DELS-MVS: Deep Epipolar Line Search for Multi-View Stereo

Christian Sormann [1]

christian.sormann@icg.tugraz.at

Emanuele Santellani [1]

emanuele.santellani@icg.tugraz.at

Mattia Rossi [2]

mattia.rossi@sony.com

Andreas Kuhn [2]

andreas.kuhn@sony.com

Friedrich Fraundorfer [1]

fraundorfer@icg.tugraz.at

[1] Graz University of Technology
Institute of Computer Graphics and Vision

[2] Sony Europe B.V.
R&D Center - Stuttgart Laboratory 1

Abstract

We propose a novel approach for deep learning-based Multi-View Stereo (MVS). For each pixel in the reference image, our method leverages a deep architecture to search for the corresponding point in the source image directly along the corresponding epipolar line. We denote our method DELS-MVS: Deep Epipolar Line Search Multi-View Stereo. Previous works in deep MVS select a range of interest within the depth space, discretize it, and sample the epipolar line according to the resulting depth values: this can result in an uneven scanning of the epipolar line, hence of the image space. Instead, our method works directly on the epipolar line: this guarantees an even scanning of the image space and avoids both the need to select a depth range of interest, which is often not known a priori and can vary dramatically from scene to scene, and the need for a suitable discretization of the depth space. In fact, our search is iterative, which avoids the building of a cost volume, costly both to store and to process. Finally, our method performs a robust geometry-aware fusion of the estimated depth maps, leveraging a confidence predicted alongside each depth. We test DELS-MVS on the ETH3D, Tanks and Temples and DTU benchmarks and achieve competitive results with respect to state-of-the-art approaches.

1. Introduction

Multi-View Stereo (MVS) systems target the reconstruction of accurate and complete depth maps of multiple calibrated images capturing the same scene. This is a long standing research topic in computer vision, as it is at the basis of any 3D reconstruction pipeline. In recent years, deep learning based methods have been gaining a lot of attention in the field, due to their more robust learned feature representations and matching similarity measures [37, 27]. A large portion of these methods make use of depth-plane sweeping cost volumes, which have two distinct disadvan-
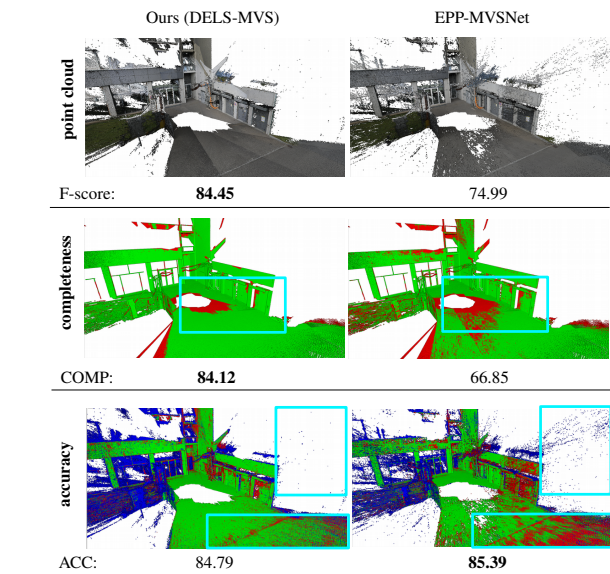

Figure 1. ETH3D high res [22] benchmark visualization for the electro scene. Our point cloud is more complete and less noisy.

tages. The first is their large memory footprint, which typically results in high computational processing costs as well. The second is the need to discretize the depth search space, which requires both to define a depth range of interest and to define a discretization scheme for it. Although the depth range can be inferred from the sparse reconstruction of a Structure from Motion (SfM) system, the sparse nature of the reconstruction can lead to under or over estimate the depth range of the portion of the scene captured by a specific image, thus preventing the reconstruction of some scene areas. Moreover, some discretization strategies suit certain scenes better than others. For objects close to the camera, a fine grained discretization toward the depth range minimum is preferable. Conversely, objects farther in the distance can be reconstructed even with a coarser discretization. The limitations behind depth discretization approaches becomes

obvious when considering scenes containing multiple objects at very different distances from the camera. A better alternative is represented by a discretization of the inverse depth range [26, 31], but this still calls for a depth range of interest. In order to tackle both of these disadvantages, we propose a new method for MVS depth estimation denoted Deep Epipolar Line Search (DELS). For each pixel in the reference image, we search for the matching point in the source image by exploring the corresponding epipolar line. The search procedure is carried out iteratively and utilizes bi-directional partitions of the epipolar line around the current estimate to update the predicted matching point. These updates are guided by a deep neural network, fed with features from both the reference and the source image.

Our approach has several advantages over those methods that first discretize a given depth range, chosen a priori, and later convert the resulting depth values into points or segments along the epipolar line, when looking for a match. First, operating on the epipolar line allows our method to exploit the image information better. In fact, due to the scene geometry and the relative pose between the reference and the source camera, a uniform discretization of the depth range could result in points clustered in a small segment of the epipolar line, thus preventing a correct match.

Second, our strategy avoids both the need to define a search depth range and the need for a depth discretization strategy customized for the scene content, as the epipolar line is explored dynamically. Our method is iterative and adopts a coarse-to-fine approach that permits to scan the epipolar line efficiently. This avoids the construction of a large fine grained depth cost volume. Finally, our method estimates a depth at the reference image for each available source one and fuses them in a geometry-aware fashion, using confidence measures estimated alongside the depth maps themselves. These confidence measures can be leveraged also during point cloud construction, in order to filter outliers, thus leading to more accurate reconstructions.

In summary, our core contributions are the following: (i) a deep, iterative and coarse-to-fine depth estimation algorithm operating directly on the epipolar lines, thus avoiding the drawbacks of depth discretization, e.g. no specification of a depth range is needed (ii) a confidence prediction module and a geometry-aware fusion strategy that, coupled, permit a robust fusion of the multiple reference image depth maps from the different source images (iii) we verify the robustness of our method by evaluating on all the most popular MVS benchmarks, namely ETH3D [22], Tanks and Temples [11] and DTU [1] and achieve competitive results.

2. Related Work

In order to estimate dense depth maps, MVS methods compute matching costs for a set of depth candidates at each pixel. Initially, a common approach for candidate genera-

tion was the discretization of a depth range of interest, referred to as plane sweep. The evaluation of the cost function at each candidate depth of each pixel leads to a 3D cost volume that can be processed to extract the final depth map [7]. Later, the PatchMatch algorithm [3] was adopted to generate candidate depth values [21, 13, 6, 30]. The main drawback of these MVS methods is their relying on hand engineered features and matching costs. Instead, the recent deep MVS methods leverage more robust learned features.

Early deep MVS methods also used depth-plane sweeping cost volumes [37, 9], as these can be regularized by 3D convolutional neural networks. However, these methods are characterized by severe memory limitations. Our proposed DELS-MVS leverages learned features, but it avoids constructing global cost volumes by employing a dynamic iterative approach guided by a neural network. This differentiates DELS-MVS from those deep MVS methods relying on coarse-to-fine cost volumes [8, 36, 4, 20, 35] or cost volume aggregation schemes [17, 16, 5, 40] to limit the high memory and computational requirements associated to cost volume processing. In particular, DELS-MVS working directly on the epipolar line, without any cost volume, sets it apart from EPP-MVSNet [17], a cost volume-based method that aggregates additional samples on the epipolar line to improve the quality of lower resolution cost volumes.

Since DELS-MVS operates directly on the epipolar line in pixel space, it does not require a min./max. depth range, as the algorithm explores the epipolar line dynamically. This differentiates DELS-MVS from EPP-MVSNet [17] and previously proposed works leveraging iterative schemes [27, 26, 24, 28, 29] or cost volumes [8, 36, 4, 20, 35], which generate hypothesis in (inverse) depth space and require a min./max. range as input.

Although an iterative depth estimation approach is adopted also in two-view stereo and SLAM methods based on the RAFT-concept [15, 25], these methods require the construction of a full cost volume that prevents them from working at high resolutions. Instead, DELS-MVS can operate at full resolution, as it avoids full cost volumes. This also sets it apart from those MVS methods that update a pre-defined cost volume using a recurrent network [26].

Finally, operating on the epipolar line allows us to use additional information when fusing source views, which differentiates DELS-MVS from previous works [39, 27]. In particular, DELS-MVS proposes a geometry aware fusion strategy, which identifies and utilizes the source image for which the most accurate depth estimate is possible. Moreover, our iterative scheme incorporates a mechanism to detect occluded regions via outer partitions.

3. Algorithm

The proposed system takes as input one reference image $\mathcal{R}$ and $N \geq 1$ source images $\mathcal{S}^{0 \leq n \leq N-1}$ along with
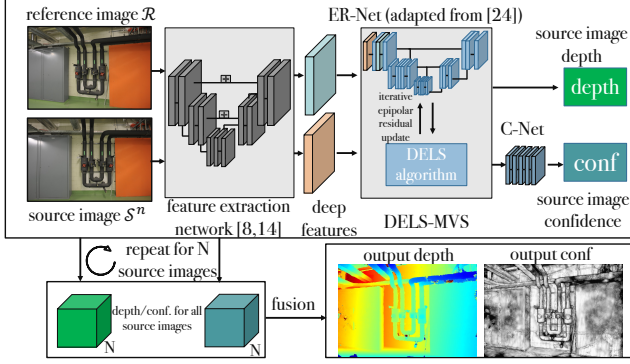
Figure 2. Overview visualization showing the individual steps performed in DELS-MVS. After extracting deep features, the deep epipolar line search yields output depth maps and confidence maps for each source image, later fused into a single output depth map.

the respective reference to source camera transformations $T_{\mathcal{R} \to \mathcal{S}^n}$. First, deep features are extracted from the reference and a given source image $\mathcal{S}^n$ by a convolutional neural network. These features are then fed to our core algorithm, whose purpose is the estimation of the depth map $\mathcal{D}^n$ at the reference image. For each reference image pixel, the goal of our algorithm is to estimate the residual between the actual pixel projections into the source image and our initial guess along the epipolar line. This approach is introduced in Section 3.1. In order to avoid scale dependencies, our algorithm estimates the residuals via iterative classification steps, which are carried out in a coarse-to-fine fashion. We name our algorithm Deep Epipolar Line Search (DELS), as the iterative classification resembles a search and leverages a deep neural network, referred to as Epipolar Residual Network (ER-Net). We describe the DELS algorithm, which represents the core of our DELS-MVS, as well as ER-Net, in Sections 3.2 and 3.3.

Finally, DELS-MVS also features a Confidence Network (C-Net) that associates a confidence map to the estimated depth map $\mathcal{D}^n$. This network is presented in Section 3.4 together with the procedure adopted to fuse all the $\mathcal{D}^{0 \le n \le N-1}$ estimated depth maps at the reference image, utilizing each source, into a single depth map. We provide an overview visualization of DELS-MVS in Figure 2.

3.1. Depth estimation via epipolar residual

The DELS algorithm is initialiazed with a constant depth map $\mathcal{D}_0^n$, where the constant is computed as the average between the minimum and maximum depth range obtained from a Structure from Motion (SfM) system.

Given a pixel (x, y) in the reference image, let $p_0^n(x, y)$ be the projection of (x, y) into the source image $\mathcal{S}^n$ according to the initial depth $\mathcal{D}_0^n(x, y)$. Similarly, indicated with $\mathcal{D}_{\text{GT}}^n$ the sought Ground Truth depth map, let $p_{\text{GT}}^n(x, y)$ be the projection of (x, y) into the source image $\mathcal{S}^n$ according to $\mathcal{D}_{\text{GT}}^n(x, y)$. Both the projections lie on the same epipolar
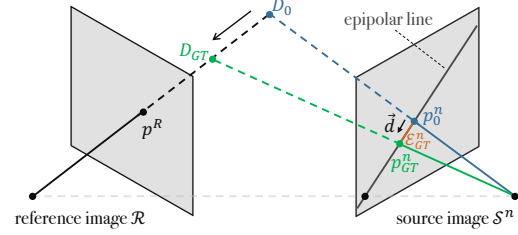

Figure 3. Visualization depicting depth estimation via epipolar residual prediction. Predicting the epipolar residual $\mathcal{E}_{\text{GT}}^n(x, y)$ permits to recover the ground truth depth.

line, as depicted in Figure 3. Our target is the estimation of the one dimensional epipolar residual $\mathcal{E}_{\text{GT}}^n \in \mathbb{R}$ such that the following relation holds true:

$$p_{\text{GT}}^n(x, y) = p_0^n(x, y) + \overrightarrow{d}(x, y)\mathcal{E}_{\text{GT}}^n(x, y) \qquad (1)$$

where the unitary vector $\overrightarrow{d}(x, y) \in \mathbb{R}^2$ defines the epipolar line direction. In fact, it is important to observe that an epipolar line does not have any direction a priori. In particular, we orient $\overrightarrow{d}(x, y)$ such that moving consistently in its direction yields the depth hypothesis of (x, y) to decrease monotonically. For the sake of completness we observe that the following relation holds true:

$$|\mathcal{E}_{\text{GT}}^n(x, y)| = \|p_{\text{GT}}^n(x, y) - p_0^n(x, y)\|_2. \qquad (2)$$

Indicated with $\mathcal{E}^n(x, y)$ the epipolar residual estimated by DELS, Eq. (1) is used to correct the original estimate $p_0^n(x, y)$, which yields $p^n(x, y)$. This can now be converted into a depth value $\mathcal{D}^n(x, y)$ for the original pixel (x, y) using the relative camera transformation $T_{\mathcal{R} \to \mathcal{S}^n}$ from the reference to the source image, which is given. This process is depicted in Figure 3.

3.2. Deep epipolar line search (DELS)

In the MVS scenario, the baselines between the different source images and the reference one can vary a lot, both within the same scene and not. Moreover, depth maps can exhibit very different ranges depending on the specific scene: from very small ones, for the reconstruction of small objects, to very large ones, for the reconstruction of outdoor scenes. In most 3D reconstruction scenarios, the scene scale is not known a priori. Overall, this makes the training of a network, regressing the epipolar error directly, a very challenging task. For this purpose, we propose to re-cast the epipolar line residual estimation problem into an iterative and coarse-to-fine classification scheme.

We denote the epipolar residual estimate after iteration i as $\mathcal{E}_i^n(x, y)$ with $i = 1, 2, \ldots, I$. In order to estimate the epipolar residual at a new iteration i, we segment the epipolar line into into k partitions $L = \{0, 1 \ldots, k - 1\}$ with k even, as shown in Figure 4. In particular we define two sets of partitions: the inner set $L_{\mathcal{I}} = \{1, \ldots, k - 2\}$ and

the outer set $L_{\mathcal{O}} = \{0, k-1\}$. The inner set is centered at the previous iteration epipolar position $p_{i-1}^n(x,y)$ resulting from $\mathcal{E}_{i-1}^n(x,y)$ via Eq. (1). This set extends from a left border $\mathcal{O}_i^L(x,y) = -\frac{k-2}{2}\delta_i(x,y)$ to a right border $\mathcal{O}_i^R(x,y) = \frac{k-2}{2}\delta_i(x,y)$ where $\delta_i(x,y) \in \mathbb{R}^+$ denotes the size of each partition. The partitions 0 and $k-1$ of the outer set model the case where a match lies in a segment of the epipolar line not covered by the current inner partitions, i.e., outside the segment enclosed by $\mathcal{O}_i^L(x,y)$ and $\mathcal{O}_i^R(x,y)$. Therefore, they provide us with an estimation on the direction to move towards in the next iteration.

We leverage a neural network to predict the partition $l_i^\star(x,y) \in L$ hosting the ground truth projection $p_{GT}^n(x,y)$. Once the segment has been identified, a relative residual $e_i^n(x,y) \in \mathbb{R}$ can be computed and added to the previous iteration residual $\mathcal{E}_{i-1}^n(x,y)$ to obtain the current iteration:

$$\mathcal{E}_i^n(x,y) = \mathcal{E}_{i-1}^n(x,y) + e_i^n(x,y), \qquad (3)$$

with $\mathcal{E}_0^n(x,y) = 0$. In particular, we consider two scenarios:

- Case 1: $l_i^\star(x,y) \in L_{\mathcal{I}}$ is selected, which predicts $p_{GT}^n(x,y)$ as located somewhere inside the finite partition $l_i^\star(x,y)$, hence $p_i^n(x,y)$ is placed at its mid point.

- Case 2: the partition $l_i^\star(x,y) \in L_{\mathcal{O}}$ is selected, which provides only a general hint regarding the search direction for $p_{GT}^n(x,y)$, hence $p_i^n(x,y)$ is placed at $\mathcal{O}_i^L(x,y) - \delta_i(x,y)/2$ and $\mathcal{O}_i^R(x,y) + \delta_i(x,y)/2$ for $l_i^\star(x,y) = 0$ and $k-1$, respectively. In particular, this strategy permits to compensate for incorrect predictions at the previous iterations, as some overlap is maintained between the inner partitions of the current and next iteration.

The two scenarios are summarized in Figure 5, where three DELS iterations are sketched. Overall, for a predicted partition $l_i^\star(x,y) \in L$, the outlined update strategy is equivalent to define the relative residual as follows:

$$e_i^n(x,y) = \mathcal{O}_i^L(x,y) + (l_i^\star(x,y) - 1)\delta_i(x,y) + \frac{\delta_i(x,y)}{2}. \qquad (4)$$

DELS coarse-to-fine scheme updates the partition width δ_i at each iteration, differentiating again between the Case 1 and 2. In fact, in the first scenario, the partition width is halved at the beginning of the iteration $i+1$, while in the second scenario it is left untouched:

$$\delta_{i+1}(x,y) = \begin{cases} \max(\frac{1}{2}\delta_i(x,y), \epsilon) & l_i^\star(x,y) \in L_{\mathcal{I}} \\ \delta_i(x,y) & l_i^\star(x,y) \in L_{\mathcal{O}} \end{cases}, \qquad (5)$$

where $\epsilon \in \mathbb{R}^+$ is a lower bound on the partition width. Finally, DELS adopts a multi-resolution strategy with three levels $j = 0, 1, 2$ representing full, half and quarter-resolution, respectively. The number of iterations at level
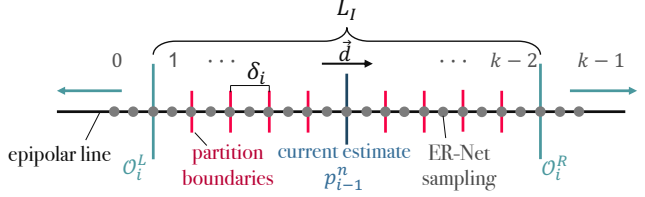

Figure 4. Visualization of the proposed partitioning of the epipolar line for classification, including sample positions of ER-Net.

j is denoted I_j. The level j is initialized with the up-scaled epipolar residual from the previous scale $j+1$, that is $\mathcal{E}_j^n = NN_\uparrow(\mathcal{E}_{j+1}^n) \cdot 2$ where $NN_\uparrow$ is the nearest neighbour upsampling operator. Further, when the partition width lower bound for a given level ϵ_j is reached and the selected partition is contained in the inner set, the iterations are interrupted for a given pixel, as the maximum achievable precision for the epipolar residual is considered achieved. This behaviour is exemplified in Figure 5 (iteration 3, left side).

3.3. Epipolar Residual Network

The classification task at each DELS iteration is carried out by a deep neural network denoted Epipolar Residual Network (ER-Net). The network is fed with the feature maps $F^\mathcal{R}$ and $F^{\mathcal{S}^n}$ extracted from the reference and source images $\mathcal{R}$ and $\mathcal{S}^n$, respectively, using the architecture proposed in [8, 14]. Additionally, ER-Net receives the previous iteration epipolar residual map $\mathcal{E}_{i-1}^n$: this permits, for each reference image pixel (x,y), to sample the features $F^{\mathcal{S}^n}$ located on the epipolar line around the latest estimate $p_{i-1}^n(x,y)$. To this purpose, we incorporate deformable convolutions [41] inside a U-Net-like architecture adapted from [24]. In particular, at the top level of the U-Net, the deformable kernel is spread along the epipolar line and aligned such that it overlaps with samples equally spaced at distance $\delta_i(x,y)/2$ from each other, before and after $p_{i-1}^n(x,y)$. In this setup, the samples in the inner set fall at the beginning, mid and end points of each partition. The sampling scheme is represented by the gray dots in Figure 4. The kernel size is chosen to cover all the samples in the inner set and extend further to the outer set. In particular we set $k = 12$ and the kernel size to five, yielding $5 \times 5 = 25$ samples. For the U-Net lower resolution levels, we keep the same partition widths and kernel size as at the highest level, as this increases the network receptive field.

The multi-channel output layer of the network output is fed to a softmax activation layer, which yields a probability for each one of the k partitions $l_i(x,y) \in L$:

$$P(l_i, x, y) = \text{ER-Net}(F^\mathcal{R}(x,y), F^{\mathcal{S}^n}(x,y), \mathcal{E}_{i-1}^n(x,y)). \qquad (6)$$

During inference, the partition with the maximum softmax probability is chosen:

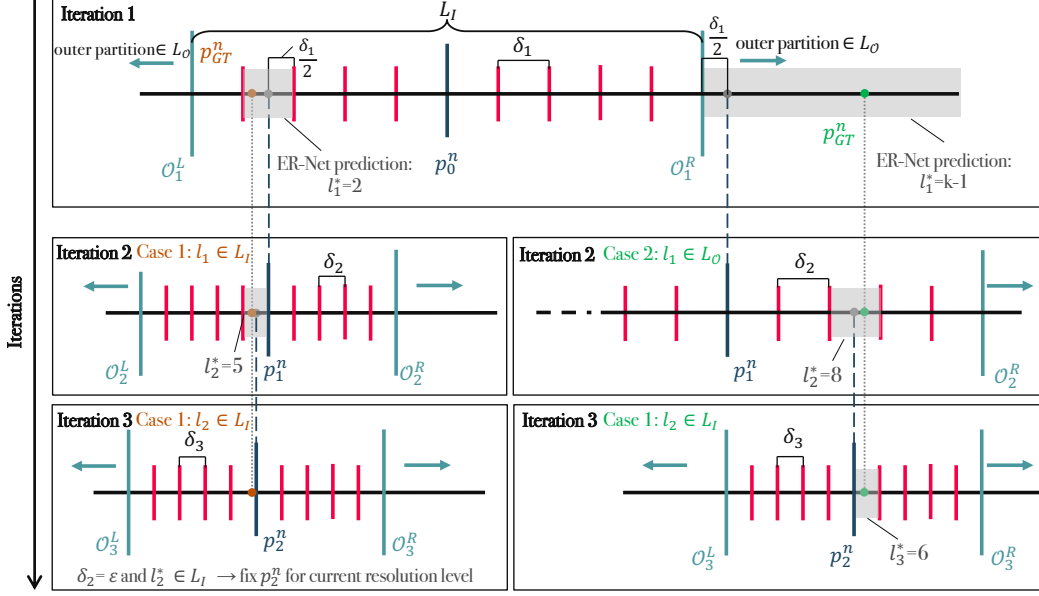$$l_i^\star(x,y) = \arg\max_{l_i} P(l_i, x, y). \qquad (7)$$

Figure 5. DELS algorithm visualization. In general, the algorithm differentiates between two cases: whether the predicted match on the epipolar line lies within the inner set $L_{\mathcal{I}}$ (case 1) or in the outer set $L_{\mathcal{O}}$ (case 2). This alters the partition width δ_i and position of the next estimate for the following iterations as described in Section 3.2.

As anticipated in Sec. 3.2, we adopt ER-Net within a multi-resolution scheme involving full, half and quarter resolution, but we do not share parameters between the levels.

3.4. Confidence Network

Our method computes N depth maps at the reference image, each one computed using a different source image. This poses the problem of how to fuse them into a single depth map taking advantage of all the estimated ones, as some reference image areas may be visible in one source image while not in another. For this purpose, we introduce a Confidence Network (C-Net), which is used to assign a confidence map C^n to each estimated depth map D^n: the confidence maps are then used to guide the fusion of the multiple available depth maps.

At each level j of our multi-resolution scheme, we compute a map resembling the pixel-wise entropy of the partition probabilities, but taking into account its evolution over the DELS iterations:

$$\tilde{C}_j^n(x,y) = \frac{1}{I_j} \sum_{i=1}^{I_j} \sum_{l_i \in L} -P(l_i,x,y)\log(P(l_i,x,y)). \quad (8)$$

We fed C-Net with $\tilde{C}^n = NN_\uparrow(\tilde{C}_2^n) + NN_\uparrow(\tilde{C}_1^n) + \tilde{C}_0^n$, which takes into account the different resolution levels. Moreover, we set $\tilde{C}(x,y)$ to the maximum entropy value in locations where the estimated partition in the last iteration has the property $l_I^\star(x,y) \in L_{\mathcal{O}}$. This incorporates the knowledge that predictions in the last iteration associated to the outer set $L_{\mathcal{O}}$ yield unreliable depth estimates, indicating that an occlusion is present. C-Net architecture consists of

4 convolution layers (kernel size 3, leaky ReLU activations) and a sigmoid activated output yielding C^n in range $[0,1]$.

3.5. Geometry-aware multi-view fusion

At each pixel (x,y), a set of N depth candidates is available, one for each source image. The depth candidate $D^n(x,y)$ is associated to the center of the final epipolar line partition predicted by the DELS algorithm, with the partition length corresponding to a depth range of size $\Delta D^n(x,y) \in \mathbb{R}$ in the scene. It is crucial to observe that, despite the epipolar line partitions sharing the same length across the different source images, the corresponding depth ranges $\Delta D^n(x,y)$ vary from source image to source image, depending on the geometry relating the source cameras to the reference one. In our method, we leverage this key fact and propose a geometry-aware fusion scheme utilizing learned confidences. In particular, depth candidates whose associated partition has both a sufficient confidence and a narrow depth range are preferred, as they correspond to reliable and fine grained depth predictions. Thus, our fusion process proceeds as follows. First, we determine the set N_ω of candidates with a confidence larger than threshold ω. Then, we select the candidate with the smallest depth partition size, that is $\mathcal{D}^{n^\star}(x,y)$ where $n^\star = \arg\min_n \Delta D^n(x,y)$ with $n \in N_\omega$ (for N_ω empty, selecting the one with highest confidence). The final depth is the average of all candidates within a relative distance threshold η from $\mathcal{D}^{n^\star}(x,y)$. For this purpose, we introduce the mask:

$$M^n(x,y) = \begin{cases} 1 & \frac{|\mathcal{D}^{n^\star}(x,y)-\mathcal{D}^n(x,y)|}{\mathcal{D}^{n^\star}(x,y)} < \eta \\ 0 & \text{else.} \end{cases} \quad (9)$$

The final output depth value can be expressed as follows:

$$\mathcal{D}(x,y) = \frac{1}{\sum_{n=0}^{N-1} M^n(x,y)} \sum_{n=0}^{N-1} M^n(x,y)\mathcal{D}^n(x,y). \tag{10}$$

4. Experimental evaluation and Training

In the following sections we introduce our training procedure and present DELS-MVS evaluation results on multiple benchmarks. Furthermore, we motivate contributions and parameters of DELS-MVS with experiments.

4.1. Training procedure

The proposed method is implemented using Py-Torch [19] and trained utilizing the ADAM optimizer [10] with batch size 1.

4.1.1 ER-Net

For our experiments on DTU [1], the network is trained on the pre-processed DTU training set of [37] with learning rate 10^{-4}. When evaluating on Tanks and Temples [11] and ETH3D [22], we additionally fine tune the network on the Blended MVS dataset [38] with learning rate 10^{-5}.

At each training iteration, we pick a random source image and initialize the DELS algorithm with a constant random depth map, similarly to [27] and [2, 24], respectively. A training step involves I_j iterations of the DELS algorithm, depending on the resolution level $j = 0, 1, 2$. The ground truth line partition at iteration i of DELS is computed from the ground truth relative epipolar residual $e_{i_{GT}}^n(x,y) = \mathcal{E}_{GT}^n(x,y) - \mathcal{E}_{i-1}^n(x,y)$ as follows:

$$(l_i)_{GT}(x,y) = \sum_{g=0}^{k-2}[e_{i_{GT}}^n(x,y) > (\mathcal{O}_i^L(x,y) + g\delta_i(x,y))], \tag{11}$$

with [...] being the Iverson bracket, which yields 1 if the condition is true and 0 otherwise. We apply a Cross Entropy (CE) loss between the softmax probabilities $P(l_i, x, y)_{l_i \in \mathcal{L}}$ and the one hot encoding of the ground truth partition $(l_i)_{GT}(x,y)$. The total loss is the sum of the cross entropy losses over the multiple iterations:

$$\text{Loss}_j^{ER}(x,y) = \sum_{i=1}^{I_j} \text{CE}(P(l_i,x,y),(l_i)_{GT}(x,y)). \tag{12}$$

During training no gradient flows to the previous iteration. Moreover, the training includes the different resolution levels progressively. Specifically, we start by training the coarsest level $j = 2$ alone for 4 epochs, then we train $j = 2$ and $j = 1$ for 3 epochs jointly, finally we train all

the three levels $j = 2, j = 1, j = 0$ together for 10 additional epochs. When fine tuning on the Blended MVS [38] dataset, this last step is extended with 5 additional epochs.

4.1.2 C-Net

In order to train our confidence network C-Net, we define a binary ground truth confidence map C_{GT}^n where 1 and 0 indicate valid and non valid depth measurements, respectively. We define the ground truth confidence at $\mathcal{S}^n$ as:

$$C_{GT}^n(x,y) = \begin{cases} 1 & \|p_{GT}^n(x,y) - p^n(x,y)\|_2 < \gamma \\ 0 & else \end{cases} \tag{13}$$

where $p_{GT}^n(x,y)$ and $p^n(x,y)$ are the epipolar line projections of the reference image pixel (x,y) according to its ground truth depth and the depth estimated by DELS, respectively. The confidence network is then trained with Binary Cross Entropy (BCE) loss between the predicted confidence C^n and its ground truth defined above:

$$\text{Loss}^C = BCE(C^n(x,y), C_{GT}^n(x,y)). \tag{14}$$

C-Net is trained separately, after ER-Net training has been completed. It is trained for 2 epochs on the DTU [1] dataset, adopting a learning rate equal to 10^{-5} and $\gamma = 0.6$. C-Net training also adopts source images picked at random.

4.2. Benchmark datasets

We evaluate the proposed DELS-MVS on the DTU [1], ETH3D [22] and Tanks and Temples [11] benchmarks. These evaluate the accuracy and completeness of the point clouds reconstructed by an MVS algorithm. For DTU [1], these two metrics are expressed as average distances in mm. Instead, ETH3D [22] and Tanks and Temples [11] express these two metrics as percentages, with Tanks and Temples [11] using the terms precision and recall for accuracy and completeness, respectively. In order to generate the DELS-MVS point clouds, we adopt the dynamic fusion approach of [34]. Moreover, we leverage our learned confidence to filter out outliers during the fusion. Specifically, in each depth map $\mathcal{D}$, we filter out those pixels with a confidence $C(x,y)$ below a threshold β, where C is computed simply by replacing D^n with C^n in Eq.(10). We set $\beta = 0.05$ for DTU, $\beta = 0.1$ for Tanks and Temples and $\beta = 0.2$ for ETH3D. The DELS-related parameters δ_1 and $\epsilon_0, \epsilon_1, \epsilon_2$ of Sec. 3.2 are set to 4 and $0.5, 1.0, 2$, respectively. The confidence-related parameters η and ω in Sec. 3.4 are set to 0.01 and 0.2. In all our tests we consider $N = 6$ source images. Concerning runtime and memory usage, on DTU [1] a 1600×1184 depth map requires 2.77s and 6.0GB of memory. On ETH3D [22], a 1984×1312 depth map requires 3.63s and 7.5GB. Finally, on Tanks and Temples [11], a 1920×1056 depth map requires 2.87s and 6.2GB.

		Training Set			Testing Set		
	method	F [%]	AC [%]	CO [%]	F [%]	AC [%]	CO [%]
classical	ACMM [30]	78.86	90.67	70.42	80.78	90.65	74.34
	COLMAP [21]	67.66	<u>91.85</u>	55.13	73.01	<u>91.97</u>	62.98
	PCF-MVS [12]	79.42	84.11	75.73	80.38	82.15	79.29
	MAR-MVS [33]	79.21	81.98	77.19	81.84	80.24	84.18
	DeepC-MVS [13]	<u>84.81</u>	90.37	<u>80.30</u>	<u>87.08</u>	89.15	<u>85.52</u>
learned	IB-MVS [24]	71.21	75.21	69.02	75.85	71.64	82.18
	Iter-MVS [26]	71.69	79.79	66.08	80.06	84.73	76.49
	PVSNet [32]	67.48	/	/	72.08	66.41	80.05
	EPP-MVSNet [17]	74.00	82.76	67.58	83.40	85.47	81.79
	PatchMatch-Net [27]	64.21	64.81	65.43	73.12	69.71	77.46
	Vis-MVSNet [39]	72.77	**83.32**	65.53	83.46	**86.86**	80.92
	Ours (DELS-MVS)	**79.15**	79.71	**79.49**	**85.41**	81.44	**90.05**

Table 1. ETH3D High-Res [22] results, showing F-score, accuracy and completeness (percentage measurement using 2cm tolerance, higher is better). Best classical category results are underlined and best learned bold.

		Intermediate Set			Advanced Set		
	method	F [%]	PR [%]	RC [%]	F [%]	PR [%]	RC [%]
classical	ACMM [30]	57.27	49.19	<u>70.85</u>	34.02	35.63	34.90
	COLMAP [21]	42.14	43.16	44.48	27.24	33.65	23.96
	PCF-MVS [12]	55.88	49.82	65.68	<u>35.69</u>	37.52	<u>35.36</u>
	DeepC-MVS [13]	<u>59.79</u>	<u>59.11</u>	61.21	34.54	<u>40.68</u>	31.30
learned	Att-MVS [16]	60.05	**61.89**	58.93	31.93	**40.58**	27.26
	CasMVSnet [8]	56.84	47.62	74.01	31.12	29.68	35.24
	VisMVSnet [39]	60.03	54.44	70.48	/	/	/
	EPP-MVSNet [17]	61.68	53.09	75.58	35.72	40.09	34.63
	IB-MVS [24]	56.02	47.71	72.64	31.96	27.85	41.48
	IterMVS [26]	56.94	47.53	74.69	34.17	28.70	44.19
	PatchMatch-Net [27]	53.15	43.64	69.37	32.31	27.27	41.66
	GBi-Net [18]	61.42	54.48	71.25	37.32	30.58	**48.83**
	Effi-MVS [28]	56.88	47.53	71.58	34.39	32.23	41.90
	NP-CVP-MVSNet[35]	59.64	47.05	**84.69**	/	/	/
	TransMVSNet [5]	63.52	55.14	76.73	37.00	33.84	44.29
	Uni-MVSNet [20]	**64.36**	57.54	73.82	**38.96**	33.76	47.22
	RayMVSNet [29]	59.48	53.21	69.21	/	/	/
	Ours (DELS-MVS)	63.08	55.56	74.03	37.81	34.49	45.71

Table 2. Tanks and Tempels [11] results, showing F-score, precision and recall (percentage measurement, higher is better). Best classical category results underlined and best learned category results are bold.

4.3. ETH3D results

We present results on the high resolution training and test sets in Table 1. The proposed DELS-MVS shows state-of-the-art results, yielding the best results among the learning-based methods on both the training and test sets. It should be noted that DELS-MVS was never trained on the ETH3D training set. Concerning the test set, DELS-MVS yields similar results to the current state-of-the-art classical method DeepC-MVS [13]. In Figure 1, we show point cloud visual comparisons using benchmark visualizations. It can be observed that our method exhibits superior completeness and less noise compared to EPP-MVSNet [17].

4.4. Tanks and Temples results

We report the benchmark results from the intermediate and advanced sets in Table 2. DELS-MVS results are competitive on both subsets, especially on the advanced set, where it achieves results close to Uni-MVSNet [20]. The advanced set contains less constrained camera poses and, compared to intermediate, it consists of more challenging reconstruction scenarios, as suggested by the lower avg. scores of all methods. We provide visual results in Fig. 6.

4.5. DTU results

The evaluation results for DTU [1] are reported in Table 3. In terms of AVG score, which summarizes the overall

	metrics [mm]		
method	AVG	AC	CO
CasMVSnet [8]	0.355	0.325	0.385
BP-MVSNet [23]	0.327	0.333	0.320
VisMVSnet [39]	0.365	0.369	0.361
EPP-MVSNet [17]	0.355	0.413	0.296
IB-MVS [24]	0.321	0.334	0.309
IterMVS [26]	0.363	0.373	0.354
PatchMatch-Net [27]	0.352	0.427	0.277
GBi-Net [18]	**0.289**	**0.315**	**0.262**
Effi-MVS [28]	0.317	0.321	0.313
NP-CVP-MVSNet[35]	0.315	0.356	0.275
TransMVSNet [5]	0.305	0.321	0.289
RayMVSNet [29]	0.330	0.341	0.319
Uni-MVSNet [20]	0.315	0.352	0.278
Ours (DELS-MVS)	0.313	0.342	0.284

Table 3. DTU [1] results, showing completeness, accuracy, average of both (distance measurement, lower is better).

	PU	fusion	β	I	N	metrics [%]		
						F	AC	CO
1	✗	avg.	✗	4,2,2	2	52.80	53.85	53.73
2	✓	avg.	✗	4,2,2	2	61.88	67.45	59.12
3	✓	outer part. mask	✗	4,2,2	2	73.99	74.47	74.09
4	✓	outer part. mask	✗	4,2,2	6	77.09	76.12	78.59
5	✓	non geom. aware	✗	4,2,2	6	78.18	76.06	81.00
6	✓	geom. aware	✗	4,2,2	6	78.76	77.03	**81.16**
7	✓	geom. aware	✗	3,1,1	6	74.14	71.52	77.64
8	✓	geom. aware	✓	4,2,2	6	**79.15**	**79.71**	79.49

Table 4. Architectural experiments on the ETH3D training set [22], where PU denotes whether the dynamic partiton update is applied. The second column describes the technique for fusing the N source depth maps. β denotes whether the output confidence was used to filter point cloud results. I_2, I_1, I_0 are the iterations.

Figure 6. Point cloud results of DELS-MVS on DTU [1] and Tanks and Temples [11].

reconstruction quality, DELS-MVS exhibits improved results compared to recent methods, such as IterMVS [26] and EPP-MVSNet [17]. Moreover, DELS-MVS achieves AVG results which are competitive with other state-of-the-art methods, Uni-MVSNet [20] and Trans-MVSNet [28].

4.6. Architectural experiments

In this section we discuss ablation experiments carried out on the ETH3D [22] high res training set and report them

in Table 4. First, we evaluate the influence of our partition update strategy in Eq. (5). Specifically, row 1 reports the results of a naive version of our algorithm where the partition size is updated exclusively when proceeding to the next level of the multi-resolution scheme. It can be observed that, in row 2, the proposed dynamic update (Sec. 3.2) leads to a significant performance improvement.

Next, we evaluate the effectiveness of the employed outer partitions in modeling unreliable or occluded estimates (Sec. 3.5). In particular, in row 2 we report the results of a fusion variant that, at each pixel, averages the depth candidates obtained from the different source images regardless of the associated partition. In row 3, instead, the fusion average masks out those depth candidates associated to an outer partition. By comparing rows 2 and 3 it can be observed that the masking based on the outer partitions yields a performance increase, thus confirming the validity of the outer partition concept. We evaluate also how the number of available source views N affects DELS-MVS, and observe that the results improve when passing from $N = 2$ to $N = 6$ in rows 3 and 4, respectively.

Regarding our proposed confidence measure, row 5 shows that selecting the depth candidates with the highest confidence $\mathcal{D}^{n^\star}(x, y)$ in the fusion stage yields a further gain. Row 6 confirms that even better results can be obtained when employing our proposed geometry aware fusion, which is made possible by DELS-MVS operating directly on the epipolar line. Row 7 shows that a lower number of iterations leads to worse results, as expected. Finally, in the last row we observe that an additional filtering of the final point clouds with our learned confidence measure leads to the best results.

5. Conclusion

We proposed DELS-MVS, a novel MVS method that leverages a deep neural network to carry out a matching search directly on the source image epipolar line. After estimating a dense depth map at the reference image for each available source, DELS-MVS employs a geometry-aware strategy to fuse them into a single depth map utilizing a learned confidence, designed to promote robustness against outliers. DELS-MVS is iterative, hence it does not require the construction of a large cost volume. Furthermore, no explicit discretization of the depth space within a min./max. range is required, as DELS-MVS explores the epipolar line dynamically. We confirmed the robustness of our approach through evaluations on the ETH3D [22], DTU [1] and Tanks and Temples [11] benchmarks, where DELS-MVS achieves competitive results compared to the state-of-the-art.

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*, pages 1–16, 2016.

[2] Abhishek Badki, Alejandro Troccoli, Kihwan Kim, Jan Kautz, Pradeep Sen, and Orazio Gallo. Bi3d: Stereo depth estimation via binary classifications. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *Proceedings of SIGGRAPH*, 2009.

[4] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[6] 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 (ICCV)*, 2015.

[7] David Gallup, Jan-Michael Frahm, Philippos Mordohai, Qingxiong Yang, and Marc Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[8] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[9] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[11] 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.

[12] Andreas Kuhn, Shan Lin, and Oliver Erdler. Plane completion and filtering for multi-view stereo reconstruction. In *Proceedings of the German Conference on Pattern Recognition (GCPR)*, 2019.

[13] Andreas Kuhn, Christian Sormann, Mattia Rossi, Oliver Erdler, and Friedrich Fraundorfer. Deepc-mvs: Deep confidence prediction for multi-view stereo reconstruction. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2020.

[14] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[15] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching. In *International Conference on 3D Vision (3DV)*, 2021.

[16] Keyang Luo, Tao Guan, Lili Ju, Yuesong Wang, Zhuo Chen, and Yawei Luo. Attention-aware multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[17] 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 International Conference on Computer Vision (ICCV)*, 2021.

[18] Z. Mi, C. Di, and D. Xu. Generalized binary search network for highly-efficient multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems 32*, 2019.

[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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[21] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[22] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[23] Christian Sormann, Patrick Knobelreiter, Andreas Kuhn, Mattia Rossi, Thomas Pock, and Friedrich Fraundorfer. Bp-mvsnet: Belief-propagation-layers for multi-view-stereo. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2020.

[24] Christian Sormann, Mattia Rossi, Andreas Kuhn, and Friedrich Fraundorfer. Ib-mvs: An iterative algorithm for deep multi-view stereo based on binary decisions. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2021.

[25] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 2021.

[26] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, and Marc Pollefeys. IterMVS: Iterative Probability Estimation for Efficient Multi-View Stereo. *arXiv:2112.05126*, 2021.

[27] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[28] Shaoqian Wang, Bo Li, and Yuchao Dai. Efficient multi-view stereo by iterative dynamic cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[29] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[30] Qingshan Xu and Wenbing Tao. Multi-scale geometric consistency guided multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[31] Qingshan Xu and Wenbing Tao. Learning Inverse Depth Regression for Multi-View Stereo with Correlation Cost Volume. In *AAAI Conference on Artificial Intelligence*, 2020.

[32] Qingshan Xu and Wenbing Tao. Pvsnet: Pixelwise visibility-aware multi-view stereo network. *arXiv:2007.07714*, 2020.

[33] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[34] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[35] Jiayu Yang, Jose M. Alvarez, and Miaomiao Liu. Non-parametric depth distribution modelling based depth inference for multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[36] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[37] 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)*, 2018.

[38] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

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

[40] Xudong Zhang, Yutao Hu, Haochen Wang, Xianbin Cao, and Baochang Zhang. Long-range attention network for multi-view stereo. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[41] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.