Lightweight and Accurate Multi-View Stereo with Confidence-Aware Diffusion Model

Fangjinhua Wang*, Qingshan Xu*, Yew-Soon Ong†, *Fellow, IEEE*, Marc Pollefeys†, *Fellow, IEEE*

Abstract—To reconstruct the 3D geometry from calibrated images, learning-based multi-view stereo (MVS) methods typically perform multi-view depth estimation and then fuse depth maps into a mesh or point cloud. To improve the computational efficiency, many methods initialize a coarse depth map and then gradually refine it in higher resolutions. Recently, diffusion models achieve great success in generation tasks. Starting from a random noise, diffusion models gradually recover the sample with an iterative denoising process. In this paper, we propose a novel MVS framework, which introduces diffusion models in MVS. Specifically, we formulate depth refinement as a conditional diffusion process. Considering the discriminative characteristic of depth estimation, we design a condition encoder to guide the diffusion process. To improve efficiency, we propose a novel diffusion network combining lightweight 2D U-Net and convolutional GRU. Moreover, we propose a novel confidence-based sampling strategy to adaptively sample depth hypotheses based on the confidence estimated by diffusion model. Based on our novel MVS framework, we propose two novel MVS methods, DiffMVS and CasDiffMVS. DiffMVS achieves competitive performance with state-of-the-art efficiency in run-time and GPU memory. CasDiffMVS achieves state-of-the-art performance on DTU, Tanks & Temples and ETH3D. Code will be available at: https://github.com/cvg/diffmvs.

Index Terms—3D Reconstruction, Multi-View Stereo, Diffusion Model, Deep Learning.

1 Introduction

Multi-view Stereo (MVS) aims to reconstruct the dense 3D geometry for an observed scene from a set of calibrated images. It has wide applications in real-world scenarios, such as robotics, autonomous driving, virtual / mixed reality and "metaverse". It typically performs multi-view depth estimation and then fuses depth maps into a point cloud or mesh [1]–[4]. When estimating depth maps, MVS is essentially an optimal correspondence search problem in a finite continuous depth space with photometric consistency assumption [2]. However, due to the interference from illumination changes, non-Lambertian surfaces and low-textured areas which are common in real-world scenes, it is challenging to accurately estimate the depth.

To address this problem, traditional methods design many hand-crafted patch similarity metrics and use the plane sweep algorithm [11] to determine the optimal depth in discrete search space [12]. However, their discrete solution usually cannot find the optima [13]. PatchMatch MVS methods [2], [4], [14] leverage the idea of nearest neighbor search and random perturbations to progressively search the optima, which improves the precision of depth estimation. However, they still struggle under the aforementioned challenges due to the traditional hand-crafted modeling.

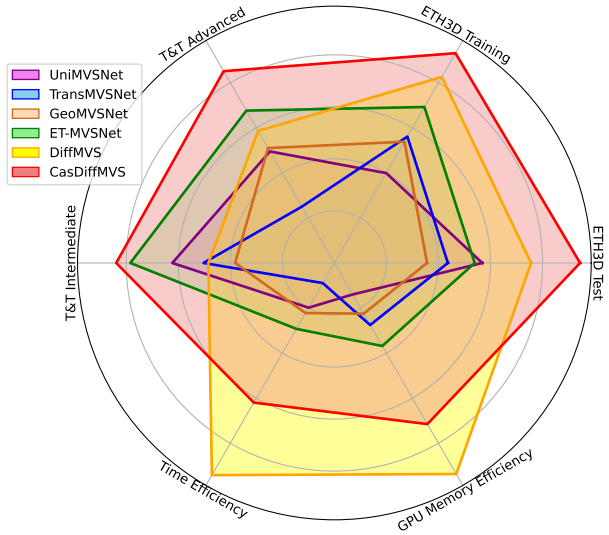


Fig. 1. **Comparison between our methods (DiffMVS and CasDiffMVS) and state-of-the-art learning-based MVS methods on reconstruction performance, efficiency in run-time and GPU memory**. While being more efficient in terms of run-time and GPU memory, CasDiffMVS achieves better reconstruction performance than UniMVSNet [5], TransMVSNet [6], GeoMVSNet [7] and ET-MVSNet [8] on Tanks & Temples (T&T) [9] and ETH3D [10]. Moreover, DiffMVS achieves highest efficiency in run-time and GPU memory, while achieving competitive performance on T&T and ETH3D. Note that for fair comparison, we evaluate the efficiency of all methods with the same input images on one workstation.

Recently, learning-based MVS methods demonstrate a significant improvement in reconstruction quality with deep image features and regularization learned by Convolutional Neural Networks (CNN). Typically, these methods [3], [15]

- *Fangjinhua Wang and Marc Pollefeys are with the Department of Computer Science, ETH Zurich, Switzerland.*
- *Qingshan Xu and Yew-Soon Ong are with the College of Computing and Data Science, Nanyang Technological University, Singapore.*
- *Yew-Soon Ong is also with Center for Frontier AI Research, Institute of High Performance Computing, A*STAR, Singapore.*
- *Marc Pollefeys is additionally with Microsoft, Zurich.*
- *This work is partially supported by a research grant from Microsoft and by the National Research Foundation (NRF), Singapore, through the AI Singapore Programme under the project titled "AI-based Urban Cooling Technology Development" (Award No. AISG3-TC-2024-014-SGKR).*

*: *Equal contribution.* †: *Corresponding authors.*

use the plane sweep algorithm [11] to warp source image features into the reference view and then construct 3D cost volumes, which are regularized by 3D CNN for depth estimation. However, the precision of these methods are still constrained by the limited discrete depth ranges. To alleviate this, recurrent methods [16]–[18] are proposed to increase the number of sampling depth. But these methods significantly sacrifice the efficiency due to their sequential processing for the 3D cost volumes. On the other hand, coarse-to-fine methods [19]–[22] initialize a low-resolution depth map with sparse samples in the depth range. Then they gradually refine it in higher resolutions with a reduced finer depth range. Therefore, these methods further improve depth map accuracy. However, their refinement heavily relies on the quality of the coarse depth map since the estimation in higher resolutions is usually restricted in a reduced depth range that is centered around the coarse depth. If the coarse estimation is incorrect, it is difficult to recover from the errors induced at coarse resolutions and the estimation may be restricted in local minima.

In this work, we rethink how to perform accurate multi-view depth estimation by not only conditioning on the previous coarse estimation, but also by introducing random perturbations. Recently, diffusion models [23], [24] show that injecting random noise can avoid collapsing into local minima [25]. Inspired by the diffusion models that can recover data samples from random noise with iterative denoising process [23], [24], we hope our multi-view depth estimation can mimic the denoising process to introduce random noise and produce accurate estimation. However, since the multi-view depth estimation is a discriminative perception task instead of unconditional generative task, introducing diffusion-based denoising process into multi-view depth estimation will face the following challenges:

- **Diffusion conditions**. In order to obtain accurate deterministic estimation, it is important to condition the diffusion network on some specific guidance. Since matching information is crucial to obtain accurate depth estimation in MVS, it is naturally to introduce matching information as diffusion condition [26]. However, whether MVS requires other diffusion conditions or how to integrate matching information into diffusion models is still under-explored.
- **Diffusion sampling**. For generative tasks, diffusion models [23], [24] typically denoise a noisy sample with the information of this sample only. For discriminative tasks, *e.g.*, feature matching, using the information of a single sample only provides zero-order optimization information, which makes it unable to leverage non-local information for more accurate first-order optimization [4], [27]–[29]. This hinders accurate estimation for discriminative perception tasks with diffusion models.
- **Diffusion efficiency**. Classical diffusion models for generation tasks [23], [24] usually adopt large U-Nets with attention mechanisms [30] to achieve impressive performance. In addition, some recent works [31] have shown that stacking U-Nets is beneficial to improve performance. However, these designs will inevitably hinder the efficiency, which is important for MVS applications.

To overcome the above challenges, we present a novel MVS framework with conditional diffusion models. Specifically, based on the coarse depth initialization, we design three key modules to construct our conditional diffusion models to refine the coarse depth map. First, we propose a condition encoder to effectively adapt diffusion models for depth estimation tasks. The condition encoder fuses matching information, image context and depth context features as the condition feature. This enables our diffusion model to perceive not only local similarity indicated by matching information, but also long-range context information provided by image and depth context features. Second, we introduce a confidence-based sampling strategy to adaptively generate multiple depth hypotheses for each pixel. Different from existing MVS methods that sample depth hypotheses around coarse estimation [19], [22], our sampling is based on the *noisy* coarse estimation perturbed by the random noise from diffusion process. This introduces randomness but may lack informative first-order optimization information if we generate depth hypotheses in a *fixed* range [19], [22] since the coarse estimation may be perturbed a lot by random noise. To solve this, our confidence-based sampling strategy leverages per-pixel predicted confidence to adaptively adjust the sampling range and capture non-local first-order optimization information, thus facilitating the denoising process. Third, we propose a novel lightweight diffusion network, which consists of 2D denoising U-Net and convolutional GRU. Since GRU can capture historical information iteratively to enhance feature expression capability and is proven effective in iterative refinement [27], we design a lightweight U-Net combined with convolutional GRU as our diffusion model. We perform multi-iteration refinement with GRU in a single diffusion timestep, which not only improves performance but also circumvents the usage of large or stacked U-Nets of existing diffusion models [23], [31] and thus improves efficiency.

With our framework, we propose two novel MVS methods, named as DiffMVS and CasDiffMVS. DiffMVS performs depth refinement with a single-stage diffusion model, while CasDiffMVS extends DiffMVS and performs depth refinement on two stages. The former is tailored for real-time applications, while the latter is designed for high-accuracy requirements. Extensive experiments demonstrate that DiffMVS achieves competitive performance with state-of-the-art (SOTA) efficiency in both run-time and memory, while CasDiffMVS achieves SOTA reconstruction performance on various benchmarks with high efficiency, as shown in Fig. 1. In summary, our contributions are as follows:

- We propose a novel MVS framework which exploits conditional diffusion models to achieve efficient, accurate and lightweight multi-view depth estimation.
- We propose a condition encoder to fuse matching information, image context and depth context features as the condition input of diffusion model. This provides the diffusion model specific guidance to generate accurate depth predictions.
- We introduce a confidence-based sampling strategy, which adaptively generates multiple samples in a

local range based on the confidence estimated from diffusion model to provide informative first-order optimization directions.
- We develop a lightweight diffusion network, which leverages convolutional GRU to replace the large denoising U-Nets used in classical diffusion models. This greatly improves the efficiency of our models.
- Based on our novel MVS framework, we propose two novel MVS methods, DiffMVS and CasDiffMVS. The former one achieves competitive performance with SOTA efficiency in both run-time and memory, while the later one achieves SOTA performance on various benchmarks.

2 Related Work

Traditional MVS. Traditional MVS methods can be mainly divided into three categories: voxel based [32]–[35], point cloud based [36], [37] and depth map based [4], [14], [38]–[40]. Depth map based methods decouple the reconstruction problem into multi-view depth estimation and depth fusion, which explicitly improve flexibility and scalability. This characteristic makes depth map based methods dominate MVS. PatchMatch [41] algorithm is commonly used by traditional depth map based methods to improve efficiency. For example, Gipuma [14] uses a red-black checkerboard pattern to propagate depth hypotheses. ACMM [4] further adopts adaptive checkerboard sampling and multi-scale geometric consistency guidance to improve performance. To further alleviate matching ambiguity in low-textured areas, HPM-MVS [29] introduces hierarchical prior mining for non-local MVS. ADP-MVS [42] designs adaptive patch deformation to measure matching cost. These PatchMatch MVS methods leverage random neighbor search and perturbations to estimate depth from a finite continuous depth space. However, traditional MVS methods rely on hand-crafted matching metrics and thus encounter challenges in challenging conditions, *e.g.*, illumination changes, low-textured areas, and non-Lambertian surfaces [9], [10], [43], [44]. Our framework leverages deep features to measure matching similarities and introduces randomness from diffusion models to avoid local minima.

Learning-based MVS. Due to the limited performance of hand-crafted modeling under challenging conditions, many learning-based MVS methods have been proposed in recent years and achieved better performance on various benchmarks [9], [10], [43]. Based on plane sweep algorithm [45], MVSNet [3] proposes differentiable homography to construct a 3D cost volume with warped image features, regularizes the cost volume with a 3D U-Net [46], and regresses the depth map from the probability volume. However, 3D CNN explicitly increases the memory consumption and run-time. To improve computational efficiency, many variants based on MVSNet [3] are proposed and can be mainly categorized into recurrent methods [16], [17], [47] and coarse-to-fine methods [5], [6], [19]–[22], [48]–[52]. Recurrent methods reduce the memory consumption by sequentially regularizing 2D cost maps from the 3D cost volume with recurrent neural networks (RNN). However, these methods have low time efficiency since they trade time for memory. Coarse-to-fine methods first estimate a coarse depth map in low resolution and then reduce the search range in higher resolutions to improve the accuracy. These methods explicitly improve the inference speed and reduce memory consumption. Recently, inspired by RAFT [27] that uses a GRU [53] to emulate first-order optimization, several methods [28], [54] use GRU for depth refinement and achieve high efficiency in memory and run-time. Different from these methods, our methods leverage the denoising process of diffusion models to refine the coarse depth estimation.

Geometry estimation with diffusion models. As a class of deep generative models, diffusion models [23], [25], [55] start from a random noise and recover the data sample with an iterative denoising process. They have achieved impressive results in image [23], [24], [56] and video [57] generation tasks. Recently, many researchers have shown that diffusion models can be used for many geometry estimation tasks, such as monocular depth estimation [58]–[60], monocular normal estimation [61], [62], depth completion [63], feature matching [64], pose estimation [65]–[67] and optical flow estimation [68], [69]. With the introduction of conditional diffusion models, these methods outperform previous state-of-the-art methods. In this paper, we introduce a novel conditional diffusion model in MVS and explore a generative paradigm for accurate and efficient reconstruction.

Confidence estimation in stereo/MVS. Confidence estimation has been shown to be effective in predicting the reliability of disparity/depth estimation in stereo/MVS. Early methods [70]–[72] utilize conventional features to train random forest classifiers for confidence estimation with a two-class label for each pixel. In stereo matching, recent methods [73]–[75] leverage CNN to predict confidence from disparity map, reference image and cost volume. In MVS, some methods also use confidence/uncertainty estimation to improve depth prediction. UCSNet [20] computes the variance of the probability volume to define the confidence and uses it to determine the depth range at the next stage. UGNet [76] employs uncertainty learning to predict the uncertainty from the cost volume directly to capture the uncertainty of the depth map. Vis-MVSNet [49] estimates the uncertainty of the pair-wise depth map to construct a more reliable aggregated cost volume to predict the final depth map. Our methods predict confidence based on our designed diffusion model, which incorporates cost volume, image context and depth context features. Moreover, our predicted confidence is combined with the noisy depth for depth sampling, which greatly enhances the power of our diffusion models.

3 Methodology

In this section, we introduce the details of our MVS framework with conditional diffusion models. The framework mainly consists of two modules: depth initialization and diffusion-based depth refinement. First, we obtain an initial coarse depth map with the depth initialization module. Then we develop a novel conditional diffusion model to refine it in higher resolutions. Based on our MVS framework, we propose DiffMVS and CasDiffMVS, which are visualized in Fig. 2. Structually, DiffMVS performs depth refinement on a single stage (stage 2) and then upsamples to

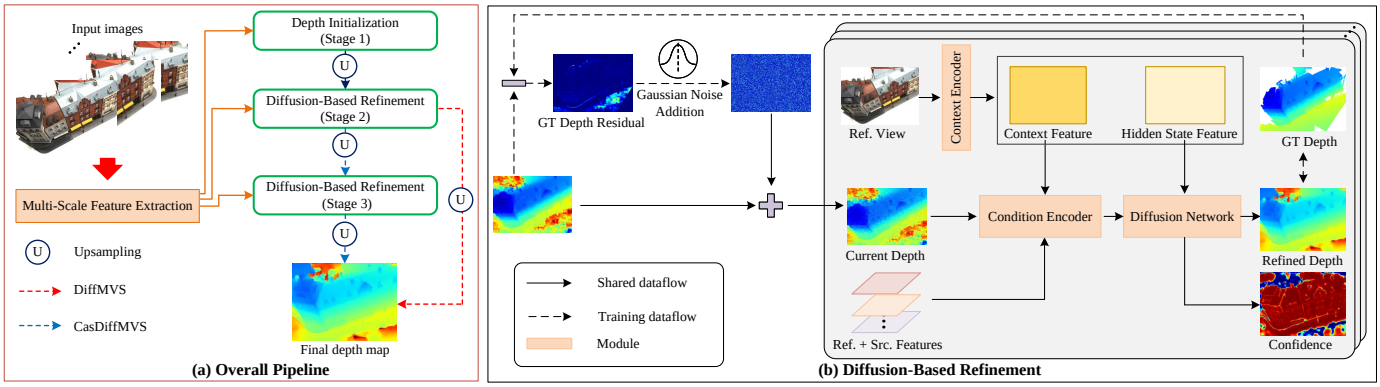


Fig. 2. **Overview of DiffMVS and CasDiffMVS.** (a) shows the overall pipeline of DiffMVS and CasDiffMVS. A coarse depth map is initialized in low resolution (stage 1). Then DiffMVS uses a single diffusion-based refinement module on stage 2 to refine the coarse depth map, while CasDiffMVS uses cascade diffusion-based refinement on stage 2, 3. Finally, the refined depth map is upsampled to full resolution. (b) shows the architecture of diffusion-based refinement. It refines the coarse depth map by denoising the current *noisy* depth map. A context encoder is used to extract image context and hidden state features from the reference view. To encode condition feature for the diffusion network, a condition encoder takes as input the image context feature, depth context feature and the cost volume constructed with reference and source features. With the condition feature and hidden state feature, the diffusion network outputs refined depth and confidence. Best viewed on a screen when zoomed in.

full resolution, while CasDiffMVS performs refinement on two stages (stage 2, 3) and then upsamples to full resolution.

3.1 Preliminaries

As a class of generative models, diffusion models [23], [25], [55] iteratively recover samples from random noise. Given the data distribution $\boldsymbol{x} \sim p(\boldsymbol{x})$, diffusion model defines a Markovian chain of *forward* process and gradually adds Gaussian noise $\mathcal{N}(0, \boldsymbol{I})$ to the sample $\boldsymbol{x}_0$. The noisy sample $\boldsymbol{x}_t$ at timestep $t \in \{1, \dots, T\}$ can be computed as:

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}), \tag{1}$$

where $\bar{\alpha}_t := \prod_{i=1}^{t} \alpha_i = \prod_{i=1}^{t}(1-\beta_i)$ and β_i represents the noise variance schedule [23].

In the *reverse* process, the denoising model $f_\theta(\boldsymbol{x}_t, t)$, parameterized with learnable parameters θ, is trained to predict the added noise $\hat{\boldsymbol{\epsilon}}$ from $\boldsymbol{x}_t$ by minimizing the training objective function $\mathcal{L}$ as follows [23]:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon} \sim \mathcal{N}(0,\boldsymbol{I}), t \sim \mathcal{U}(1,T)} ||\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}||^2, \tag{2}$$

where $\mathcal{U}(1, T)$ denotes the uniform distribution. To better control the generation, conditional diffusion models introduce condition c, *e.g.*, text prompt [24], image [59], [77], into the denoising model as $f_\theta(\boldsymbol{x}_t, c, t)$. Note that the diffusion models for discriminative tasks [31], [59], [64], [78], [79] are all conditional diffusion models.

At the inference stage, $\boldsymbol{x}_0$ is reconstructed from a Gaussian noise $\boldsymbol{x}_T$ in an iterative way based on the diffusion network $f_\theta(\cdot)$ and an update rule [23], [80]. To achieve good performance, DDPM [23] needs to sample many steps and thus has low sampling efficiency. In contrast, DDIM [80] proposes non-Markovian forward process, which is capable to generate high-quality samples with much fewer sampling steps.

3.2 Problem Formulation and Motivation

In this work, we aim to solve multi-view depth estimation tasks in MVS with a conditional diffusion model. Recently, most state-of-the-art MVS methods [6], [19], [22], [54] initialize a coarse depth in low resolution and then refine it in high resolutions, which achieves impressive performance in both efficiency and accuracy. Following these methods, we initialize a low-resolution depth map and then progressively upsample and refine it with diffusion process in a cascade pipeline.

Formally, for a reference image $\boldsymbol{I}_0$ and its neighboring images $\{\boldsymbol{I}_i\}_{i=1}^{N-1}$, we denote the image features extracted from these images as $\boldsymbol{F}_0$ and $\{\boldsymbol{F}_i\}_{i=1}^{N-1}$. The objective of multi-view depth estimation is to find the depth map D corresponding to the reference image $\boldsymbol{I}_0$. In our framework, we aim to refine an initial depth map, $\boldsymbol{D}_{\text{init}}$, with a diffusion model and image features. Specifically, we seek to find $\boldsymbol{D}^*$ that maximizes the posterior probability of the estimated depth map given the initial depth map $\boldsymbol{D}_{\text{init}}$ and image features $\boldsymbol{\mathcal{F}} = \{\boldsymbol{F}_0, \{\boldsymbol{F}_i\}_{i=1}^{N-1}\}$, *i.e.*, $p(\boldsymbol{D}|\boldsymbol{\mathcal{F}}, \boldsymbol{D}_{\text{init}})$. To find the depth map $\boldsymbol{D}^*$ that maximizes the posterior, we can use the maximum a posteriori (MAP) approach:

$$\begin{aligned}\boldsymbol{D}^* &= \arg\max_{\boldsymbol{D}} p(\boldsymbol{D}|\boldsymbol{\mathcal{F}}, \boldsymbol{D}_{\text{init}}) = \arg\max_{\boldsymbol{D}} p(\boldsymbol{\mathcal{F}}|\boldsymbol{D}) \cdot p(\boldsymbol{D}|\boldsymbol{D}_{\text{init}}) \\ &= \arg\max_{\boldsymbol{D}} \{\log p(\boldsymbol{\mathcal{F}}|\boldsymbol{D}) + \log p(\boldsymbol{D}|\boldsymbol{D}_{\text{init}})\}.\end{aligned} \tag{3}$$

In this probabilistic interpretation, the first term (data term) represents the matching evidence between image features $\boldsymbol{F}_0$ and $\{\boldsymbol{F}_i\}_{i=1}^{N-1}$, and the second term (prior term) encodes prior knowledge of the depth map $\boldsymbol{D}$.

Previous methods focus on constructing cost volumes / matching confidence and regressing the final depth map, including TransMVSNet [6], MVSTER [52] and DUSt3R [81]. This might be analogy to $\arg\max_{\boldsymbol{D}} \log p(\boldsymbol{\mathcal{F}}|\boldsymbol{D})$. On the one hand, these methods usually construct the data term using a refined depth range centered around the initial depth map, $\boldsymbol{D}_{\text{init}}$. This often causes these methods to get stuck in local minima. On the other hand, the prior term $\log p(\boldsymbol{D}|\boldsymbol{D}_{\text{init}})$ is not explicitly considered in these regression-based methods. In fact, it is difficult for the data term to learn an accurate depth estimation in ambiguous areas, *e.g.* textureless areas and occlusions, which are common in the real world.

Since conditional diffusion models excel at learning the data distribution $p(\boldsymbol{D}|\boldsymbol{\mathcal{F}}, \boldsymbol{D}_{\text{init}})$ given inputs $\boldsymbol{\mathcal{F}}$ and $\boldsymbol{D}_{\text{init}}$, we can leverage them to explicitly incorporate both data term and prior term within a unified probabilistic framework. The data term encodes the geometry consistency between $\boldsymbol{F}_0$ and $\{\boldsymbol{F}_i\}_{i=1}^{N-1}$, which guides the generation process to be consistent with conditions. Meanwhile, the prior term encodes the structural and geometric constraints, which alleviates the ambiguity in challenging regions. Although diffusion models in image generation tasks enhance diversity without conditions, our model reduces diversity as it is conditioned on the geometry consistency between $\boldsymbol{F}_0$ and $\{\boldsymbol{F}_i\}_{i=1}^{N-1}$, which is a characteristic of conditional diffusion models [82]. The random noise within our diffusion model aims to avoid local minima. Therefore, inspired by the ability to avoid falling into the local minima and the explicit prior modeling in diffusion models, we aim to explore diffusion models for MVS.

3.3 Multi-scale feature extraction

Given a reference image $\boldsymbol{I}_0 \in \mathbb{R}^{H\times W\times 3}$ and its neighboring source images $\{\boldsymbol{I}_i\}_{i=1}^{N-1} \in \mathbb{R}^{H\times W\times 3}$, we extract multi-scale features with a Feature Pyramid Network (FPN) [83], where N is the number of input images, H and W denote the image height and width, respectively. Specifically, we extract image features at M different stages (DiffMVS: $M = 2$, CasDiffMVS: $M = 3$). We denote the image feature of $\boldsymbol{I}_i$ at stage m with $\boldsymbol{F}_i^m \in \mathbb{R}^{H_m\times W_m\times C_m}$, where $H_m = H/2^{4-m}, W_m = W/2^{4-m}$, $m = 1, \cdots, M$. For simplicity, we omit the superscript m below.

For the reference image $\boldsymbol{I}_0$, we further use a context FPN encoder to extract the multi-scale context features $\boldsymbol{F}_c$ and hidden state features $\boldsymbol{h}_0$ for upsampling and diffusion model.

3.4 Depth initialization

We initialize the coarse depth map $\boldsymbol{D}_{\text{init}}$ at stage 1 (1/8 resolution) and then refine it in higher resolutions with diffusion model. The initialization pipeline is shown in Fig. 3.

Cost volume construction. Given a pre-defined depth range $[d_{\min}, d_{\max}]$, we uniformly sample per-pixel depth hypotheses $\{d_j\}_{j=1}^{D_0}$ in the *inverse* range $[1/d_{\max}, 1/d_{\min}]$, where D_0 is the number of initial samples. This inverse sampling is considered more suitable for large-scale scenes [15], [16], [22], [51]. Then we warp the features of source views into the reference view at these depth hypotheses. Specifically, for a pixel $\boldsymbol{p}$ in the reference view, we compute the corresponding pixel $\boldsymbol{p}_{i,j} := \boldsymbol{p}_i(d_j)$ in source view i as follows:

$$\boldsymbol{p}_{i,j} = \boldsymbol{K}_i \cdot (\boldsymbol{R}_{0,i} \cdot (\boldsymbol{K}_0^{-1} \cdot \boldsymbol{p} \cdot d_j) + \boldsymbol{t}_{0,i}), \tag{4}$$

where $\boldsymbol{K}_0, \boldsymbol{K}_i$ denote intrinsic and $[\boldsymbol{R}_{0,i}|\boldsymbol{t}_{0,i}]$ denotes relative transformation between reference view and source view i. Then we obtain the warped source feature $\boldsymbol{F}_i(\boldsymbol{p}_{i,j})$ with bilinear interpolation.

To reduce computation, we follow [15], [22], [28] and use group-wise correlation to compute the similarity between reference feature $\boldsymbol{F}_0(\boldsymbol{p})$ and warped source feature $\boldsymbol{F}_i(\boldsymbol{p}_{i,j})$.

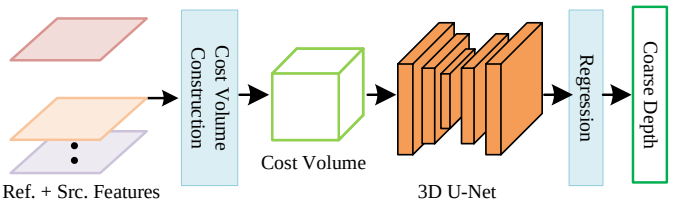


Fig. 3. **Pipeline of depth initialization.** A lightweight cost volume is constructed and regularized by a 3D U-Net to produce the initial depth $\boldsymbol{D}_{\text{init}}$. See Sec. 3.4 for more details. Best viewed on a screen when zoomed in.

By evenly dividing C_1 feature channels into $G = 4$ groups, we compute the g-th group similarity $\boldsymbol{s}_i(\boldsymbol{p}, j)^g$ as:

$$\boldsymbol{s}_i(\boldsymbol{p}, j)^g = \frac{1}{C_1/G} \langle \boldsymbol{F}_0(\boldsymbol{p})^g, \boldsymbol{F}_i(\boldsymbol{p}_{i,j})^g \rangle, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product. This results in $N-1$ similarity volumes $\{\boldsymbol{S}_i\}_{i=1}^{N-1} \in \mathbb{R}^{H_1\times W_1\times D_0\times G}$.

To aggregate these $N-1$ similarity volumes, the naïve way is to compute the average of them [15]. However, this does not take occlusions into consideration, which will lead to invalid matching and inaccurate estimations [2]. Following [22], [48], [51], we estimate the pixel-wise view weight for each source view i with a lightweight network to provide visibility information and robustly aggregate matching information. Specifically, for each $\boldsymbol{S}_i$, we apply two 3D convolution layers and then *softmax* along the depth dimension to produce $\boldsymbol{w}_i \in \mathbb{R}^{H_1\times W_1\times D_0}$. For pixel $\boldsymbol{p}$ and source view i, the corresponding view weight $\boldsymbol{W}_i(\boldsymbol{p})$ is computed as:

$$\boldsymbol{W}_i(\boldsymbol{p}) = \max_j \boldsymbol{w}_i(\boldsymbol{p}, j). \tag{6}$$

Finally, the cost volume $\boldsymbol{V}$ is aggregated as:

$$\boldsymbol{V} = \frac{\sum_{i=1}^{N-1} \boldsymbol{W}_i \cdot \boldsymbol{S}_i}{\sum_{i=1}^{N-1} \boldsymbol{W}_i}. \tag{7}$$

Depth prediction. We regularize $\boldsymbol{V}$ with a lightweight 3D U-Net and then apply *softmax* along the depth dimension to produce the probability volume $\boldsymbol{P} \in \mathbb{R}^{H_1\times W_1\times D_0}$. The initial depth $\boldsymbol{D}_{\text{init}}$ is computed as the expectation in *inverse* range:

$$\boldsymbol{D}_{\text{init}} = \left(\sum_{j=1}^{D_0} \boldsymbol{P}(j) \cdot \frac{1}{d_j} \right)^{-1}, \tag{8}$$

where $\boldsymbol{P}(j)$ is the probability of all pixels at depth d_j.

3.5 Diffusion-based refinement

The initial depth $\boldsymbol{D}_{\text{init}}$ is not accurate since it is estimated at low resolution with sparse depth hypotheses. We propose a conditional diffusion model, shown in Fig. 2(b), to refine the coarse depth on higher resolutions. Unlike unconditional diffusion models in generation tasks [23], depth estimation is a discriminative perception task and thus our diffusion model is conditioned on the feature encoded by *condition encoder*, so that the generation diversity can be constrained and thus depth map can be accurately denoised.

Forward process. To make our method robustly generalize to scenes with different scales, we use the normalized *inverse*

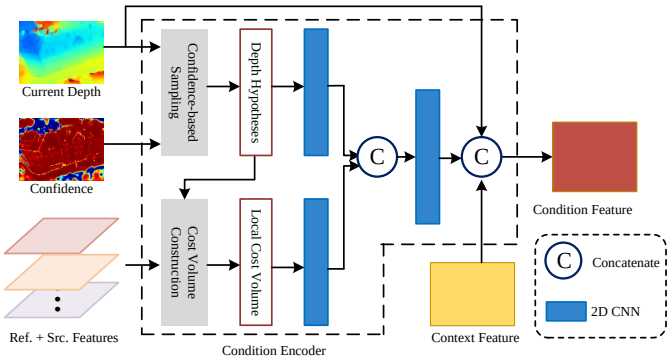


Fig. 4. **Structure of our condition encoder.** With the new depth hypotheses generated with confidence-based sampling, we compute the local cost volume as introduced in Sec. 3.4. Our condition encoder applies 2D convolution layers to encode geometric matching information from the local cost volume, depth hypotheses and image context feature as the condition feature. Best viewed on a screen when zoomed in.

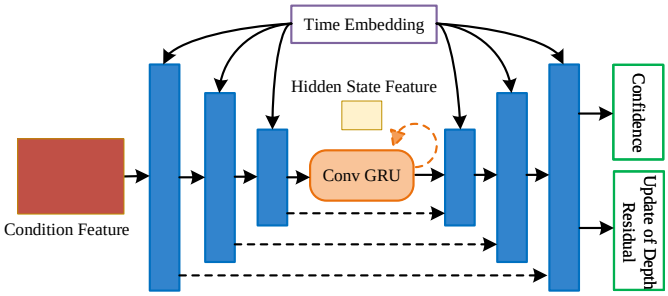


Fig. 5. **Structure of our diffusion network.** The network combines a 2D U-Net with convolutional GRU. The condition feature is used as the condition for 2D U-Net to denoise the depth residual. Best viewed on a screen when zoomed in.

depth $\bar{\boldsymbol{D}} \in [0, 1]$ for depth map $\boldsymbol{D}$ throughout the diffusion process:

$$\bar{\boldsymbol{D}} = \left(\frac{1}{\boldsymbol{D}} - \frac{1}{d_{\max}}\right) / \left(\frac{1}{d_{\min}} - \frac{1}{d_{\max}}\right). \tag{9}$$

For each refinement stage, we denote the initial depth upsampled from previous stage as $\boldsymbol{D}_0$. We compute the ground truth depth residual $\boldsymbol{x}_0$ as:

$$\boldsymbol{x}_0 = \bar{\boldsymbol{D}}_{\text{gt}} - \bar{\boldsymbol{D}}_0, \tag{10}$$

where $\bar{\boldsymbol{D}}_{\text{gt}}$ denotes the normalized *inverse* of ground truth. During training, we uniformly sample timestep $t \sim \boldsymbol{U}(1, T)$ and compute the noisy $\boldsymbol{x}_t$ with Eq. 1.

Reverse process. To refine the depth, our diffusion model denoises the depth residual $\boldsymbol{x}_0$ with the reverse process. Following prior works [23], [24], we design our conditional denoising diffusion model based on a 2D U-Net architecture, shown in Fig. 5.

Recently, RAFT [27] is popular in optical flow estimation since it performs iterative refinements with GRU to achieve impressive performance with low complexity. Motivated by RAFT, many stereo [84] / MVS [28], [54], [85] methods adopt iterative GRU refinements with lightweight 2D convolutions to outperform a single refinement with cumbersome 3D convolutions. Since we focus on both efficiency and performance in this work, we follow these methods and introduce convolutional GRU into a lightweight 2D U-Net (See below for details). Consequently, in each diffusion timestep, we iteratively refine K times. Note that recent diffusion models, *e.g.*, DiffusionDet [31] and DifFlow3D [79], also update multiple iterations in each diffusion timestep to improve performance in discriminative perception tasks.

Specifically, in the k-th iteration of diffusion timestep t, our diffusion model predicts the update of depth residual, $\Delta\boldsymbol{x}_{t,k}$, and the confidence $\boldsymbol{C}_{t,k}$ of current estimation. After K iterations, the depth residual is denoised as:

$$\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_t + \Delta\boldsymbol{x}_t = \boldsymbol{x}_t + \sum_{k=1}^{K} \Delta\boldsymbol{x}_{t,k}, \tag{11}$$

where $\hat{\boldsymbol{x}}_0$ is the denoised prediction of the ground truth residual $\boldsymbol{x}_0$. Based on Eq. 10, the final refined *inverse* depth can be represented as $\bar{\boldsymbol{D}}_{t,K} = \bar{\boldsymbol{D}}_0 + \hat{\boldsymbol{x}}_0$. In the k-th iteration, we compute the intermediate refined *inverse* depth $\bar{\boldsymbol{D}}_{t,k}$ for training process:

$$\bar{\boldsymbol{D}}_{t,k} = \bar{\boldsymbol{D}}_0 + \boldsymbol{x}_t + \sum_{n=1}^{k} \Delta\boldsymbol{x}_{t,n}. \tag{12}$$

Condition encoder. For simplicity, we omit the subscript t of diffusion timestep below. In the k-th iteration, we first sample D_1 per-pixel new hypotheses in a local range around previous depth estimation $\bar{\boldsymbol{D}}_{k-1}$, resulting in $\bar{\boldsymbol{D}}_k^{\text{sample}} \in \mathbb{R}^{H_m \times W_m \times D_1}$ (details of sampling strategy with confidence will be introduced later). Second, we compute the local cost volume $\boldsymbol{L}_k \in \mathbb{R}^{H_m \times W_m \times D_1 \times G}$ for the new hypotheses (Sec. 3.4), which is further reshaped into $\mathbb{R}^{H_m \times W_m \times (D_1 \times G)}$. For the view weights $\{\boldsymbol{W}_i\}_{i=1}^{N-1}$ in Eq. 7, we reuse those predicted at stage 1 and upsample them with nearest neighbors.

For diffusion model, the depth samples as well as corresponding costs can guide the model to find a refined depth value. In addition, the consistency between depth map and image is also helpful to refine the depth map [22]. Therefore, we propose a lightweight *condition encoder* to inject useful conditions into the diffusion model. The structure of our condition encoder is depicted in Fig 4. Specifically, the cost volume $\boldsymbol{L}_k$ is processed by 2D convolutional layers. Additionally, we apply 2D convolutional layers on $\bar{\boldsymbol{D}}_k^{\text{sample}}$ to generate depth context features. Then we concatenate these two features and apply 2D convolutional layers, which is finally concatenated with the previous depth $\bar{\boldsymbol{D}}_{k-1}$ and reference context feature $\boldsymbol{F}_c$ as the condition input of diffusion model.

Denoising U-Net with GRU. Following prior works [23], [24], we design our conditional diffusion model based on a 2D U-Net architecture. Following DDPM [23], we inject the timestep embedding into the layers of U-Net. We experimentally find that the attention operation used in DDPM does not explicitly improve the performance. Therefore, we do not introduce attention into the U-Net so that the models are efficient and lightweight.

As mentioned before, we introduce GRU into the 2D U-Net in consideration of efficiency and performance. The structure of our diffusion network is depicted in Fig. 5. Specifically, we introduce a convolutional GRU in the lowest resolution of U-Net. To initialize the hidden state of GRU,

we apply 2D convolutional layers followed by *tanh* nonlinearity on the hidden state feature $\boldsymbol{h}_0$ of the reference image. In the k-th iteration, the U-Net encodes the conditional feature from the *condition encoder* to update the hidden state $\boldsymbol{h}_{k-1}$ of the GRU into $\boldsymbol{h}_k$. Then $\boldsymbol{h}_k$ is decoded to predict the update of depth residual $\Delta \boldsymbol{x}_k$, as well as the confidence $\boldsymbol{C}_k$ of current depth estimation (*sigmoid* is applied on confidence to ensure $\boldsymbol{C}_k \in [0, 1]$).

Confidence-based sampling. Utilizing non-local information with multiple samples is proven effective in many MVS methods to provide first-order optimization information [4], [27]–[29]. Therefore, different from classical diffusion models [23] that use a single sample, we propose confidence-based sampling strategy and adaptively generate multiple samples. To refine the coarse depth map, many MVS methods [6], [19], [22], [28], [54] use a constant search range for all pixels to generate new depth hypotheses around the previous estimation. However, it is reasonable to include confidence/uncertainty of the estimation in the sampling. Specifically, the search range should be reduced for pixels with accurate estimation to further improve accuracy, while enlarged for pixels with erroneous estimation so that the search range may cover the ground truth. For example, UCSNet [20] computes the per-pixel sampling range with the variance of the probability volume on multiple stages. However, this is not suitable in our framework since we do not estimate the probability volume during refinement.

In our methods, we adaptively adjust the per-pixel sampling range $\boldsymbol{R}_k$ based on the confidence $\boldsymbol{C}_{k-1}$ from the diffusion model. For the first iteration $k = 1$, we set $\boldsymbol{R}_1 = \boldsymbol{R}_{\text{init}}$, where $\boldsymbol{R}_{\text{init}}$ is a pre-defined range for each stage. For $k > 1$, $\boldsymbol{R}_k$ is computed as:

$$\boldsymbol{R}_k = (1 - \boldsymbol{C}_{k-1}) \cdot (\boldsymbol{R}_{\max} - \boldsymbol{R}_{\min}) + \boldsymbol{R}_{\min}, \tag{13}$$

where $\boldsymbol{R}_{\min} = \lambda_{\min} \cdot \boldsymbol{R}_{\text{init}}$, $\boldsymbol{R}_{\max} = \lambda_{\max} \cdot \boldsymbol{R}_{\text{init}}$ are pre-defined limits of the sampling range. Then we uniformly sample D_1 depth hypotheses in the inverse range $[\bar{\boldsymbol{D}}_{k-1} - \boldsymbol{R}_k, \bar{\boldsymbol{D}}_{k-1} + \boldsymbol{R}_k]$.

3.6 Learned upsampling

Since our methods use multi-stage structure, we need to upsample the depth between different stages. Instead of simple bilinear or nearest interpolation [19], [22], we use learned upsampling with mask [27], [28], which experimentally improves performance. Specifically, we compute the depth value of each pixel in the upsampled depth as the convex combination of a 3×3 grid of its coarse resolution neighbors. For stage m, we predict a mask, with a shape as $H_m \times W_m \times (r \times r \times 9)$, by applying 2D convolution layers on the reference context feature $\boldsymbol{F}_c^m$, where r is the upsample ratio. Then we apply *softmax* over the weights of the 9 neighbors and predict the upsampled depth as the weighted combination over the neighborhood.

3.7 Training loss

Our loss function $\mathcal{L}_{\text{full}}$ includes the losses for all the depth maps, *i.e.*, initial depth $\boldsymbol{D}_{\text{init}}$, intermediate depth inside the diffusion-based refinement $\{\boldsymbol{D}_{t,k}\}_{k=1}^{K}$ at timestep t, and all the upsampled depth maps. For clarification, we sort all depth maps based on the estimation order as $\{\boldsymbol{D}_j\}_{j=1}^{J}$, where J is the number of depth maps. Following IterMVS [28], we transform the depth map $\boldsymbol{D}$ into the normalized inverse space with Eq. 9 and compute the L_1 loss w.r.t. the ground truth depth map $\boldsymbol{D}_{\text{gt}}$:

$$\mathcal{L} = |\bar{\boldsymbol{D}} - \bar{\boldsymbol{D}}_{\text{gt}}|, \tag{14}$$

where $\bar{\boldsymbol{D}}, \bar{\boldsymbol{D}}_{\text{gt}}$ are the normalized inverse depth for $\boldsymbol{D}, \boldsymbol{D}_{\text{gt}}$. For the depth map $\boldsymbol{D}$ in diffusion-based refinement where we estimate the corresponding confidence $\boldsymbol{C}$, we include $\boldsymbol{C}$ inside the loss function as:

$$\mathcal{L} = \frac{|\bar{\boldsymbol{D}} - \bar{\boldsymbol{D}}_{\text{gt}}|}{1 - \boldsymbol{C}} + \lambda_C \cdot \log(1 - \boldsymbol{C}), \tag{15}$$

where $\lambda_C = 0.05$ is the weight. We use confidence $\boldsymbol{C}$ to adjust the weight for L_1 loss, and $\lambda_C \cdot \log(1 - \boldsymbol{C})$ as a regularization term to avoid trivial solution as $\boldsymbol{C} = 0$.

Following [27], [28], [54], we use exponentially increasing weights for the depth maps to balance the depth supervisions across different stages and iterations. The motivation is that the depth should be estimated from coarse to fine and thus the error is gradually penalized more. Therefore, the final loss $\mathcal{L}_{\text{full}}$ is written as:

$$\mathcal{L}_{\text{full}} = \sum_{j=1}^{J} \beta^{J-j} \mathcal{L}_j, \tag{16}$$

where $\mathcal{L}_j$ is the loss for $\boldsymbol{D}_j$, $\beta = 0.9$ is the weight.

4 Experiments

4.1 Datasets

Following prior works, we use DTU [43] and BlendedMVS [44] for training, and evaluate the performance on DTU, Tanks & Temples [9] and ETH3D [10].

DTU dataset. DTU [43] is an indoor multi-view stereo dataset collected with known accurate camera trajectory. It contains 124 different object-centered scenes captured with 7 different lighting conditions. Following previous methods, we use the training, testing and validation sets introduced in SurfaceNet [86].

BlendedMVS. BlendedMVS [44] is a large-scale synthetic dataset with various scenes. It provides over $17k$ high-quality training samples with accurate depth maps and camera poses. It is divided into the training set and validation set. Compared with DTU [43], BlendedMVS contains scenes with various scales, *e.g.*, from object level to city level, and is thus commonly used to improve the generalization ability of learning-based MVS methods.

Tanks & Temples. Tanks & Temples [9] is a large-scale real-world dataset consisting of both indoor and outdoor scenes, which is divided into training set and test set. Learning-based MVS methods [3], [19], [22] commonly test the zero-shot generalization ability on the test set. The test set is further divided into *intermediate* and *advanced* subsets, where the *advanced* subset is more challenging than the *intermediate* subset because of the complex structures.

ETH3D. ETH3D [10] is a large-scale dataset captured in complex real-world scenes under challenging conditions, *e.g.*, low-textured regions and wide baselines. It contains both outdoor and indoor scenes. The dataset is divided

into training set and test set. Both sets are commonly used to test the zero-shot generalization ability of learning-based MVS methods [28], [50], [52].

4.2 Implementations

In this section, we discuss the implementation details of DiffMVS and CasDiffMVS, including hyper-parameter settings, training and inference details.

Hyper-parameters. For depth initialization, we set the number of initial samples $D_0 = 48$ for both DiffMVS and CasDiffMVS. For confidence-based sampling, we set $D_1 = 6$, $\lambda_{\text{min}} = 0.25$, $\lambda_{\text{max}} = 4$, $\boldsymbol{R}_{\text{init}} = 3/192.0$ for DiffMVS. For CasDiffMVS, we set $\boldsymbol{R}_{\text{init}} = 1/96.0$ for stage 2 and $\boldsymbol{R}_{\text{init}} = 1/192.0$ for stage 3. We set $D_1 = 4$, $\lambda_{\text{min}} = 0.125$, $\lambda_{\text{max}} = 8$ for both stages of CasDiffMVS.

For diffusion models, we set the total diffusion timesteps as $T = 1000$. In each diffusion timestep, we refine $K = 4$ times for DiffMVS and $K = 3$ times for CasDiffMVS. Since we aim to refine the coarse initial depth and *normalize* the depth values during diffusion into the range $[-1, 1]$, we find that using standard Gaussian noise, *i.e.*, $\mathcal{N}(0, \boldsymbol{I})$, like DDPM [23] will introduce too strong noise for refinement. Therefore, we set the Gaussian noise for stage m as $\mathcal{N}(0, \sigma_m^2 \boldsymbol{I})$ to control the noise scale. Specifically, we set $\sigma_2 = 0.5$ for the refinement on stage 2, and $\sigma_3 = 0.1$ for the refinement on stage 3 when training on DTU. Considering that the DTU-trained models provide a good initialization for fine-tuning and the domain gap between the DTU and large-scale BlendedMVS, we further design a *noise-scaling strategy* to fine-tune our models on BlendedMVS. Specifically, σ_2 and σ_3 will be halved at the start of fine-tuning and after 8 epochs. We empirically find this enables our models to generalize better on Tanks & Temples [9] and ETH3D [10].

Training. We implement DiffMVS and CasDiffMVS with PyTorch [87], and use Adam [88] ($\beta_1 = 0.9, \beta_2 = 0.999$) as the optimizer. First, we train our models on DTU training set [43] for evaluation on DTU testing set. For evaluation on Tanks & Temples [9] and ETH3D [10], we further finetune the DTU-pretrained models on BlendedMVS [44]. The image resolution is set to 640×512 for DTU, and 768×576 for BlendedMVS. The number of input views N is set to 5 for DTU and 9 for BlendedMVS. On both datasets, we set the batch size as 4 and train the models under OneCycleLR scheduler with a maximum learning rate as 0.001. We train DiffMVS and CasDiffMVS for 12 and 16 epochs respectively on each dataset.

Inference. During inference, we adopt DDIM [80] to improve sampling efficiency. We set DDIM sampling timestep as $T_s = 1$ since we observe that our methods converge fast with 1 sampling timestep only and the performance will not explicitly improve with more sampling timesteps.

After depth estimation, we use photometric and geometric consistency to filter outliers in the depth maps, following common practices [3], [47], and back-project the valid pixels into the 3D space as the final point cloud.

4.3 Evaluation

In this section, we evaluate both DiffMVS and CasDiffMVS on popular MVS benchmarks [9], [10], [43] and compare with state-of-the-art methods. In addition, since efficiency in run-time and memory consumption is important in practice, we compare the efficiency of our methods with state-of-the-art methods.

TABLE 1
Quantitative results on DTU [43]. Methods are separated into four categories (from top to bottom): traditional methods, learning-based methods without refinement, with single-stage refinement and with multi-stage refinement. Red, orange, and yellow highlights indicate the 1st, 2nd, and 3rd-best performing method.

Methods	Acc.(mm) ↓	Comp.(mm) ↓	Overall(mm) ↓
Gipuma [14]	0.283	0.873	0.578
COLMAP [38]	0.400	0.664	0.532
MVSNet [3]	0.396	0.527	0.462
R-MVSNet [16]	0.383	0.452	0.417
AA-RMVSNet [17]	0.376	0.339	0.357
IterMVS [28]	0.373	0.354	0.363
DiffMVS	0.318	0.297	0.308
CasMVSNet [19]	0.325	0.385	0.355
PatchmatchNet [22]	0.427	0.277	0.352
EPP-MVSNet [50]	0.413	0.296	0.355
PVSNet [51]	0.337	0.315	0.326
UniMVSNet [5]	0.352	0.278	0.315
TransMVSNet [6]	0.321	0.289	0.305
Effi-MVS [54]	0.321	0.313	0.317
MVSTER [52]	0.340	0.266	0.303
GeoMVSNet [7]	0.331	0.259	0.295
ET-MVSNet [8]	0.329	0.253	0.291
EI-MVSNet [89]	0.346	0.260	0.303
Effi-MVS+ [90]	0.327	0.275	0.301
GC-MVSNet [91]	0.330	0.260	0.295
CANet [92]	0.351	0.248	0.299
CasDiffMVS	0.310	0.286	0.298

Evaluation on DTU. Following prior works, we evaluate with our models trained on DTU training set only. We set the image size and number of views N to 1600×1152 and 5 respectively. Table 1 summarizes the quantitative results. Compared with IterMVS [28], the current state-of-the-art method with single-stage refinement, DiffMVS outperforms it with a large margin in all metrics. Moreover, DiffMVS performs better than many learning-based methods with multi-stage refinement, *e.g.*, Effi-MVS [54], UniMVSNet [5], in *overall* quality. As the multi-stage extension of DiffMVS, CasDiffMVS outperforms other learning-based methods in *accuracy* and achieves very competitive performance in *overall* quality when compared with the top-performing methods [7], [8].

Evaluation on Tanks & Temples. To demonstrate the zero-shot generalization capability of our methods, we evaluate on Tanks & Temples. We use the depth range, camera parameters and view selection provided by Patchmatch-Net [22]. The image size and number of views are set to 1920×1056 and 10 respectively. For depth filtering, we use dynamic geometric consistency checking method in-

TABLE 2
Quantitative results on Tanks & Temples [9] using F-score (%) (higher is better). Methods are separated into four categories (from top to bottom): traditional methods, learning-based methods without refinement, with single-stage refinement and with multi-stage refinement. Red, orange, and yellow highlights indicate the 1st, 2nd, and 3rd-best performing method.

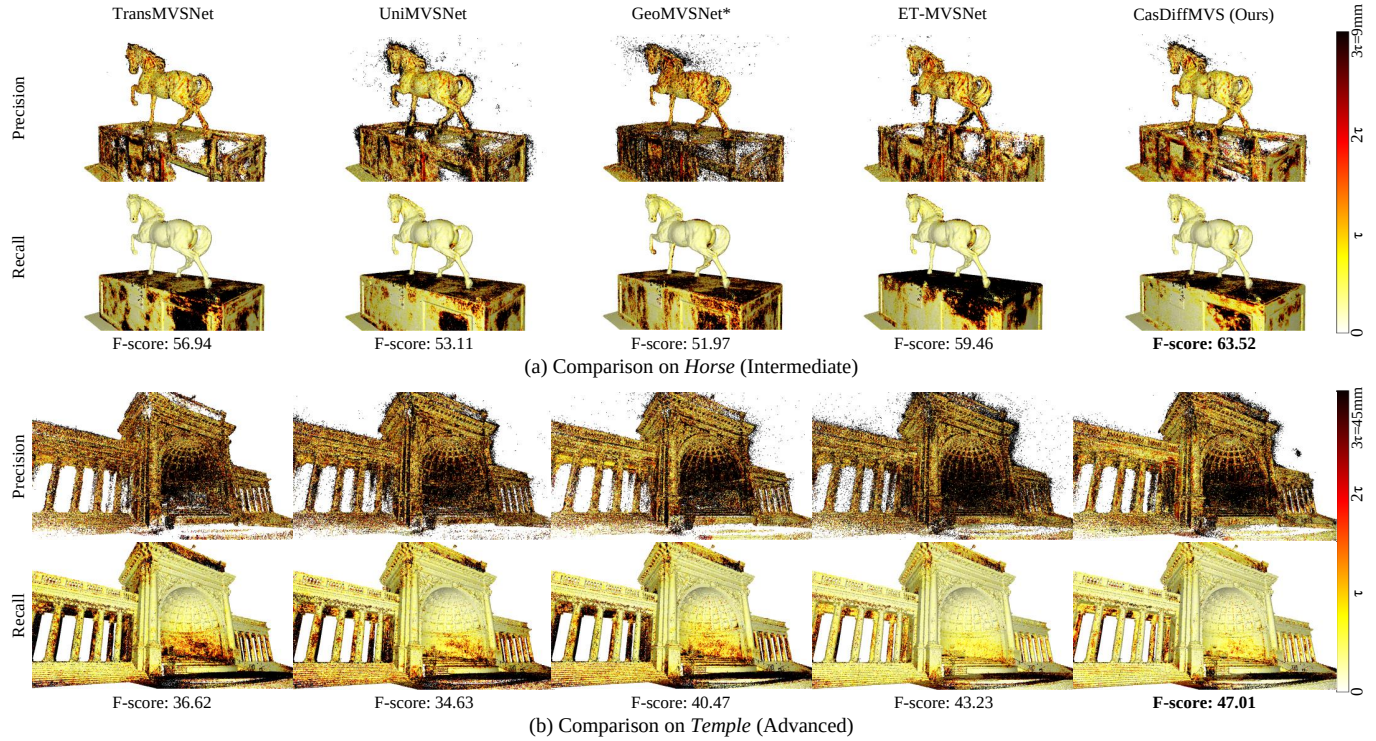
Methods	Intermediate									Advanced						
	Mean	Family	Francis	Horse	Light.	M60	Panther	Play.	Train	Mean	Audi.	Ball.	Court.	Museum	Palace	Temple
COLMAP [2]	42.14	50.41	22.25	26.63	56.43	44.83	46.97	48.53	42.04	27.24	16.02	25.23	34.70	41.51	18.05	27.94
ACMM [4]	57.27	69.24	51.45	46.97	63.20	55.07	57.64	60.08	54.48	34.02	23.41	32.91	41.17	48.13	23.87	34.60
HPM-MVS [29]	61.39	73.40	57.67	56.96	64.70	60.39	61.21	60.35	56.43	40.80	32.85	46.00	40.92	53.04	29.63	42.37
R-MVSNet [16]	48.40	69.96	46.65	32.59	42.95	51.88	48.80	52.00	42.38	24.91	12.55	29.09	25.06	38.68	19.14	24.96
IterMVS [28]	56.94	76.12	55.80	50.53	56.05	57.68	52.62	55.70	50.99	34.17	25.90	38.41	31.16	44.83	29.59	35.15
DiffMVS	63.39	78.14	62.73	62.29	63.20	61.10	61.84	59.65	58.18	39.69	31.10	43.45	37.85	48.74	32.94	44.05
CasMVSNet [19]	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
PatchmatchNet [22]	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
EPP-MVSNet [50]	61.68	77.68	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
PVSNet [51]	59.11	78.13	61.62	52.11	56.90	60.12	53.77	57.58	52.64	35.51	24.40	40.96	34.23	47.95	29.02	36.50
UniMVSNet [5]	64.36	81.20	66.43	53.11	63.46	66.09	64.84	62.23	57.53	38.96	28.33	44.36	39.74	52.89	33.80	34.63
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
Effi-MVS [54]	56.88	72.21	51.05	51.78	58.63	58.71	56.21	57.07	49.38	34.39	20.22	42.39	33.73	45.08	29.81	35.09
MVSTER [52]	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
GeoMVSNet* [7]	62.67	80.12	66.14	51.97	65.95	61.70	60.40	61.20	53.91	39.08	25.30	45.75	37.72	50.36	34.85	40.47
ET-MVSNet [8]	65.49	81.65	68.79	59.46	65.72	64.22	64.03	61.23	58.79	40.41	28.86	45.18	38.66	51.10	35.39	43.23
EI-MVSNet [89]	65.52	81.59	67.67	61.67	63.18	65.10	63.42	60.62	60.95	40.68	29.97	45.86	38.45	49.50	35.78	44.53
Effi-MVS+ [90]	64.07	79.87	66.77	57.29	66.35	62.83	61.11	62.14	56.23	41.20	32.04	47.04	38.84	51.26	34.95	43.06
GC-MVSNet [91]	62.74	80.87	67.13	53.82	61.05	62.60	59.64	58.68	58.48	38.74	25.37	46.50	36.65	49.97	35.81	38.11
CANet [92]	65.05	80.41	63.85	59.62	67.32	65.03	64.18	62.05	57.90	41.22	31.26	46.17	40.54	52.70	33.49	43.15
CasDiffMVS	65.87	81.74	69.21	63.52	65.89	62.92	62.35	61.31	60.00	41.81	32.66	45.70	39.34	50.93	35.25	47.01

*: GeoMVSNet does not provide official checkpoint on BlendedMVS. Following official scripts, we finetune the DTU pre-trained model on BlendedMVS.

troduced in [47]. The quantitative results are summarized in Table 2. For the learning-based methods with single-stage refinement, our DiffMVS is 11.33% and 16.15% better than IterMVS [28] on intermediate and advanced set respectively. Remarkably, it also outperforms some learning-based methods with multi-stage refinement, such as PVSNet [51] and MVSTER [52]. For the learning-based methods with multi-stage refinement, our CasDiffMVS achieves state-of-the-art performance on both sets. In Fig. 6, we visualize the reconstruction errors on 'Horse' and 'Temples' scenes. CasDiffMVS demonstrates significant improvements over existing methods and produces more complete surfaces. Overall, our methods demonstrate very competitive generalization performance.

Evaluation on ETH3D. We further evaluate the generalization ability of our methods on the challenging ETH3D. We set the image size and number of views to 1920×1280 and 10 respectively. The results are summarized in Table 3. Since ETH3D is a large-scale dataset with large baselines, it poses great challenges to the generalization ability of learning-based methods. Traditional PatchMatch MVS methods [29], [40] leverage the fast depth search and random depth perturbations to achieve promising results on this dataset. By introducing the diffusion denoising process, our methods can better consider depth sampling on this dataset. Therefore, our CasDiffMVS achieves the best performance among the learning-based methods on both training and test sets. Moreover, compared with HPM-MVS [29], the state-of-the-art traditional method, CasDiffMVS achieves competitive performance on the test set. In addition, compared with IterMVS [28], our DiffMVS outperforms it on both training set and test set. Note that, DiffMVS also performs better than many state-of-the-art multi-stage methods, *e.g.*, GeoMVSNet [7], ET-MVSNet [8], Effi-MVS+ [90], on both sets. Fig. 7 shows the reconstruction error comparisons for 'Relief' and 'Terrace' scenes. CasDiffMVS achieves more accurate and complete reconstructions than other methods. These results further demonstrate the generalization capabilities of our methods in challenging scenarios.

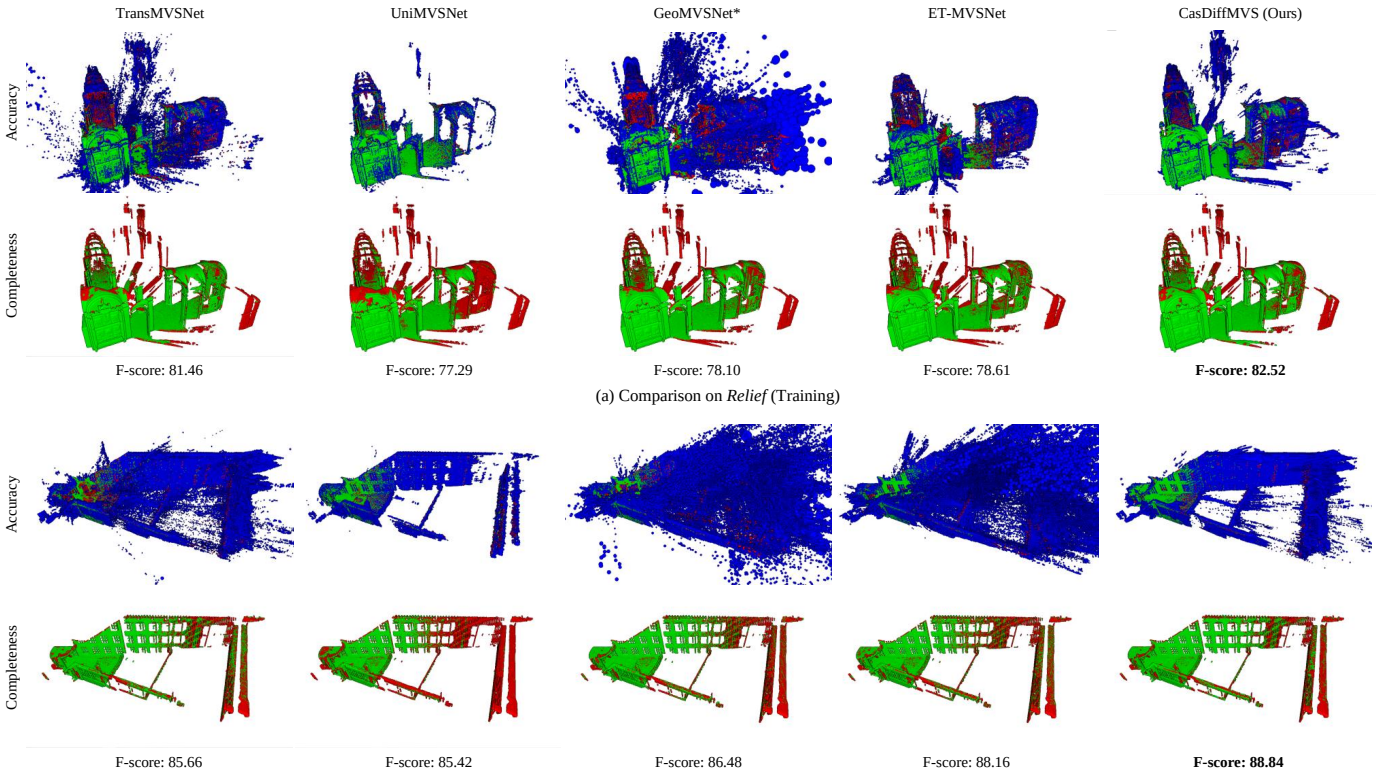
Efficiency comparison. Efficiency in memory and run-time is important in industrial applications, especially for mobile devices with limited computational resources. Therefore, we compare the run-time and GPU memory consumption of our methods with the state-of-the-art MVS methods on a workstation with one NVIDIA 2080 Ti GPU and visualize the results in Fig. 8. Our DiffMVS achieves highest efficiency in both run-time and GPU memory. Compared with IterMVS [28], the current most efficient method, DiffMVS consumes 9.13% less GPU memory and is 69.49% faster. Moreover, on DTU [43], Tanks & Temples [9] and ETH3D [10], DiffMVS mostly outperforms state-of-the-art efficient methods [22], [28], [52], [54] and achieves competitive performance as TransMVSNet [6] and UniMVSNet [5], while being much more efficient. CasDiffMVS has more computational overheads than DiffMVS because of the two-stage diffusion-based refinement. However, CasDiffMVS still achieves similar efficiency as PatchmatchNet [22] in both GPU memory and run-time, while being more efficient than the top-performing methods [5]–[8]. Moreover, CasDiffMVS achieves very competitive performance on three benchmarks when compared with these top-performing methods [5]–[8].



(a) Comparison on *Horse* (Intermediate)

(b) Comparison on *Temple* (Advanced)

Fig. 6. Qualitative comparisons of reconstruction errors on Tanks and Temples [9]. We visualize precision and recall error maps for 'Horse' and 'Temple' scenes.



(a) Comparison on *Relief* (Training)

(b) Comparison on *Terrace* (Training)

Fig. 7. Qualitative comparisons of reconstruction errors on ETH3D [10]. We visualize accuracy and completeness error maps for indoor 'Relief' and outdoor 'Terrace' scenes. Green points are accurate, red points are inaccurate and blue points are unobserved with respect to the ground truth.

TABLE 3
Quantitative results of different methods on ETH3D [10] using F_1-score (at evaluation threshold $2cm$, higher is better). Methods are separated into three categories (from top to bottom): traditional methods, learning-based methods with single-stage refinement and with multi-stage refinement.

Methods	Training			Test		
	Acc. ↑	Comp. ↑	F_1-score ↑	Acc. ↑	Comp. ↑	F_1-score ↑
COLMAP [2]	91.85	55.13	67.66	91.97	62.98	73.01
ACMM [4]	90.67	70.42	78.86	90.65	74.34	80.78
HPM-MVS [29]	90.66	79.50	84.58	92.13	83.25	87.11
IterMVS [28]	79.79	66.08	71.69	84.73	76.49	80.06
DiffMVS	76.74	74.32	74.86	80.40	84.28	82.10
PatchmatchNet [22]	64.81	65.43	64.21	69.71	77.46	73.12
EPP-MVSNet [50]	82.76	67.58	74.00	85.47	81.49	83.40
PVSNet [51]	83.00	71.76	76.57	81.55	83.97	82.62
UniMVSNet [5]	85.39	57.83	67.18	89.12	72.74	79.10
TransMVSNet [6]	69.62	71.47	70.10	73.26	81.84	76.98
MVSTER [52]	68.08	76.92	72.06	77.09	82.47	79.01
GeoMVSNet* [7]	68.71	71.22	69.69	69.77	83.68	75.70
ET-MVSNet [8]	74.71	71.38	72.46	75.28	84.01	78.63
EI-MVSNet [89]	-	-	-	85.12	83.77	84.19
Effi-MVS+ [90]	82.33	70.42	75.28	83.90	83.81	83.62
CasDiffMVS	79.93	75.20	76.76	85.21	85.37	85.11

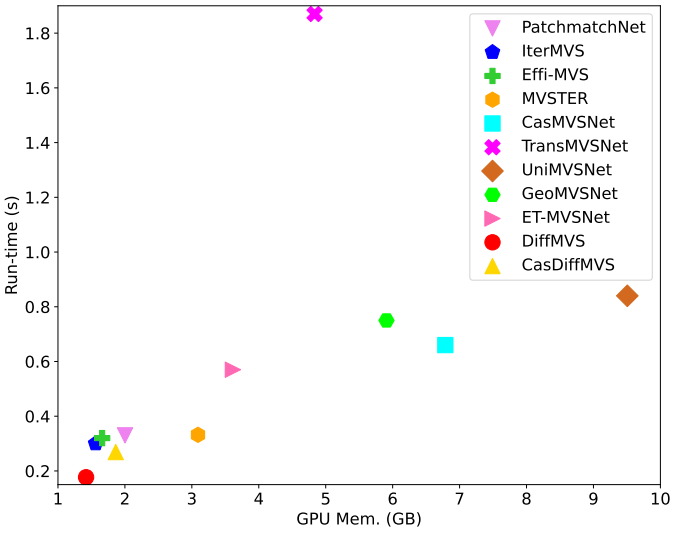


Fig. 8. Efficiency comparison with state-of-the-art MVS methods [5]–[8], [19], [22], [28], [52], [54] on DTU [43] (image size: 1600×1152, number of input views $N = 5$). For fair comparison, all experiments are done on one workstation with a NVIDIA 2080 Ti GPU.

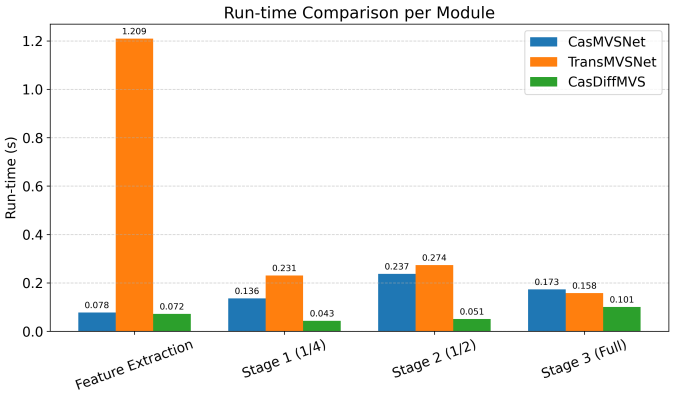


Fig. 9. Run-time comparison per module on DTU [43] (image size: 1600×1152, number of input views $N = 5$). For fair comparison, all experiments are done on one workstation with a NVIDIA 2080 Ti GPU.

To better illustrate how our proposed method CasDiffMVS achieves its runtime advantages over 3D CNN or transformer-based frameworks, we show run-time comparison per module for CasMVSNet, TransMVSNet and CasDiffMVS in Fig. 9. We observe that TransMVSNet is the slowest because of the expensive attention operation introduced in feature extraction. For depth inference in different stages, CasDiffMVS runs much faster than CasMVSNet and TransMVSNet, where both CasMVSNet and TransMVSNet use 3D CNN. Compared to these methods with 3D CNN and attention modules that are computationally expensive, we carefully design the network architecture to reduce computation, *e.g.* no 3D CNN or attention. In addition, since our framework predicts an initial depth map $\boldsymbol{D}_{\text{init}}$ and then uses a diffusion model to refine it, our framework requires fewer sampling timesteps (See more analysis in the next section). As a result, our CasDiffMVS is more efficient than 3D CNN or transformer-based frameworks.

4.4 Ablation study

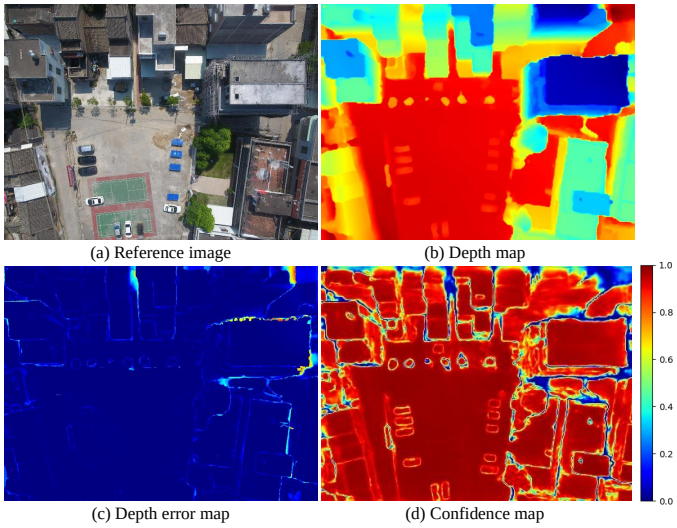
In this section, we conduct an ablation study to validate the effectiveness of different components in our pipeline. If not specified, the experiments are conducted with DiffMVS. For evaluation on DTU [43] testing set, we use the models trained on DTU training set only. For evaluation on ETH3D [10] training set, we use the models finetuned on BlendedMVS [44].

TABLE 4
Ablation study of DiffMVS on DTU [43] and ETH3D [10]. Settings used in our method are underlined.

Experiments	Methods	DTU testing set			ETH3D training set		
		Acc. ↓	Comp. ↓	Overall ↓	Acc. ↑	Comp. ↑	F_1-score ↑
Diffusion Models	(I) w./o. diffusion	0.324	0.312	0.318	72.68	70.21	70.69
	(II) noise in training	0.339	0.320	0.329	73.55	72.48	72.47
	(III) noise in training & testing	0.328	0.304	0.316	73.06	72.08	72.01
	(IV) w. diffusion	**0.318**	**0.297**	**0.308**	**76.74**	**74.32**	**74.86**
Diffusion Conditions	(V) w./o. cost volume	3.094	2.269	2.682	48.95	47.29	47.16
	(VI) w./o. depth context	0.333	0.298	0.316	44.94	43.54	43.07
	(VII) w./o. image context	0.320	0.302	0.311	72.46	72.89	72.08
	(VIII) w. all conditions	**0.318**	**0.297**	**0.308**	**76.74**	**74.32**	**74.86**
Diffusion Sampling	(IX) single sample	0.355	0.324	0.340	65.96	69.15	66.77
	(X) w./o. confidence	0.337	0.301	0.319	72.82	71.00	71.19
	(XI) confidence for regularization	0.335	0.301	0.318	75.44	72.20	73.22
	(XII) w. confidence-based sampling	**0.318**	**0.297**	**0.308**	**76.74**	**74.32**	**74.86**
Diffusion Efficiency	(XIII) single U-Net	2.760	1.874	2.317	50.99	50.99	50.20
	(XIV) stacked U-Nets	0.321	0.300	0.310	71.99	73.00	71.93
	(XV) GRU	**0.318**	**0.297**	**0.308**	**76.74**	**74.32**	**74.86**

Diffusion models. In this ablation, we remove the diffusion process from DiffMVS as the ablation model, named as *DiffMVS0*. Specifically, we remove the timesteps and Gaussian noise throughout training and testing. However, we keep the condition encoder, 2D U-Net, convolutional GRU and confidence-based sampling for fair comparison. That is, *DiffMVS0* is in fact a vanilla coarse-to-fine or iterative refinement approach. As shown in Row I of Table 4, *DiffMVS0* performs worse on both DTU and ETH3D, with performance drops of 3.2% and 5.6%, respectively. This effectively demonstrates that the performance gains of DiffMVS stem from our designed diffusion mechanism. Since the diffusion process introduces random noise into the model, we experiment with introducing noise augmentation in *DiffMVS0*. During training, we add random Gaussian noise on depth map and keep noise scale the same as DiffMVS for fair comparison. During testing, we try two settings: adding noise during testing (Row III) or not (Row II). We find that the zero-shot generalization ability on ETH3D improves when introducing random noise during training. However, the performance on both DTU and ETH3D are still worse than DiffMVS. Therefore, we conclude that the diffusion process is effective and improves the robustness of reconstruction.

Diffusion conditions. In this ablation, we investigate the efficacy of different diffusion conditions in our proposed condition encoder, including cost volume, depth context and image context features. As shown in Row V of Table 4, the reconstruction quality degrades a lot on both DTU and ETH3D without the cost volume as condition. This demonstrates that the cost volume plays a crucial role in our diffusion encoder as it encodes geometric matching information. By removing the depth context, the results in Row VI of Table 4 indicate that the depth context is beneficial to improve the generalization capability of our method on ETH3D. This is because the depth context features provide relative depth position information, allowing our condition encoder to consider corresponding depth ranges for new scenes. Furthermore, by removing the image context, we observe performance degradation from Row VII of Table 4. In fact, the image context features provide semantic information of objects, reflecting depth continuities of objects to some extent. Therefore, the image context features can also facilitate our condition encoder.



(a) Reference image (b) Depth map
(c) Depth error map (d) Confidence map

Fig. 10. Visualization of reference image, depth map, depth error map and confidence map on the validation set of BlendedMVS [44].

Diffusion sampling. Recall that we propose confidence-based sampling strategy to adaptively adjust sampling range, where we generate multiple samples for the diffusion model. The confidence is learned in an unsupervised manner (Eq. 15) and used to linearly adjust the sampling range (Eq. 13). In this ablation experiment, we first remove

TABLE 5
Evaluation of DiffMVS with different DDIM sampling steps T_s on DTU.

T_s	Depth Error (mm) ↓	Acc. ↓	Comp. ↓	Overall ↓
1	4.68	**0.318**	**0.297**	**0.308**
2	**4.63**	0.321	**0.297**	0.309

learned confidence and use a single sample, instead of generating multiple samples in a local range. As shown in Row IX of Table 4, the performance on both datasets is the worst. Second, we remove learned confidence and generate multiple samples in a *fixed* sampling range. As shown in Row X of Table 4, the performance improves when compared with using single sample (Row IX). However, it is worse than our model (Row XII). Third, we use learned confidence for regularization only (Eq. 15), *i.e.*, we do not use confidence to adjust the sampling range. We observe that the zero-shot generalization ability on ETH3D (Row XI) is better than the ablation model without confidence (Row X). However, the performance on both datasets is still worse than our model since we further use confidence to adaptively adjust sampling range.

In Fig. 10, we visualize the learned confidence on the validation set of BlendedMVS [44]. Though we do not explicitly supervise the confidence during training, it reliably reflects the depth error distribution, *i.e.*, high confidence in regions with low depth error and low confidence in those with high depth error.

Diffusion efficiency. In Sec. 3.5, we design a lightweight and effective diffusion network to denoise the depth residual, which takes advantage of U-Net and convolutional GRU. To further study its effectiveness, we replace this design by using a single U-Net and stacked U-Nets as the diffusion network, respectively. Note that for stacked U-Nets, we use $K = 4$ U-Nets for fair comparison and each U-Net includes sampling, condition encoding and update. As shown in Row XIII and XIV of Table 4, the performance of single U-Net degrades significantly on both DTU and ETH3D, while the performance of stacked U-Nets drops slightly on DTU but obviously on ETH3D. Comparing Row XIII and XIV in Table 4, we conjecture that it is important to use multiple updates in one diffusion sampling to improve convergence since only limited information is used in each update. The comparison between Row XIV and XV in Table 4 verifies that the hidden state feature of the convolutional GRU can help the denoising process. Moreover, by introducing the convolutional GRU, the U-Net in our diffusion network can be reused to perform iterative refinement, instead of stacking multiple U-Nets. This effectively reduces model size with 33.7% less parameters compared with stacked U-Nets, making our method more suitable for application scenarios with limited resources.

DDIM sampling. By default, we set DDIM sampling timestep as $T_s = 1$. We change T_s and summarize the results in Table 5. When we increase T_s, we find that depth accuracy slightly improves, while the quality of the point cloud is almost the same. Since we focus on both accuracy and efficiency, we set $T_s = 1$ to reduce run-time. Unlike previous depth estimation works [60] using diffusion models that initialize with pure random noise, our framework predicts an initial depth map $\boldsymbol{D}_{\text{init}}$ and then uses a diffusion model to refine it. Although not accurate enough, the initial depth map provides a relatively good initial value for the diffusion model to search for more accurate depth values in the neighborhood of $\boldsymbol{D}_{\text{init}}$. Therefore, our framework requires fewer sampling timesteps and thus reduces the run-time.

TABLE 6
Evaluation of CasDiffMVS with different noise scales on DTU and ETH3D. Note that on ETH3D, the noise scales become one-fourth of the original values because of our noise-scaling strategy.

(σ_2, σ_3)	DTU testing set			ETH3D training set		
	Acc. ↓	Comp. ↓	Overall ↓	Acc. ↑	Comp. ↑	F_1-score ↑
(0.25,0.05)	0.361	0.323	0.342	70.01	74.94	71.73
(0.50,0.10)	**0.310**	**0.286**	**0.298**	**79.93**	75.20	**76.76**
(1.00,0.20)	0.315	0.296	0.306	76.95	**76.20**	76.09

TABLE 7
Evaluation of DiffMVS with different random seeds on DTU.

Acc.(mm) ↓	Comp.(mm) ↓	Overall(mm) ↓
0.3190 ± 0.0002	0.2976 ± 0.0006	0.3083 ± 0.0002

Noise scale. To investigate the influence of noise scales in our diffusion model, we set different values for (σ_2, σ_3) to train our CasDiffMVS and evaluate it on DTU test set and ETH3D training set. The results are reported in Table 6. We observe that our default setting $(0.50, 0.10)$ achieves the best reconstruction performance on both sets. For the smaller one $(0.25, 0.05)$, the random noise is too small to help our CasDiffMVS to avoid local minima and thus the performance degrades significantly. For the larger one $(1.00, 0.20)$, the random noise is beneficial to our CasDiffMVS to escape local minima but the performance slightly drops. This is because our CasDiffMVS aims to refine a reasonable initial depth which does not contain too much noise. Too much noise will contaminate the initial depth and prevent our diffusion condition from generating favorable guidance. Therefore, our default setting can better reflect the noise level of the initial depth and introduce favorable perturbations to avoid local minima.

Random seeds. We evaluate DiffMVS on DTU with 10 different random seeds and summarize the results (mean and standard variance) in Table 7. We observe that the quantitative results are stable with different random seeds. In previous monocular depth estimation methods [60] with diffusion models, the depth map is fully initialized using random noise, and thus specific modules are proposed to combat the diversity of diffusion models. In contrast, we use diffusion models to refine a reasonable initial depth map, $\boldsymbol{D}_{\text{init}}$, and search for more accurate depth values in the neighborhood of $\boldsymbol{D}_{\text{init}}$. Therefore, our framework is more stable w.r.t. random seeds.

5 Conclusion

In this paper, we introduce diffusion models in MVS for efficient and accurate reconstruction. We formulate depth refinement as a conditional diffusion process and propose

a condition encoder for guidance. Moreover, we propose a confidence-based sampling strategy to adaptively adjust the per-pixel sampling range and thus improve accuracy. Instead of using large denoising U-Nets as classical diffusion models, we design a lightweight diffusion network, which combines a lightweight 2D U-Net and convolutional GRU, to improve both performance and efficiency. Based on our framework, we propose two novel MVS methods, DiffMVS and CasDiffMVS. Extensive experiments demonstrate that DiffMVS achieves competitive performance with state-of-the-art efficiency in both run-time and memory, while CasDiffMVS achieves SOTA performance on DTU, Tanks & Temples and ETH3D. Because of the high efficiency, impressive performance and lightweight structure, our methods can serve as new strong baselines for future research in MVS.

REFERENCES

[1] Y. Furukawa, C. Hernández *et al.*, "Multi-view stereo: A tutorial," *Foundations and Trends® in Computer Graphics and Vision*, vol. 9, no. 1-2, pp. 1–148, 2015. 1

[2] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *CVPR*, 2016. 1, 5, 9, 11

[3] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "MVSNet: Depth inference for unstructured multi-view stereo," in *ECCV*, 2018. 1, 3, 7, 8

[4] Q. Xu and W. Tao, "Multi-scale geometric consistency guided multi-view stereo," in *CVPR*, 2019. 1, 2, 3, 7, 9, 11

[5] R. Peng, R. Wang, Z. Wang, Y. Lai, and R. Wang, "Rethinking depth estimation for multi-view stereo: A unified representation," in *CVPR*, 2022, pp. 8645–8654. 1, 3, 8, 9, 11

[6] Y. Ding, W. Yuan, Q. Zhu, H. Zhang, X. Liu, Y. Wang, and X. Liu, "Transmvsnet: Global context-aware multi-view stereo network with transformers," in *CVPR*, 2022, pp. 8585–8594. 1, 3, 4, 7, 8, 9, 11

[7] Z. Zhang, R. Peng, Y. Hu, and R. Wang, "Geomvsnet: Learning multi-view stereo with geometry perception," in *CVPR*, 2023, pp. 21 508–21 518. 1, 8, 9, 11

[8] T. Liu, X. Ye, W. Zhao, Z. Pan, M. Shi, and Z. Cao, "When epipolar constraint meets non-local operators in multi-view stereo," in *ICCV*, 2023, pp. 18 088–18 097. 1, 8, 9, 11

[9] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *TOG*, 2017. 1, 3, 7, 8, 9, 10

[10] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, "A multi-view stereo benchmark with high-resolution images and multi-camera videos," in *CVPR*, 2017. 1, 3, 7, 8, 9, 10, 11, 12

[11] R. Collins, "A space-sweep approach to true multi-image matching," in *CVPR*, 1996. 1, 2

[12] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nistér, and M. Pollefeys, "Real-time visibility-based fusion of depth maps," in *ICCV*, 2007, pp. 1–8. 1

[13] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla, "Using multiple hypotheses to improve depth-maps for multi-view stereo," in *ECCV*, 2008, pp. 766–779. 1

[14] S. Galliani, K. Lasinger, and K. Schindler, "Massively parallel multiview stereopsis by surface normal diffusion," in *ICCV*, 2015. 1, 3, 8

[15] Q. Xu and W. Tao, "Learning inverse depth regression for multi-view stereo with correlation cost volume," in *AAAI*, 2020. 1, 5

[16] Y. Yao, Z. Luo, S. Li, T. Shen, T. Fang, and L. Quan, "Recurrent MVSNet for high-resolution multi-view stereo depth inference," in *CVPR*, 2019. 2, 3, 5, 8, 9

[17] Z. Wei, Q. Zhu, C. Min, Y. Chen, and G. Wang, "Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network," *arXiv preprint arXiv:2108.03824*, 2021. 2, 3, 8

[18] Q. Xu, M. R. Oswald, W. Tao, M. Pollefeys, and Z. Cui, "Non-local recurrent regularization networks for multi-view stereo," *arXiv preprint arXiv:2110.06436*, 2021. 2

[19] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan, "Cascade cost volume for high-resolution multi-view stereo and stereo matching," in *CVPR*, 2020. 2, 3, 4, 7, 8, 9, 11

[20] S. Cheng, Z. Xu, S. Zhu, Z. Li, L. E. Li, R. Ramamoorthi, and H. Su, "Deep stereo using adaptive thin volume representation with uncertainty awareness," in *CVPR*, 2020. 2, 3, 7

[21] J. Yang, W. Mao, J. M. Alvarez, and M. Liu, "Cost volume pyramid based depth inference for multi-view stereo," in *CVPR*, 2020. 2, 3

[22] F. Wang, S. Galliani, C. Vogel, P. Speciale, and M. Pollefeys, "Patchmatchnet: Learned multi-view patchmatch stereo," in *CVPR*, June 2021, pp. 14 194–14 203. 2, 3, 4, 5, 6, 7, 8, 9, 11

[23] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020. 2, 3, 4, 5, 6, 7, 8

[24] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *CVPR*, 2022, pp. 10 684–10 695. 2, 3, 4, 6

[25] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020. 2, 3, 4

[26] R. Shao, Z. Zheng, H. Zhang, J. Sun, and Y. Liu, "Diffustereo: High quality human reconstruction via diffusion-based stereo using sparse cameras," in *ECCV*, 2022, pp. 702–720. 2

[27] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *ECCV*. Springer, 2020, pp. 402–419. 2, 3, 6, 7

[28] F. Wang, S. Galliani, C. Vogel, and M. Pollefeys, "Itermvs: iterative probability estimation for efficient multi-view stereo," in *CVPR*, 2022, pp. 8606–8615. 2, 3, 5, 6, 7, 8, 9, 11

[29] C. Ren, Q. Xu, S. Zhang, and J. Yang, "Hierarchical prior mining for non-local multi-view stereo," in *ICCV*, 2023, pp. 3611–3620. 2, 3, 7, 9, 11

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. 2

[31] S. Chen, P. Sun, Y. Song, and P. Luo, "Diffusiondet: Diffusion model for object detection," *arXiv preprint arXiv:2211.09788*, 2022. 2, 4, 6

[32] A. O. Ulusoy, M. J. Black, and A. Geiger, "Semantic multi-view stereo: Jointly estimating objects and voxels," in *CVPR*, 2017. 3

[33] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *IJCV*, vol. 38, no. 3, pp. 199–218, 2000. 3

[34] S. M. Seitz and C. R. Dyer, "Photorealistic scene reconstruction by voxel coloring," *IJCV*, vol. 35, no. 2, pp. 151–173, 1999. 3

[35] I. Kostrikov, E. Horbert, and B. Leibe, "Probabilistic labeling cost for high-accuracy multi-view reconstruction," in *CVPR*, 2014, pp. 1534–1541. 3

[36] M. Lhuillier and L. Quan, "A quasi-dense approach to surface reconstruction from uncalibrated images," *PAMI*, vol. 27, no. 3, pp. 418–433, 2005. 3

[37] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *PAMI*, 2010. 3

[38] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys, "Pixel-wise view selection for unstructured multi-view stereo," in *ECCV*, 2016, pp. 501–518. 3, 8

[39] Q. Xu and W. Tao, "Planar prior assisted patchmatch multi-view stereo," in *AAAI*, 2020, pp. 12 516–12 523. 3

[40] Q. Xu, W. Kong, W. Tao, and M. Pollefeys, "Multi-scale geometric consistency guided and planar prior assisted multi-view stereo," *PAMI*, 2022. 3, 9

[41] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009. 3

[42] Y. Wang, Z. Zeng, T. Guan, W. Yang, Z. Chen, W. Liu, L. Xu, and Y. Luo, "Adaptive patch deformation for textureless-resilient multi-view stereo," in *CVPR*, 2023, pp. 1621–1630. 3

[43] H. Aanæs, R. R. Jensen, G. Vogiatzis, E. Tola, and A. B. Dahl, "Large-scale data for multiple-view stereopsis," *IJCV*, 2016. 3, 7, 8, 9, 11, 12

[44] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan, "Blendedmvs: A large-scale dataset for generalized multi-view stereo networks," in *CVPR*, 2020, pp. 1790–1799. 3, 7, 8, 11, 12, 13

[45] R. T. Collins, "A space-sweep approach to true multi-image matching," in *CVPR*, 1996, pp. 358–363. 3

[46] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241. 3

[47] J. Yan, Z. Wei, H. Yi, M. Ding, R. Zhang, Y. Chen, G. Wang, and Y.-W. Tai, "Dense hybrid recurrent multi-view stereo net with dynamic consistency checking," in *ECCV*. Springer, 2020, pp. 674–689. 3, 8, 9

[48] Q. Xu and W. Tao, "PVSNet: Pixelwise visibility-aware multi-view stereo network," *ArXiv*, 2020. 3, 5

[49] J. Zhang, Y. Yao, S. Li, Z. Luo, and T. Fang, "Visibility-aware multi-view stereo network," in *BMVC*, 2020. 3

[50] X. Ma, Y. Gong, Q. Wang, J. Huang, L. Chen, and F. Yu, "Epp-mvsnet: Epipolar-assembling based depth prediction for multi-view stereo," in *ICCV*, 2021, pp. 5732–5740. 3, 8, 9, 11

[51] Q. Xu, W. Su, Y. Qi, W. Tao, and M. Pollefeys, "Learning inverse depth regression for pixelwise visibility-aware multi-view stereo networks," *IJCV*, vol. 130, no. 8, pp. 2040–2059, 2022. 3, 5, 8, 9, 11

[52] X. Wang, Z. Zhu, G. Huang, F. Qin, Y. Ye, Y. He, X. Chi, and X. Wang, "Mvster: epipolar transformer for efficient multi-view stereo," in *ECCV*, 2022, pp. 573–591. 3, 4, 8, 9, 11

[53] K. Cho, B. V. Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014. 3

[54] S. Wang, B. Li, and Y. Dai, "Efficient multi-view stereo by iterative dynamic cost volume," in *CVPR*, 2022, pp. 8655–8664. 3, 4, 6, 7, 8, 9, 11

[55] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in neural information processing systems*, vol. 32, 2019. 3, 4

[56] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in Neural Information Processing Systems*, vol. 34, pp. 8780–8794, 2021. 3

[57] J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet, "Video diffusion models," *NeurIPS*, vol. 35, pp. 8633–8646, 2022. 3

[58] Y. Duan, X. Guo, and Z. Zhu, "Diffusiondepth: Diffusion denoising approach for monocular depth estimation," *arXiv preprint arXiv:2303.05021*, 2023. 3

[59] Y. Ji, Z. Chen, E. Xie, L. Hong, X. Liu, Z. Liu, T. Lu, Z. Li, and P. Luo, "Ddp: Diffusion model for dense visual prediction," *arXiv preprint arXiv:2303.17559*, 2023. 3, 4

[60] B. Ke, A. Obukhov, S. Huang, N. Metzger, R. C. Daudt, and K. Schindler, "Repurposing diffusion-based image generators for monocular depth estimation," in *CVPR*, 2024, pp. 9492–9502. 3, 13

[61] X. Fu, W. Yin, M. Hu, K. Wang, Y. Ma, P. Tan, S. Shen, D. Lin, and X. Long, "Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image," in *ECCV*. Springer, 2024, pp. 241–258. 3

[62] C. Ye, L. Qiu, X. Gu, Q. Zuo, Y. Wu, Z. Dong, L. Bo, Y. Xiu, and X. Han, "Stablenormal: Reducing diffusion variance for stable and sharp normal," *ACM Transactions on Graphics (TOG)*, vol. 43, no. 6, pp. 1–18, 2024. 3

[63] H. Guo, H. Zhu, S. Peng, H. Lin, Y. Yan, T. Xie, W. Wang, X. Zhou, and H. Bao, "Multi-view reconstruction via sfm-guided monocular depth estimation," in *CVPR*, 2025. 3

[64] J. Nam, G. Lee, S. Kim, H. Kim, H. Cho, S. Kim, and S. Kim, "Diffusion model for dense matching," in *The Twelfth International Conference on Learning Representations*, 2023. 3, 4

[65] J. Wang, C. Rupprecht, and D. Novotny, "Posediffusion: Solving pose estimation via diffusion-aided bundle adjustment," in *ICCV*, 2023, pp. 9773–9783. 3

[66] J. Y. Zhang, A. Lin, M. Kumar, T.-H. Yang, D. Ramanan, and S. Tulsiani, "Cameras as rays: Pose estimation via ray diffusion," *ICLR*, 2024. 3

[67] Y. Lu, J. Zhang, T. Fang, J.-D. Nahmias, Y. Tsin, L. Quan, X. Cao, Y. Yao, and S. Li, "Matrix3d: Large photogrammetry model all-in-one," *CVPR*, 2025. 3

[68] S. Saxena, C. Herrmann, J. Hur, A. Kar, M. Norouzi, D. Sun, and D. J. Fleet, "The surprising effectiveness of diffusion models for optical flow and monocular depth estimation," *NeurIPS*, vol. 36, pp. 39 443–39 469, 2023. 3

[69] Q. Dong, B. Zhao, and Y. Fu, "Open-ddvm: A reproduction and extension of diffusion model for optical flow estimation," *arXiv preprint arXiv:2312.01746*, 2023. 3

[70] M.-G. Park and K.-J. Yoon, "Learning and selecting confidence measures for robust stereo matching," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 6, pp. 1397–1411, 2018. 3

[71] M. Poggi and S. Mattoccia, "Learning from scratch a confidence measure." in *BMVC*, 2016. 3

[72] R. Haeusler, R. Nair, and D. Kondermann, "Ensemble learning for confidence measures in stereo vision," in *CVPR*, 2013, pp. 305–312. 3

[73] F. Tosi, M. Poggi, A. Benincasa, and S. Mattoccia, "Beyond local reasoning for stereo confidence estimation with deep learning," in *ECCV*, 2018, pp. 319–334. 3

[74] A. Shaked and L. Wolf, "Improved stereo matching with constant highway networks and reflective confidence learning," in *CVPR*, 2017, pp. 4641–4650. 3

[75] S. Kim, S. Kim, D. Min, and K. Sohn, "Laf-net: Locally adaptive fusion networks for stereo confidence estimation," in *CVPR*, 2019, pp. 205–214. 3

[76] W. Su, Q. Xu, and W. Tao, "Uncertainty guided multi-view stereo network for depth estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 11, pp. 7796–7808, 2022. 3

[77] S. Saxena, A. Kar, M. Norouzi, and D. J. Fleet, "Monocular depth estimation using diffusion models," *arXiv preprint arXiv:2302.14816*, 2023. 4

[78] L. Karazija, I. Laina, A. Vedaldi, and C. Rupprecht, "Diffusion models for zero-shot open-vocabulary segmentation," *arXiv preprint arXiv:2306.09316*, 2023. 4

[79] J. Liu, G. Wang, W. Ye, C. Jiang, J. Han, Z. Liu, G. Zhang, D. Du, and H. Wang, "Difflow3d: Toward robust uncertainty-aware scene flow estimation with diffusion model," *arXiv preprint arXiv:2311.17456*, 2023. 4, 6

[80] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020. 4, 8

[81] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, "Dust3r: Geometric 3d vision made easy," in *CVPR*, 2024, pp. 20 697–20 709. 4

[82] J. Ho and T. Salimans, "Classifier-free diffusion guidance," *arXiv preprint arXiv:2207.12598*, 2022. 5

[83] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," *CVPR*, 2017. 5

[84] L. Lipson, Z. Teed, and J. Deng, "Raft-stereo: Multilevel recurrent field transforms for stereo matching," *arXiv preprint arXiv:2109.07547*, 2021. 6

[85] Z. Ma, Z. Teed, and J. Deng, "Multiview stereo with cascaded epipolar raft," *arXiv preprint arXiv:2205.04502*, 2022. 6

[86] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang, "Surfacenet: An end-to-end 3D neural network for multiview stereopsis," in *ICCV*, 2017. 7

[87] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *ArXiv*, 2019. 8

[88] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." in *ICLR*, 2015. 8

[89] J. Chang, J. He, T. Zhang, J. Yu, and F. Wu, "Ei-mvsnet: Epipolar-guided multi-view stereo network with interval-aware label," *TIP*, vol. 33, pp. 753–766, 2024. 8, 9, 11

[90] S. Wang, B. Li, and Y. Dai, "Efficient multi-view stereo by dynamic cost volume and cross-scale propagation," *CSVT*, 2024. 8, 9, 11

[91] V. K. Vats, S. Joshi, D. J. Crandall, M. A. Reza, and S.-h. Jung, "Gc-mvsnet: Multi-view, multi-scale, geometrically-consistent multi-view stereo," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 3242–3252. 8, 9

[92] W. Su and W. Tao, "Context-aware multi-view stereo network for efficient edge-preserving depth estimation," *IJCV*, pp. 1–25, 2025. 8, 9