixels, a conventional PM is applied, and for unreliable pixels, its match strategy is replaced by deformable PM (Sec. 3.2). Similarly, after obtaining the depth, the pixel's reliability and the deformable patch of the unreliable pixel are calculated again, which are then fed to the next layer. Finally, at the 1-st layer, depth estimations are fused to get a dense point cloud. In addition, we make some changes to propagation and refinement to better exploit pixels' reliability (Sec. 3.5).

3.2. Deformable PatchMatch

It is necessary to start with a review of the conventional PM method. Given a plane hypothesis, we can obtain the projected patch on the source image for a fixed-size patch centered on a pixel in the reference image. The matching cost between the two patches is usually calculated based on the NCC metric. Specifically, suppose that the reference image I_i with camera parameters $\mathbf{P}_i = \{\mathbf{K}_i, \mathbf{R}_i, \mathbf{C}_i\}$ has source image I_j with camera parameters $\mathbf{P}_j = \{\mathbf{K}_j, \mathbf{R}_j, \mathbf{C}_j\}$, where $\mathbf{K}$ is the intrinsic parameters, $\mathbf{R}$ is the rotation matrix, and $\mathbf{C}$ is the camera center. For a pixel p in I_i with homogeneous coordinate $\mathbf{p} = [u, v, 1]^T$, suppose its plane hypothesis is given by $\mathbf{f}_p = [\mathbf{n}^T, d]^T$, where $\mathbf{n}$ represents the plane's normal and d is the depth. According to [25], we can define homography as

$$\mathbf{H}_{ij} = \mathbf{K}_j(\mathbf{R}_j\mathbf{R}_i^{-1} + \frac{\mathbf{R}_j(\mathbf{C}_i - \mathbf{C}_j)\mathbf{n}^T}{\mathbf{n}^T d \mathbf{K}_i^{-1}\mathbf{p}})\mathbf{K}_i^{-1}. \quad (1)$$

We set a square window $\mathbf{B}_p$ centered on pixel p to represent the reference patch. For $\mathbf{B}_p$, we can find its corresponding patch $\mathbf{B}_p^j$ in the source image I_j using $\mathbf{H}_{ij}$.The matching cost is computed as one minus the NCC score

$$m_j(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_p) = 1 - \frac{cov(\mathbf{B}_p, \mathbf{B}_p^j)}{\sqrt{cov(\mathbf{B}_p, \mathbf{B}_p)cov(\mathbf{B}_p^j, \mathbf{B}_p^j)}}, \quad (2)$$

where $cov(\mathbf{X}, \mathbf{Y}) = E(x - E(x))E(y - E(y))$, x and y is the color value in $\mathbf{X}$ and $\mathbf{Y}$, $E(\cdot)$ is the expected value. Then we can get $m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_p)$ considering all the source images by aggregation using view weights [37].

Unliking the conventional PM that only considers pixels in the square window, we propose the deformable PM, which calculates the matching cost for an unreliable pixel within a deformable patch that covers enough anchor pixels with high reliability. Suppose that $\mathbf{p}$ is an unreliable pixel with a deformable patch that contains anchor pixels $\mathbf{S}$. Given the plane hypothesis $\mathbf{f}_p$, we define the matching cost computed by deformable PM as

$$\begin{aligned} m_D(\mathbf{p}, \mathbf{f}_p, \mathbf{S}) =& \lambda m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_p) \\ &+ (1 - \lambda)\frac{\sum_{\mathbf{s}\in\mathbf{S}} m(\mathbf{s}, \mathbf{f}_p, \mathbf{B}_s)}{|\mathbf{S}|}, \end{aligned} \quad (3)$$

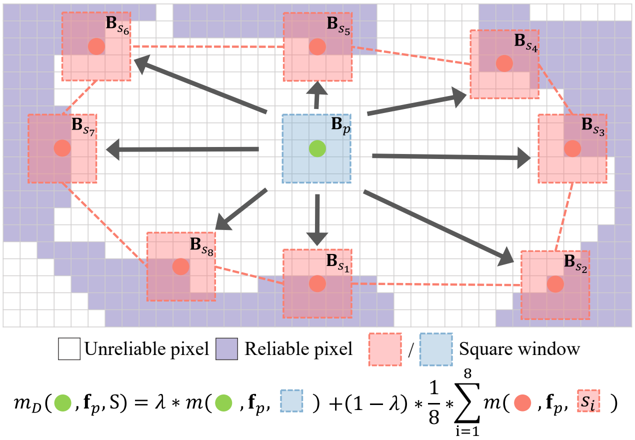

$\mathbf{B}_{s_6}$
$\mathbf{B}_{s_5}$
$\mathbf{B}_{s_4}$
$\mathbf{B}_{s_7}$
$\mathbf{B}_p$
$\mathbf{B}_{s_3}$
$\mathbf{B}_{s_8}$
$\mathbf{B}_{s_1}$
$\mathbf{B}_{s_2}$
□ Unreliable pixel ▨ Reliable pixel 🟥 / 🟦 Square window
$m_D(●, \mathbf{f}_p, \mathrm{S}) = \lambda * m(●, \mathbf{f}_p, 🟦) + (1 - \lambda) * \frac{1}{8} * \sum_{\mathrm{i}=1}^{8} m(●, \mathbf{f}_p, \boxed{s_i})$

Figure 4. **Visualization for computation of deformable PM.** The green point represents the central unreliable pixel. The red dashed lines form our deformable patch with anchor pixels $\mathbf{s}(i = 1...8)$ represented using red points. The matching cost of the center point is obtained by a weighted aggregation form.

where λ is a weight value used to adjust the effect of anchor pixels on the center pixel $\mathbf{p}$. $\mathbf{B}_p$'s window size is set to $w \times w$, the increment is set to θ, $\mathbf{B}_s$'s window size is the same as $\mathbf{B}_p$'s, but the increment is set to $\frac{w}{2}$, which speeds up computation. In experiments, we set $\lambda = 0.25$, $w = 11$, $\theta = 2$ and $|\mathbf{S}| = 8$.

The computation of $m_D(\mathbf{p}, \mathbf{f}_p, \mathbf{S})$ is visualized in Fig. 4. In order to obtain robust local features, we generate a local window $\mathbf{B}_s(\mathbf{s} \in \mathbf{S})$ for each anchor pixel $\mathbf{s}$ to get $m(\mathbf{s}, \mathbf{f}_p, \mathbf{B}_s)$. Then we apply separate calculations for each window $\mathbf{B}_p, \mathbf{B}_s(\mathbf{s} \in \mathbf{S})$, which are finally aggregated by weight to retain more feature information. The reason why we adopt an aggregation by weight instead of calculating $m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_{all}), \mathbf{B}_{all} = \mathbf{B}_p \cup \{\mathbf{B}_s | \mathbf{s} \in \mathbf{S}\}$ is that usually the number of unreliable pixels contained in $\mathbf{B}_{all}$ will exceed that of reliable pixels. Directly calculating $m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_{all})$ will result in the feature information from reliable pixels being filtered out as noise (More experiments in Sec. 4.5).

3.3. Adaptive Patch Deformation

Patches need to be adaptively deformed in advance to facilitate the calculation of the matching cost for unreliable pixels. When deforming a patch, the following principles should be observed as much as possible:

- The deformable patch of an unreliable pixel should adaptively cover enough nearby anchor pixels;
- Depths in the deformable patch should be continuous, ensuring pixels in the patch are correlative;
- Anchor pixels should try to be closer to the unreliable pixel to provide a better-fitting deformable patch;
- Anchor pixels should be found in various directions to increase the robustness of deformable PM.

Thus, we propose an adaptive patch deformation algorithm using a spoke-like approach and RANSAC to satisfy the above requirements. To facilitate the subsequent processing, the nearest reliable pixel for each pixel is obtained in advance as

$$\mathcal{N}(\mathbf{p}) = \begin{cases} \mathbf{p} & if \mathbb{R}(\mathbf{p}) = 1; \\ \underset{\mathbf{q}\in\boldsymbol{\Omega},\mathbb{R}(q)=1}{\text{argmin}} \ ||\mathbf{q}-\mathbf{p}|| & if \mathbb{R}(\mathbf{p}) = 0, \end{cases} \tag{4}$$

where Ω is a search range centered on $\mathbf{p}$ (in experiments, we define Ω as a 100×100 square window), $\mathbb{R}(\mathbf{p})$ is an indicator for $\mathbf{p}$, where 1 indicates reliable while 0 means unreliable.

The main idea of our algorithm is to obtain ϕ candidate reliable pixels $\{\mathbf{C}_i\}_{i=1}^{\phi}$, and then retain the well-adapted ones by RANSAC. When searching for candidate pixels for central pixel $\mathbf{p}$ in all directions, we adopt a spoke-like approach that slices the searching space into ϕ sectors with the same angle. For each sector, given an initial searching radius, a random direction vector in this sector is generated to obtain the searched pixel $\mathbf{q}$. If a valid $\mathcal{N}(\mathbf{q})$ exists and $\mathcal{N}(\mathbf{q})$ is within the sector, mark that a suitable candidate reliable pixel has been found in this sector. Otherwise, several more random searching under this radius is performed. If no candidate is still found, enlarge the radius and repeat the above process as shown in Fig. 5. This algorithm ensures that one reliable pixel exists in each direction and that candidate pixels are as close to the center as possible.

After obtaining $\{\mathbf{C}_i\}_{i=1}^{\phi}$, the filtering process is performed by the RANSAC algorithm, which aims to improve the anti-occlusion ability. Since our method is based on PM, the pixels in the deformable patch are implicitly required to have the same plane hypothesis. If there are candidates that can not fit into an accordant plane, we consider them outliers. Thus, for each iteration, three candidate pixels are randomly sampled, and the fitting plane π is formed by their 3D points. The central pixel is required to be inside the triangle formed by these three candidate pixels, ensuring that anchor pixels are in various directions of the central pixel. After that, the distances $\{D_i\}_{i=1}^{\phi}$ between π and the 3D points corresponding to the candidate pixels are calculated. Then the $\mathbf{cost}(\pi)$ for this random sample is obtained by

$$\mathbf{cost}(\pi) = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{\phi} \mathbb{I}(D_i > \varepsilon) \\ D_p \end{bmatrix}, \tag{5}$$

where ε is a threshold to filter outliers, $\mathbb{I}(\cdot)$ is an indicator function such that $\mathbb{I}(true) = 1$ and $\mathbb{I}(false) = 0$, D_p is the distance of center pixel $\mathbf{p}$'s 3D point to the fitting plane π. When comparing costs of different selections, α is first considered, with smaller α representing a better planar fit. If two costs are equal in the α dimension, then the effect of β is considered. After a certain number of iterations, if the best-fitting plane π_{best} can be found (sufficient points distributing near the plane), up to $|\mathbf{S}|$ reliable pixels from $\{\mathbf{C}_i | D_i \leq \varepsilon, i = 1...\phi\}$ are selected based on their distance to the fitting plane π_{best}. Then they form a deformable patch. Otherwise, the central pixel may lie in a non-planar area. In this case, we remain to adopt conventional PM to obtain the matching cost.

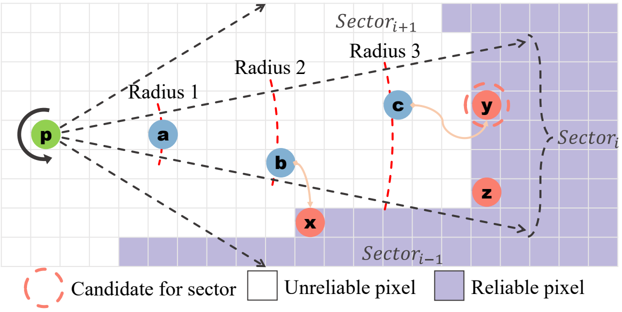


Figure 5. **A demo of spoke-like searching.** For the unreliable pixel p, the black dashed lines form the searching sectors. The red dashed curves represent different searching radii. For $Sector_i$, at radius 1, unreliable pixel a doesn't have nearby reliable pixel $\mathcal{N}$(a). At radius 2, b has $\mathcal{N}$(b)=x, but x is out of this sector. At radius 3, c has reliable pixel y and z nearby, since y is closer to c than z, so $\mathcal{N}$(c)=y, which will be selected as the candidate pixel for this sector.

To better cope with non-planar areas, we initially relax the planar fit threshold ε, treating them as planar areas, and gradually shrink the fitting threshold as the optimization proceeds so that the non-planar areas can get rid of the imposed planar constraint. By such means, it is easier for non-planar areas to find the correct depth estimation under a good initialization from the deformable PM. In experiments, we set $\phi = 32$ and gradually decrease ε from 1% depth range of the scene to 0.5%.

3.4. Pixel Reliability Evalution

Now there is only one problem left: how to evaluate the reliability of pixels. It is inappropriate to strictly divide pixels into two categories (reliable and unreliable) at the beginning. As optimization proceeds, the depth estimation of more pixels will fall into the searching range where no matching ambiguity exists, making them stable enough to serve as anchor pixels for deformable PM. More anchor pixels can bring a better-fitting deformable patch, further improving depth estimation accuracy. Thus we propose a mechanism for pixel reliability evaluation by checking the distribution of matching costs with the proceeding of optimization. Specifically, for each pixel, we use the matching costs computed by conventional PM in the vicinity of the current depth estimation to form a matching cost profile.

Algorithm 1: Pixel reliability evaluation

```
input  : Profile $\mathcal{P}(\mathcal{D})$ for pixel $p$, $\mathcal{D}_\prime$ and $\eta$
output: pixel $p$'s state
$\{\mathcal{P}(\mathcal{D}_{\mathcal{LM}})\} \leftarrow$ FindLMins($\mathcal{P}(\mathcal{D})$);
$\mathcal{P}(\mathcal{D}_{\mathcal{GM}}) \leftarrow$ FindGMin($\mathcal{P}(\mathcal{D})$);
if $|\mathcal{D}_{\mathcal{GM}} - \mathcal{D}_\prime| > \eta$ or $\mathcal{P}(\mathcal{D}_{\mathcal{GM}}) > t_1$ then
    return $Unreliable$;
if $|\{\mathcal{P}(\mathcal{D}_{\mathcal{LM}})\}| == 1$ then
    if $\mathcal{P}(\mathcal{D}_{\mathcal{GM}}) < t_2$ then return $Reliable$;
    else return $Unreliable$;
var $\leftarrow 0$;
foreach $\mathcal{P}$ in $\{\mathcal{P}(\mathcal{D}_{\mathcal{LM}})\}$ do
    var += $(\mathcal{P} - \mathcal{P}(\mathcal{D}_{\mathcal{GM}}))^2$;
var = $\sqrt{\text{var}}/(|\{\mathcal{P}(\mathcal{D}_{\mathcal{LM}})\}| - 1)$;
if var $> t_3$ then return $Reliable$;
else return $Unreliable$;
```

Then the reliability is evaluated by checking the valleys and convergence of the profile.

Since the depth range varies in different scenes, sampling directly on it doesn't have universality, so we perform the sampling operation under disparity space. For each pixel $\mathbf{p}$ in the reference image I_0, given the plane hypothesis $\mathbf{f}_p = [\mathbf{n}^T, d]^T$ and the camera focal length f, the average disparity is calculated as

$$\mathcal{D}_\prime = \frac{f * E(b_{I_j})}{d}, I_j \in S_{good}, \tag{6}$$

where S_{good} is a subset of $\{I_i\}_{i=1}^{N-1}$, which is selected by joint view selection [37], b_{I_j} is defined as the length of baseline between reference image I_0 and source image I_j, $E(\cdot)$ is the expected value. Then we compute the matching cost by conventional PM to get a profile $\mathcal{P}(\mathcal{D})$

$$\begin{aligned} &\mathcal{P}(\mathcal{D}) = m(\mathbf{p}, [\mathbf{n}^T, \frac{f * E(b_{I_j})}{\mathcal{D}}]^T, \mathbf{B}), \\ &\mathcal{D} \in [\mathcal{D}_\prime - \delta, \mathcal{D}_\prime + \delta], I_j \in S_{good}, \end{aligned} \tag{7}$$

where $\mathbf{B}$ is a conventional square window centered on p, δ is the sample range. After obtaining the matching cost profile, we can evaluate the pixel's reliability according to the geometric properties of the matching cost profile, as shown in Algo. 1. The main idea of the algorithm is to calculate the distinctiveness of the global minimum compared with other local minimums. For reliable pixels, the global minimum should be more distinctive than that of unreliable pixels.

We classify the pixels into two states: $Reliable$ and $Unreliable$. Function $FindLMins$ aims to find local minimums in the profile, and $FindGMin$ aims to find the global minimum. Parameters t_1, t_2, t_3 are the threshold, which we set to $0.50, 0.15, 0.20$, respectively, in our experiments. η ($< \delta$) is a threshold to determine whether the optimization of depth estimation has converged. As the optimization proceeds, η will be dynamically reduced to make the determination of reliable pixels more stringent. In experiments, we set $\delta = 30$ and $\eta = max(6 - 2 * i, 2)$, where i is the iteration number.

3.5. Propagation and Refinement

We improve the propagation and refinement process based on [37] to better utilize the deformable patch information of unreliable pixels.

When performing joint view selection on unreliable pixels, we directly use anchor pixels in the deformable patch to calculate the cost matrix M proposed by [37], which improves the robustness of view selection.

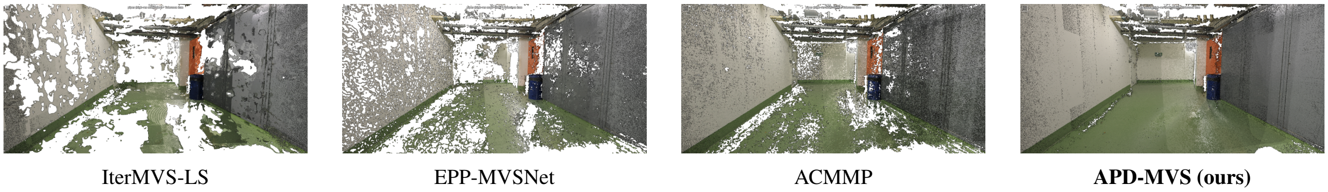
During propagation, the candidate plane hypotheses propagated to the unreliable pixel are replaced by those of anchor pixels. Besides, the fitting plane hypothesis from the deformable patch is added to the candidates in the refinement step to speed up the convergence. The above manners make the depth estimation of unreliable pixels partially rely on that of reliable pixels. Thus, the reliable pixels are processed first, and later for the unreliable pixels in each iteration.

When the optimization for depth estimation of a truly reliable pixel has not converged yet, i.e., $|\mathcal{D}_{\mathcal{GM}} - \mathcal{D}_\prime| > \eta$ in Algo. 1, the reliable pixel will adopt deformable PM as if it were an unreliable pixel. On the one hand, this is to exploit the surrounding information to speed up convergence and, on the other hand, to prevent such kind of reliable pixels from impairing the unreliable pixel's adaptive patch deformation. However, imposed planar constraints will cause a decrease in accuracy for depth estimation at such kinds of pixels. So a local refinement is added at the end of the optimization. By performing a small sampling around the original depth, we obtain the optimum depth based on the cost value from the conventional PM. If the cost of the optimum depth is much smaller than that of the original depth, the optimum depth will be adopted, increasing the accuracy at truly reliable pixels while having little influence on those truly unreliable pixels.

4. Experiments

4.1. Datasets and Implementation

Datasets. To verify the effectiveness of our method, we use ETH3D [23] dataset and Tanks and Temples [10] dataset in our experiments. The ETH3D dataset is used to test the performance when processing large-scale scenes with high-resolution images. The Tanks and Temples dataset also contains large-scale scenes but has a smaller resolution (about 1,920 × 1,080) and we use this dataset to demonstrate the



IterMVS-LS

EPP-MVSNet

ACMMP

APD-MVS (ours)

Figure 6. **Qualitative results of pipes on ETH3D.** It is obvious that the completeness of our results is better.

Method	Train						Test					
	2cm			10cm			2cm			10cm		
	F_1	Comp	Acc	F_1	Comp	Acc	F_1	Comp	Acc	F_1	Comp	Acc
PatchmatchNet [28]	64.21	65.43	64.81	85.70	83.28	89.98	73.12	77.46	69.71	91.91	92.05	91.98
GBi-Net [19]	70.78	69.21	73.17	90.21	86.16	95.21	78.40	75.65	82.02	91.35	86.67	96.99
IterMVS-LS [27]	71.69	66.08	79.79	88.60	82.62	96.35	80.06	76.49	84.73	92.29	88.34	96.92
MVSTER [29]	72.06	76.92	68.08	91.73	91.91	91.97	79.01	82.47	77.09	93.20	92.71	94.21
EPP-MVSNet [18]	74.00	67.58	82.76	92.13	87.72	97.29	83.40	81.79	85.47	95.22	93.75	96.84
Colmap [22]	67.66	55.13	**91.85**	87.61	79.47	**98.75**	73.01	62.98	**91.97**	90.4	84.54	**98.25**
ACMM [37]	78.86	70.42	90.67	91.70	86.40	98.12	80.78	74.34	90.65	92.96	88.77	98.05
PCF-MVS [12]	79.42	75.73	84.11	92.98	90.42	95.98	80.38	79.29	82.15	91.56	91.26	92.12
MAR-MVS [40]	79.21	77.19	81.98	92.71	90.44	95.51	81.84	84.18	80.24	94.22	94.43	94.21
ACMP [39]	79.79	72.15	90.12	92.03	87.15	97.97	81.51	75.58	90.54	92.62	88.71	97.47
ACMMP [36]	83.42	77.61	90.63	95.54	93.32	97.99	85.89	81.49	91.91	96.27	94.67	98.05
APD-MVS (ours)	**86.84**	**84.83**	89.14	**97.12**	**96.79**	97.47	**87.44**	**85.93**	89.54	**96.95**	**96.95**	97.00

Table 1. **Quantitative results on ETH3D benchmark.** Best results are marked in bold. Our method ranks first in terms of the F_1-score.

generalization ability.

Implementation. We take [37] as our basline and form a pyramid structure by image scaling. The number of pyramid layers is related to the image resolution, divided into four layers on ETH3D and three on Tanks and Temples. At each layer, we perform four iterations on each image. We adopt an NCC-based matching metric when implementing our PM-based MVS method, APD-MVS.

4.2. Results on ETH3D

We followed the fusion method of [36] to obtain point clouds. Fig. 6 shows a qualitative comparison, and it is evident that our method achieves higher completeness while containing fewer outliers. Tab. 1 shows the quantitative analysis, where the first group is learning-based and the second is traditional. Methods such as [12, 36] can not guarantee the convergence of the optimization for depth estimation in textureless regions since they only regard the fitting plane hypothesis as prior information, resulting in an incomplete point cloud. Furthermore, due to the high resolution of images, only a few learning-based methods can accomplish the reconstruction task on this dataset, such as [18, 19, 28, 46]. However, their performance is still not satisfactory.

4.3. Results on Tanks and Temples

There are more learning-based methods tested on this dataset, and to ensure fairness, we use the fusion strategy of learning-based methods [20, 41]. Tab. 2 shows the quantitative analysis. We have the best performance among the latest traditional methods. Meanwhile, our method can achieve competitive performance compared with the latest learning-based methods. Especially when dealing with textureless scenes, such as Horse and Auditorium, our method can exceed learning-based methods significantly. Qualitative results are in supplementary materials.

4.4. Memory Comparison

We test the memory cost on NVIDIA TITAN. As shown in Fig. 1, despite the effort of [19, 27, 28], learning-based methods still can not well balance memory consumption and reconstruction results. On the contrary, traditional methods typically don't cost much GPU memory. Tab. 3 gives a quantitative comparison. CasMVSNet [9] is a widely-used learning-based baseline, while Patchmatch-Net [28], GBi-Net [19] and IterMVS [27] are designed to reduce memory consumption. Compared with the latest traditional method ACMMP [36], the performance of these learning-based methods is still unsatisfactory. Considering all these latest methods, our APD-MVS can achieve lower memory consumption and better reconstruction results.

4.5. More Experiments

The weighted aggregation form of deformable PM. As mentioned in Sec. 3.2, $m_D(\mathbf{p}, \mathbf{f}_p, \mathbf{S})$, termed weighted

Method	Intermediate									Advanced						
	Mean	Fam.	Fra.	Hor.	Lig.	M60.	Pan.	Pla.	Tra.	Mean	Aud.	Bal.	Cou.	Mus.	Pal.	Tem.
PatchmatchNet [28]	53.15	66.99	52.64	43.24	54.87	52.87	49.54	54.21	50.81	32.31	23.69	37.73	30.04	41.80	28.31	32.29
CasMVSNet [9]	56.84	76.37	58.45	46.26	55.81	56.11	54.06	58.18	49.51	31.12	19.81	38.46	29.10	43.87	27.36	28.11
AA-RMVSNet [32]	61.51	77.77	59.53	51.53	64.02	**64.05**	59.47	60.85	54.90	33.53	20.96	40.15	32.05	46.01	29.28	32.71
EPP-MVSNet [18]	61.68	77.86	60.54	52.96	62.33	61.69	**60.34**	**62.44**	55.30	35.72	21.28	39.74	35.34	49.21	30.00	38.75
GBi-Net [19]	61.42	79.77	**67.69**	51.81	61.25	60.37	55.87	60.67	53.89	37.32	29.77	42.12	36.30	47.69	31.11	36.93
MVSTER [29]	60.92	80.21	63.51	52.30	61.38	61.47	58.16	58.98	51.38	37.53	26.68	42.14	35.65	49.37	32.16	39.19
TransMVSNet [6]	63.52	**80.92**	65.83	56.94	62.54	63.06	60.00	60.20	**58.67**	37.00	24.84	**44.59**	34.77	46.49	**34.69**	36.62
Colmap [22]	42.14	50.41	22.25	25.63	56.43	44.83	46.97	48.53	42.04	27.24	16.02	25.23	34.70	41.51	18.05	27.94
PCF-MVS [12]	55.88	70.99	49.60	40.34	63.44	57.79	58.91	56.59	49.40	35.69	28.33	38.64	35.95	48.36	26.17	36.69
ACMM [37]	57.27	69.24	51.45	46.97	63.20	55.07	57.64	60.08	54.48	34.02	23.41	32.91	41.17	48.13	23.87	34.60
ACMP [39]	58.41	70.30	54.06	54.11	61.65	54.16	57.60	58.12	57.25	37.44	30.12	34.68	**44.58**	50.64	27.20	37.43
ACMMP [36]	59.38	70.93	55.39	51.80	63.83	55.94	59.47	59.51	58.20	37.84	30.05	35.36	44.51	50.95	27.43	38.73
APD-MVS (ours)	**63.64**	75.01	63.70	**65.22**	**65.84**	60.27	60.10	61.66	57.29	**39.91**	**32.54**	42.79	39.24	**51.03**	33.08	**40.77**

Table 2. **Quantitative results of F-score on Tanks and Temples benchmark.** Best results are marked in bold. The second-best results are marked with an underscore. Our results exceed existing traditional methods and are comparable to the latest learning-based ones.

Method	GPU Mem. (GB)		
	Res. (8.04%)	Res. (50.0%)	Res. (100%)
CasMVSNet [9]	7.8	-	-
GBi-Net [19]	3.6	20.7	-
PatchmatchNet [28]	3.5	18.6	-
IterMVS-LS [27]	2.5	11.2	22.0
ACMMP [36]	1.4	4.5	7.9
APD-MVS (ours)	1.4	3.7	6.6

Table 3. **Quantitative comparison of GPU memory cost at different resolution.** We set the resolution of ETH3D ($6,221 \times 4,146$) as 100%.

Method	2cm			10cm		
	F_1	Comp	Acc	F_1	Comp	Acc
APD-MVS / MF	83.73	81.43	86.39	95.68	94.91	96.51
APD-MVS / WF	**86.84**	**84.83**	**89.14**	**97.12**	**96.79**	**97.12**

Table 4. **Quantitative comparison of the weighted form (WF) and mixed form (MF) on ETH3D.**

form, is better than $m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_{all})$, termed mixed form. In experiments, we find that the optimization for depth estimation of some unreliable pixels can not converge when using $m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_{all})$, which can harm the performance of our APD-MVS. The reason behind the above result is that $\mathbf{B}_{all}$ usually contains far more unreliable pixels than reliable ones, calculating $m(\mathbf{p}, \mathbf{f}_p, \mathbf{B}_{all})$ will lead to the reliable pixels being treated as noise due to the anti-interference capability of NCC used in conventional PM, and the feature information will be lost. Via the weighted aggregation form, the feature information from reliable pixels can be retained appropriately, and the final matching result will be better. The comparison results are shown in Tab. 4.

Pixel reliability evaluation using the matching cost profile. We propose to utilize the geometric features of matching cost profiles to detect reliable pixels. In such a manner, more anchor pixels can be found, ensuring better-fitting deformable patches are generated. For comparison with other evaluation mechanisms combined with our APD-MVS, we calculate the colors' *standard deviation* in a fixed-size window to evaluate pixels' reliability similar to [21] (APD-MVS / SD) while regarding pixels with *matching costs under a certain threshold* [36] as reliable pixels (APD-MVS / MT). As shown in Tab. 5, the performances based on them are much worse than that based on our *matching cost profile analyzing* (APD-MVS / PA).

Method	2cm			10cm		
	F_1	Comp	Acc	F_1	Comp	Acc
APD-MVS / SD	81.57	77.19	87.11	94.46	92.21	**97.17**
APD-MVS / MT	76.71	76.52	77.54	91.66	93.01	90.60
APD-MVS / PA	**86.84**	**84.83**	**89.14**	**97.12**	**96.79**	97.12

Table 5. **Quantitative comparison of different reliability evaluation methods on ETH3D**. We combine our APD-MVS with matching cost profile analyzing (APD-MVS / PA), standard deviation (APD-MVS / SD), and matching cost under a certain threshold (APD-MVS / MT).

5. Conclusion

In this paper, we propose adaptive patch deformation. Based on that we realize a traditional PM-based MVS method, APD-MVS, which can be both textureless-resilient and memory-friendly. The SOTA performance with lower memory consumption proves that our approach is groundbreaking. However, a few problems still need to be solved when facing large-scale textureless curved surfaces, preventing further improvement. These may be solved in future work by introducing a curved surface assumption when conducting our deformable PM.

Acknowledgements Project funded by the National Key R&D Program of China (No: 2021YFF0500302) and China Post-doctoral Science Foundation (No: 2022T150567). The computation is completed in the HPC Platform of Huazhong University of Science and Technology.

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2):153–168, 2016. 1

[2] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo-stereo matching with slanted support windows. In *Bmvc*, volume 11, pages 1–11, 2011. 1, 3

[3] Daniel Cremers and Kalin Kolev. Multiview stereo and silhouette consistency via convex functionals over convex domains. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1161–1174, 2010. 3

[4] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 2

[5] Yikang Ding, Qingtian Zhu1 Xiangyue Liu1 Wentao Yuan, Haotian Zhang, and Chi Zhang. Kd-mvs: Knowledge distillation based self-supervised learning for multi-view stereo. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*, pages 630–646. Springer Nature Switzerland Cham, 2022. 1

[6] Yikang Ding, Wentao Yuan, Qingtian Zhu, Haotian Zhang, Xiangyue Liu, Yuanjiang Wang, and Xiao Liu. Transmvsnet: Global context-aware multi-view stereo network with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8585–8594, 2022. 1, 2, 3, 8

[7] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 3

[8] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 873–881, 2015. 3

[9] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. 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*, pages 2495–2504, 2020. 3, 7, 8

[10] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. 1, 6

[11] Andreas Kuhn, Heiko Hirschmüller, Daniel Scharstein, and Helmut Mayer. A tv prior for high-quality scalable multi-view stereo reconstruction. *International Journal of Computer Vision*, 124(1):2–17, 2017. 3

[12] Andreas Kuhn, Shan Lin, and Oliver Erdler. Plane completion and filtering for multi-view stereo reconstruction. In *German Conference on Pattern Recognition*, pages 18–32. Springer, 2019. 1, 2, 3, 7, 8

[13] Zhaoxin Li, Wangmeng Zuo, Zhaoqi Wang, and Lei Zhang. Confidence-based large-scale dense multi-view stereo. *IEEE Transactions on Image Processing*, 29:7176–7191, 2020. 1

[14] Jie Liao, Yanping Fu, Qingan Yan, and Chunxia Xiao. Pyramid multi-view stereo with local consistency. In *Computer Graphics Forum*, volume 38, pages 335–346. Wiley Online Library, 2019. 2, 3

[15] Keyang Luo, Tao Guan, Lili Ju, Yuesong Wang, Zhuo Chen, and Yawei Luo. Attention-aware multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1590–1599, 2020. 2

[16] Yawei Luo, Ping Liu, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Category-level adversarial adaptation for semantic segmentation using purified features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[17] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2507–2516, 2019. 2

[18] Xinjun Ma, Yue Gong, Qirui Wang, Jingwei Huang, Lei Chen, and Fan Yu. Epp-mvsnet: Epipolar-assembling based depth prediction for multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5732–5740, 2021. 7, 8

[19] Zhenxing Mi, Chang Di, and Dan Xu. Generalized binary search network for highly-efficient multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12991–13000, 2022. 1, 2, 3, 7, 8

[20] Rui Peng, Rongjie Wang, Zhenyu Wang, Yawen Lai, and Ronggang Wang. Rethinking depth estimation for multi-view stereo: A unified representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8645–8654, 2022. 1, 7

[21] Andrea Romanoni and Matteo Matteucci. Tapa-mvs: Textureless-aware patchmatch multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10413–10422, 2019. 2, 3, 8

[22] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision*, pages 501–518. Springer, 2016. 1, 7, 8

[23] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 3, 6

[24] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006. 2

[25] Shuhan Shen. Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. *IEEE transactions on image processing*, 22(5):1901–1914, 2013. 4

[26] George Vogiatzis, Carlos Hernández Esteban, Philip HS Torr, and Roberto Cipolla. Multiview stereo via volumet-

ric graph-cuts and occlusion robust photo-consistency. *IEEE transactions on pattern analysis and machine intelligence*, 29(12):2241–2246, 2007. 3

[27] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, and Marc Pollefeys. Itermvs: Iterative probability estimation for efficient multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8606–8615, 2022. 1, 2, 3, 7, 8

[28] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14194–14203, 2021. 1, 2, 3, 7, 8

[29] Xiaofeng Wang, Zheng Zhu, Guan Huang, Fangbo Qin, Yun Ye, Yijia He, Xu Chi, and Xingang Wang. Mvster: Epipolar transformer for efficient multi-view stereo. In *European Conference on Computer Vision*, pages 573–591. Springer, 2022. 1, 3, 7, 8

[30] Yuesong Wang, Tao Guan, Zhuo Chen, Yawei Luo, Keyang Luo, and Lili Ju. Mesh-guided multi-view stereo with pyramid architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2039–2048, 2020. 2

[31] Yuesong Wang, Keyang Luo, Zhuo Chen, Lili Ju, and Tao Guan. Deepfusion: A simple way to improve traditional multi-view stereo methods using deep learning. *Knowledge-Based Systems*, 221:106968, 2021. 1

[32] Zizhuang Wei, Qingtian Zhu, Chen Min, Yisong Chen, and Guoping Wang. Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6187–6196, 2021. 1, 2, 3, 8

[33] Junhua Xi, Yifei Shi, Yijie Wang, Yulan Guo, and Kai Xu. Raymvsnet: Learning ray-based 1d implicit fields for accurate multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8595–8605, 2022. 1

[34] Luoyuan Xu, Tao Guan, Yuesong Wang, Yawei Luo, Zhuo Chen, Wenkai Liu, and Wei Yang. Self-supervised multi-view stereo via adjacent geometry guided volume completion. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2202–2210, 2022. 2

[35] Luoyuan Xu, Yawei Luo, Keyang Luo, Yuesong Wang, Tao Guan, Zhuo Chen, and Wenkai Liu. Exploiting the structure information of suppositional mesh for unsupervised multi-view stereo. *IEEE MultiMedia*, 29(1):94–103, 2021. 3

[36] Qingshan Xu, Weihang Kong, Wenbing Tao, and Marc Pollefeys. Multi-scale geometric consistency guided and planar prior assisted multi-view stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 1, 2, 3, 7, 8

[37] Qingshan Xu and Wenbing Tao. Multi-scale geometric consistency guided multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5483–5492, 2019. 1, 2, 3, 4, 6, 7, 8

[38] Qingshan Xu and Wenbing Tao. Learning inverse depth regression for multi-view stereo with correlation cost volume. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12508–12515, 2020. 2

[39] Qingshan Xu and Wenbing Tao. Planar prior assisted patchmatch multi-view stereo. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12516–12523, 2020. 1, 2, 3, 7, 8

[40] Zhenyu Xu, Yiguang Liu, Xuelei Shi, Ying Wang, and Yunan Zheng. Marmvs: Matching ambiguity reduced multiple view stereo for efficient large scale scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5981–5990, 2020. 2, 3, 7

[41] Jianfeng Yan, Zizhuang Wei, Hongwei Yi, Mingyu Ding, Runze Zhang, Yisong Chen, Guoping Wang, and Yu-Wing Tai. Dense hybrid recurrent multi-view stereo net with dynamic consistency checking. In *European conference on computer vision*, pages 674–689. Springer, 2020. 7

[42] Jiayu Yang, Wei Mao, Jose M Alvarez, and Miaomiao Liu. Cost volume pyramid based depth inference for multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4877–4886, 2020. 3

[43] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 3

[44] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1790–1799, 2020. 1

[45] Hongwei Yi, Zizhuang Wei, Mingyu Ding, Runze Zhang, Yisong Chen, Guoping Wang, and Yu-Wing Tai. Pyramid multi-view stereo net with self-adaptive view aggregation. In *European Conference on Computer Vision*, pages 766–782. Springer, 2020. 3

[46] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network. *British Machine Vision Conference (BMVC)*, 2020. 7

[47] Enliang Zheng, Enrique Dunn, Vladimir Jojic, and Jan-Michael Frahm. Patchmatch based joint view selection and depthmap estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1517, 2014. 1