

Learning-based Multi-View Stereo: A Survey

Fangjinhua Wang^{*†}, Qingtian Zhu^{*}, Di Chang^{*}, Quankai Gao, Junlin Han, Tong Zhang,
Richard Hartley, *Fellow, IEEE*, Marc Pollefeys[‡], *Fellow, IEEE*

Abstract—3D reconstruction aims to recover the dense 3D structure of a scene. It plays an essential role in various applications such as Augmented/Virtual Reality (AR/VR), autonomous driving and robotics. Leveraging multiple views of a scene captured from different viewpoints, Multi-View Stereo (MVS) algorithms synthesize a comprehensive 3D representation, enabling precise reconstruction in complex environments. Due to its efficiency and effectiveness, MVS has become a pivotal method for image-based 3D reconstruction. Recently, with the success of deep learning, many learning-based MVS methods have been proposed, achieving impressive performance against traditional methods. We categorize these learning-based methods as: depth map-based, voxel-based, NeRF-based, 3D Gaussian Splatting-based, and large feed-forward methods. Among these, we focus significantly on depth map-based methods, which are the main family of MVS due to their conciseness, flexibility and scalability. In this survey, we provide a comprehensive review of the literature at the time of this writing. We investigate these learning-based methods, summarize their performances on popular benchmarks, and discuss promising future research directions in this area.

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

1 INTRODUCTION

3D reconstruction describes the general task of recovering the 3D structure of a scene. It is widely employed in augmented/virtual reality (AR/VR), autonomous driving, and robotics [1]. The advancement of 3D acquisition techniques has led to the increased affordability and reliability of depth sensors, such as depth cameras and LiDARs. These sensors are extensively utilized for real-time tasks, enabling rough estimations of the surrounding environment, such as simultaneous localization and mapping (SLAM) [2], [3], [4], [5] or dense reconstruction [6], [7], [8]. Nevertheless, depth maps captured by such sensors tend to be partial and sparse, resulting in incomplete 3D representations with limited geometric details. In addition, these sensors are active and usually consume a lot of power. In contrast, camera-based solutions, commonly found in edge devices like smartphones and AR/VR headsets, offer a more economically viable alternative for 3D reconstruction.

One fundamental technique in image-based 3D reconstruction is Multi-View Stereo (MVS). Given a set of calibrated images, MVS aims to reconstruct dense 3D geometry for an observed scene. Based on the scene representations, traditional MVS methods can be divided into three categories: volumetric, point cloud, and depth map. Volumetric methods [9], [10], [11], [12] discretize the 3D space into voxels and label each as inside or outside of the surface. They

are limited to scenes of small scale due to the large memory consumption. Point cloud methods [13], [14] operate directly on 3D points and often employ propagation to gradually densify the reconstruction. As point cloud propagation occurs sequentially, these methods are difficult to parallelize, leading to longer processing times. [15]. In addition, the irregularity and large size of point cloud are also not very suitable for deep learning, especially in large-scale scenes. In contrast, methods relying on depth maps [16], [17], [18], [19], [20] use patch matching with photometric consistency to estimate the depth maps for individual images. Subsequently, these depth maps are fused into a dense representation, *e.g.*, point cloud or mesh, as a post-processing step. Such design decouples the reconstruction problem into per-view depth estimation and depth fusion, which explicitly improves the flexibility and scalability. Although MVS has been studied extensively for several decades, traditional MVS methods rely on hand-crafted matching metrics and thus encounter challenges in handling various conditions, *e.g.*, illumination changes, low-textured areas, and non-Lambertian surfaces [21], [22], [23], [24].

To overcome these challenges, recent works [15], [25] have shifted towards learning-based approaches, have adopted learning-based approaches, using convolutional neural networks (CNNs) that have excelled in 2D vision tasks. These methods have significantly outperformed traditional methods on various benchmarks [21], [22], [23], [26].

In this survey, we categorize existing learning-based MVS methods based on their characteristics as follows: depth map-based, voxel-based, NeRF-based, 3D Gaussian Splatting-based and large feed-forward methods. Voxel-based methods estimate the geometry with volumetric representation and implicit function, *e.g.*, Signed Distance Functions (SDF). They are limited to small-scale scenes due to the high memory consumption. NeRF and 3D Gaussian Splatting-based methods adapt NeRF [27] and 3D Gaussian Splatting [28], which are originally used for novel

• Fangjinhua Wang and Marc Pollefeys are with the Department of Computer Science, ETH Zurich, Switzerland.
 • Qingtian Zhu is with the Graduate School of Information Science and Technology, University of Tokyo, Japan.
 • Di Chang and Quankai Gao are with the Department of Computer Science, University of Southern California, USA.
 • Junlin Han is with the Department of Engineering Science, University of Oxford, UK.
 • Tong Zhang is with the School of Computer and Communication Sciences, EPFL, Switzerland.
 • Richard Hartley is with Australian National University, Australia.
 • Marc Pollefeys is additionally with Microsoft, Zurich.

^{*}: Equal contribution. [†]: Project lead. [‡]: Corresponding author.

view synthesis, to extract surface from the implicit field or point cloud. They typically need to optimize the geometry for each new scene, which needs lots of run-time and memory consumption. Large feed-forward methods typically use large transformer models to directly learn the 3D representation from given images. They require massive computation because of the huge network, and are mainly limited to object-level scenes [29]. Depth map-based methods utilize deep learning in depth estimation and then fuse depth maps with traditional fusion algorithm. Comparatively, depth map-based methods [15], [25] are the main family of learning-based methods since they inherit the advantages from those traditional methods [17], [18], [19], [20] by decoupling 3D reconstruction into depth estimation and depth fusion. Therefore, we focus more on the depth map-based methods and discuss them in details.

For clarity, we further categorize depth map-based methods into *online* and *offline* methods, shown in Fig. 1. Specifically, online MVS methods [25], [30] usually perform multi-view depth estimation with a video sequence of low-resolution images, *e.g.*, ScanNet [26]. These methods typically rely on low-resolution inputs and lightweight network structures to ensure fast inference, prioritizing speed and simplicity. The main goal of online MVS is to deliver real-time reconstruction for time-sensitive applications such as augmented reality and live video processing. In contrast, offline MVS methods [15], [31] prioritize better quality reconstruction at the cost of computation. These methods excel in multi-view depth estimation and point cloud fusion using high-resolution image sets, *e.g.*, DTU [21], Tanks and Temples [22] and ETH3D [23]. By operating on higher-resolution inputs and employing complex network, they thoroughly analyze the multi-view information in the images. The emphasis on achieving high accuracy and capturing intricate details often comes at the cost of requiring considerable computational resources and time. As a result, offline MVS is frequently employed in applications demanding precise and photorealistic scene representations, including 3D modeling and archaeological reconstruction.

In summary, our survey covers the most recent literature on learning-based MVS methods, including four main families: depth map-based, voxel-based, NeRF-based, 3D Gaussian Splatting-based and large feed-forward methods. We provide a comprehensive review and insights on different aspects, including the pipelines and algorithmic intricacies. Moreover, we summarize the performance of the reviewed methods on different benchmarks, and discuss the potential future directions for deep learning-based MVS methods.

2 PRELIMINARIES

Depth map-based MVS, including most traditional and learning-based methods, typically consists of several components: camera calibration, view selection, multi-view depth estimation and depth fusion. In this section, we introduce these components to provide readers a clear picture of the MVS. Note that camera calibration and view selection are components for other learning-based methods as well. Sec. 2.1 introduces camera calibration with Structure from Motion (SfM) or SLAM. Sec. 2.2 discusses how to select neighboring views to reconstruct the geometry. Sec. 2.3

explains how to build cost volumes in learning-based MVS methods with plane sweep [32]. Sec. 2.4 introduces the typical depth fusion strategies after depth estimation. Sec. 2.5 lists common datasets and benchmarks for MVS and Sec. 2.6 summarizes the common evaluation metrics.

2.1 Camera Calibration

Camera calibration is a process of determining the intrinsic and extrinsic parameters of a camera to understand its geometry and characteristics accurately [57]. It serves as the foundational step in MVS, ensuring that the subsequent reconstruction process is built on accurate and consistent geometric information, ultimately leading to a more reliable and precise 3D representation of the scene. Typically, obtaining calibrated camera parameters is usually achieved by running off-the-shelf SfM algorithms [17], [58] or SLAM [59], which jointly optimize sparse triangulated 3D points and camera parameters. The camera parameters include the extrinsic matrix $\mathbf{T} = [\mathbf{R}|t]$ and intrinsic matrix \mathbf{K} . Depth map-based MVS methods [15], [31], [39] require a bounded depth range $[d_{\min}, d_{\max}]$ to improve the estimation accuracy. For offline methods [15], [31], the depth range can be estimated by projecting the sparse point cloud from SfM to each viewpoint and compute the minimum and maximum z values [15]. In contrast, online methods [25], [30] usually set constant depth ranges, *e.g.*, $[0.25m, 20.00m]$, since the scene scale is usually fixed as room.

2.2 View Selection

The selection of views is an important step for reconstruction. It is important to balance triangulation quality, matching accuracy, and view frustum overlap [30]. Currently, there are two main strategies for view selection.

First, for most online MVS depth estimation methods [25], [30], [60], a frame is selected as a keyframe when its pose has sufficient difference compared with the previous keyframe. Then each keyframe adopts several previous keyframes to estimate depth. GP-MVS [60] proposes a heuristic pose-distance measure as:

$$\text{disc}(\mathbf{T}_{ij}) = \sqrt{\|\mathbf{t}_{ij}\|^2 + \frac{2}{3} \text{tr}(\mathbb{I} - \mathbf{R}_{ij})}, \quad (1)$$

where $\mathbf{T}_{ij} = [\mathbf{R}_{ij}|t_{ij}]$ is the relative transformation between view i and j . This strategy is used by many following methods [30], [34].

Second, for most offline MVS methods [15], [31], [36], view selection is done with the sparse point cloud obtained by Structure-from-Motion [33], [58]. For a reference view i , MVSNet [15] computes a score $s(i, j) = \sum_{\mathbf{P}} \eta(\theta_{ij}(\mathbf{P}))$ for the neighboring view j , where \mathbf{P} is a 3D point observed by both view i and j . $\theta_{ij}(\mathbf{P}) = (180/\pi) \arccos((\mathbf{c}_i - \mathbf{P}) \cdot (\mathbf{c}_j - \mathbf{P}))$ represents the baseline angle for \mathbf{P} and $\mathbf{c}_i, \mathbf{c}_j$ are the camera centers. $\eta(\cdot)$ is a piece-wise Gaussian function [61] to favor a certain baseline angle θ_0 :

$$\eta(\theta) = \begin{cases} \exp\left(-\frac{(\theta - \theta_0)^2}{2\sigma_1^2}\right), & \theta \leq \theta_0 \\ \exp\left(-\frac{(\theta - \theta_0)^2}{2\sigma_2^2}\right), & \theta > \theta_0 \end{cases} \quad (2)$$

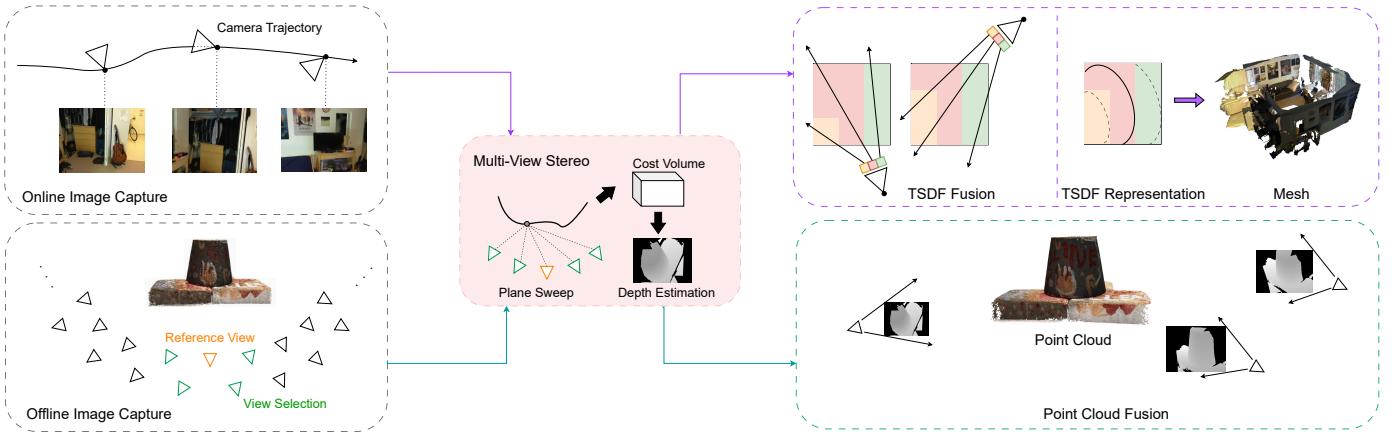


Fig. 1. An overall illustration of both online and offline depth map-based MVS pipelines. Online MVS usually deals with sequential data, e.g., video, and employs TSDF volumes as an intermediate representation for mesh extraction. Given a full set of images, offline MVS holds the global information of the captured scene, and usually fuses estimated depth maps into a point cloud with filtering.

where θ_0 , σ_0 , σ_1 are hyper-parameters. Then the view selection is done by choosing neighboring views with highest scores. Almost all the following offline MVS methods [31], [36], [39] use the same strategy.

2.3 Multi-view Depth Estimation with Plane Sweep

To form a structured data format that is more suitable for convolution operations, most learning-based MVS methods rely on the plane sweep algorithm [32] to calculate matching costs. The plane sweep algorithm discretizes the depth space with a set of fronto-parallel planes along the depth direction. This practice is deeply inspired by learning-based binocular stereo methods [62], [63], [64], which assess matching costs for a set of disparity hypotheses and subsequently estimate the disparity.

In a nutshell, the plane sweep algorithm entails iteratively sweeping planes through the object space, computing homographies between images, and selecting depth values based on consensus among different views, ultimately facilitating accurate 3D reconstruction. The plane sweep algorithm discretizes the depth space using a series of parallel planes along the depth direction. It operates by sweeping a conceptual plane through the object space and evaluating the spatial distribution of geometric surfaces.

In practice, we divide the depth range, which is either manually set [25] or estimated by SfM [15], into discrete samples and assign hypotheses at these values. To map coordinates at depth hypothesis d , homography transformation [15] $\mathbf{p}_i(d) \sim \mathbf{H}_i(d) \cdot \mathbf{p}$, is applied, where $\mathbf{p}_i(d)$ is the corresponding pixel in the i -th source view for pixel \mathbf{p} of the reference view. The homography $\mathbf{H}_i(d)$ between the i -th source view and the reference view 0 can be computed as:

$$\mathbf{H}_i(d) = \mathbf{K}_i \mathbf{R}_i \left(\mathbb{I} - \frac{(-\mathbf{R}_0^\top \mathbf{t}_0 + \mathbf{R}_i^\top \mathbf{t}_i) \mathbf{n}^\top \mathbf{R}_0}{d} \right) \mathbf{R}_0^\top \mathbf{K}_0^{-1}, \quad (3)$$

where \mathbf{K}_0 , \mathbf{K}_i denote camera intrinsics, $[\mathbf{R}_0|\mathbf{t}_0]$, and $[\mathbf{R}_i|\mathbf{t}_i]$ denote camera extrinsics, \mathbf{n} denote the principle axis of the reference view. Equivalently, we can also project a reference pixel into source views with depth hypothesis [35], [39], [40]. We compute $\mathbf{p}_i(d)$ in the i -th source view for pixel \mathbf{p} in the reference and depth hypothesis d as follows:

$$\mathbf{p}_i(d) = \mathbf{K}_i \cdot \left(\mathbf{R}_i \mathbf{R}_0^\top \cdot (\mathbf{K}_0^{-1} \cdot \mathbf{p} \cdot d) - \mathbf{R}_i \mathbf{R}_0^\top \cdot \mathbf{t}_0 + \mathbf{t}_i \right). \quad (4)$$

The warped source feature maps are then obtained via differentiable interpolation. To better elaborate the process of plane sweep, we include a toy example in the supplementary. The photometric similarity (or matching cost) is measured in a one-to-many manner between reference and warped source features for the following depth estimation, which will be discussed in Sec. 3.2. By performing plane sweep, we obtain a cost volume with a regular spatial shape, making it easy for CNNs to process in parallel.

2.4 Depth Fusion

For depth map-based MVS, after estimating all the depth maps, we need to fuse them into a dense 3D representation, e.g., point cloud or mesh. Online MVS methods [30], [34] usually adopt TSDF (Truncated Signed Distance Function) fusion [65], [66] to fuse the depth maps into a TSDF volume and then use Marching Cube [67] to extract the mesh. However, there usually exist outliers in the depth maps, which may reduce the reconstruction accuracy. To overcome this problem and improve the accuracy, offline MVS methods [15], [31], [39] typically filter the depth maps before fusing into a point cloud, which is motivated by Galliani *et al.* [18]. There are two main filtering steps: photometric consistency filtering and geometric consistency filtering [15]. For more details, please refer to the supplementary.

2.5 Datasets and Benchmarks

Along with this survey, we also include a brief summary of commonly used public MVS datasets and benchmarks for training and evaluation. For more details, please refer to supplementary.

2.6 Evaluation Metrics

Based on the ground truth, e.g., depth maps or point clouds, evaluation can be categorized into 2D and 3D metrics.

2D Metrics: 2D depth-based metrics are commonly used by online MVS methods [25], [30] to evaluate the depth

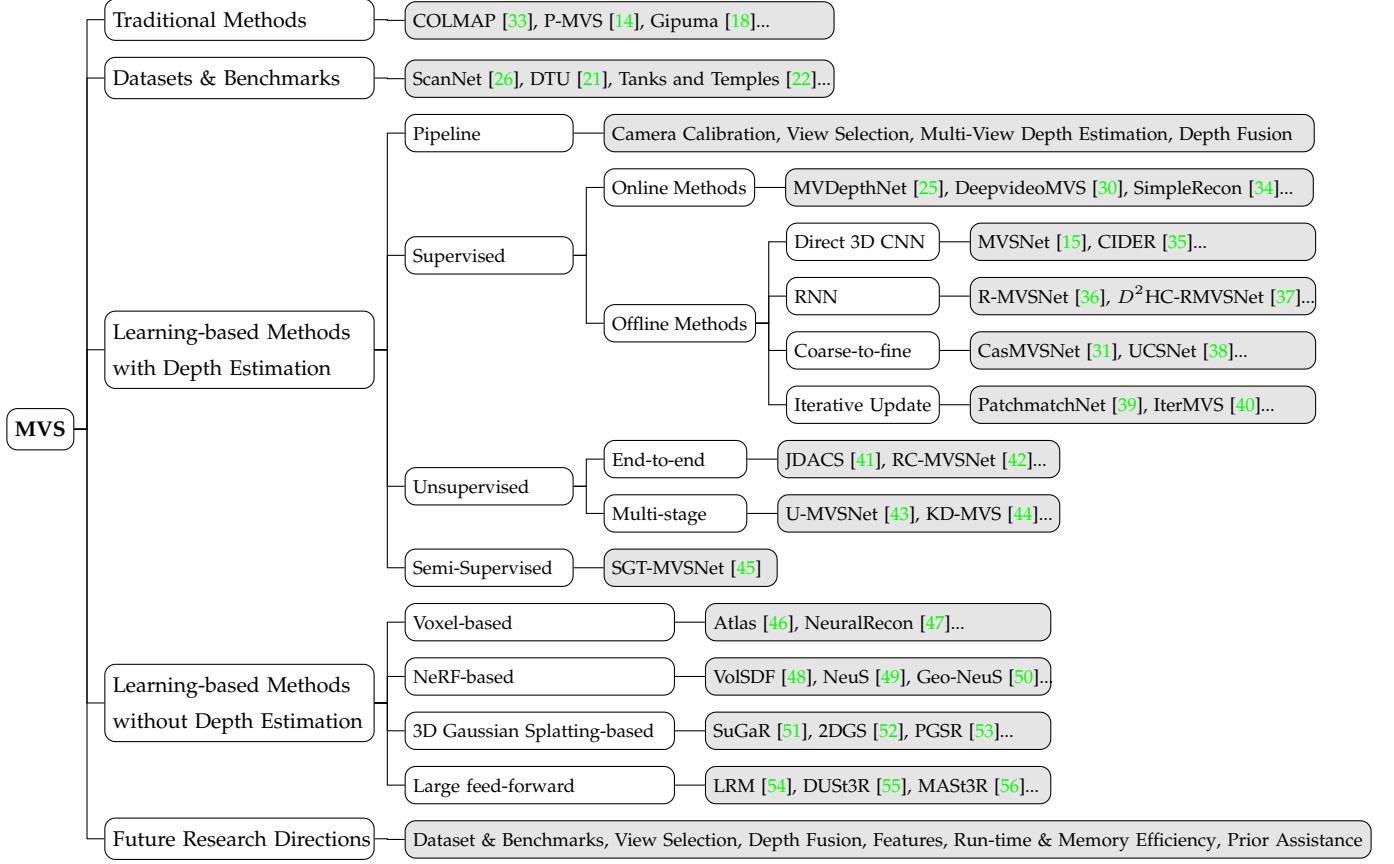


Fig. 2. Taxonomy of Multi-View Stereo.

maps [68], [69]. The common metrics are: mean absolute depth error (Abs), mean absolute relative depth error (Abs Rel) and inlier ratio with threshold 1.25 ($\delta < 1.25$). For more details, please refer to supplementary.

3D Metrics: 3D point cloud evaluation is widely used by offline MVS methods [15], [36]. Note that the reconstructed point clouds should be aligned to ground truth point clouds before evaluation, *e.g.*, by Iterative Closest Point (ICP), if their extrinsics are differently calibrated. The metrics are Precision/Accuracy, Recall/Completeness and F-score. For more details, please refer to supplementary.

3 SUPERVISED METHOD WITH DEPTH ESTIMATION

This section mainly introduces supervised learning-based MVS methods with depth estimation. A typical depth map-based MVS pipeline mainly consists of the feature extraction (Sec. 3.1), cost volume construction (Sec. 3.2), cost volume regularization (Sec. 3.3) and depth estimation (Sec. 3.5). The pipelines of MVDepthNet [25] and MVSNet [15] are illustrated in Fig. 3 and Fig. 4, which are the representative methods of online and offline MVS methods respectively.

3.1 Feature Extraction

Considering efficiency, most methods use simple CNN structures to extract deep features from images, *e.g.*, ResNet [70], U-Net [71] and FPN [72]. For online MVS methods, feature extraction networks are usually used in

line with the real-time operation goal. DeepVideoMVS [30] combines MNasNet [73], which is lightweight and has low latency, with FPN. SimpleRecon [34] proposes utilizing the first two blocks from ResNet18 [70] and an EfficientNet-v2 [74] encoder, which maintains efficiency and yields a sizeable improvement in depth map accuracy. For offline MVS methods, MVSNet [15] uses a stacked 8-layer 2D CNN to extract deep features for all the images. Coarse-to-fine methods further extract multi-scale features for estimation on multiple scales with FPN [31], [38], [39] or multi-scale RGB images [75], [76]. Recently, many following works have been paying more attention to feature extraction to improve the representation power of deep features. [77], [78], [79] introduce deformable convolutions to adaptively learn receptive fields for areas with varying richness of texture. CDS-MVSNet [80] uses a dynamic scale network, CDSFNet, to extract the discriminative features by analyzing the normal curvature of the image surface. GeoMVSNet [76] proposes a geometry fusion network that fuses the image features with coarse depth maps. With FPN as the main feature extractor, WT-MVSNet [81] and MVSFormer [82], further introduces Vision Transformers [83], [84], [85] for image feature enhancement. ET-MVSNet [86] integrates the Epipolar Transformer into the feature extraction, which performs non-local feature aggregation on the epipolar lines.

3.2 Cost Volume Construction

For both online and offline MVS methods, the cost volume is constructed with the plane sweep algorithm as discussed

in Sec. 2.3.

3.2.1 Online MVS

To reduce computation and improve the efficiency for online applications, online MVS methods usually construct 3D cost volumes $\mathbf{C} \in \mathbb{R}^{H \times W \times D}$, which store a single value as matching cost for each pixel \mathbf{p} and depth sample d_i . MVDepthNet [25] and GP-MVS [60] compute the per-pixel intensity difference between the reference and each source view as matching cost. If there is more than one source view, the cost volumes are averaged. Instead of intensity difference, Neural RGB-D [87] computes deep feature difference. DeepVideoMVS [30] and MaGNet [88] calculate the cost from pixel-wise correlation between the reference feature and the warped source feature. In addition to the dot product between reference image features and warped source image features, SimpleRecon [34] further adds metadata, *e.g.*, ray direction and relative pose, into the 4D cost volume. Then an MLP is used to reduce the dimension of 4D cost volume to a 3D cost volume. DoubleTake [89] further includes depth hint, the depth map rendered from the previously predicted geometry, into the cost volume.

3.2.2 Offline MVS

Offline MVS methods focus on reconstructing high-quality dense geometry with high-resolution images. To encode more matching information and improve the quality, offline methods [15], [31] usually construct 4D cost volumes $\mathbf{C} \in \mathbb{R}^{H \times W \times D \times C}$, where each pixel \mathbf{p} and depth sample d_i is associated with a matching cost of dimension C . Since there may exist an arbitrary number of source views and serious occlusion [23], robustly aggregating matching information from all the source views is an important step. MVSNet [15] proposes a variance-based cost metric as follows:

$$\mathbf{C} = \text{Var}(\mathbf{V}_0, \dots, \mathbf{V}_{N-1}) = \frac{\sum_{i=0}^{N-1} (\mathbf{V}_i - \bar{\mathbf{V}})^2}{N}, \quad (5)$$

where $\{\mathbf{V}_i\}_{i=0}^{N-1}$ are (warped) feature volumes, $\bar{\mathbf{V}}$ is the average feature volume. To further reduce dimension, CIDER [35] adopts group-wise correlation [90] to compute a lightweight cost volume between reference and each warped source view. Then $N - 1$ cost volumes are averaged for regularization. Without considering occlusions, these methods consider all source views equally with the averaging operation.

Nonetheless, it is essential to emphasize that *occlusions* are crucial to consider, as they pose common challenges in MVS, frequently leading to invalid matching and inaccurate estimations [33]. Incorporating visibility information to aggregate matching details from source views can substantially bolster robustness against occlusions, thereby enhancing the accuracy of the reconstruction process [33].

To estimate view weights for source views, PVA-MVSNet [91] applies gated convolution [92] to adaptively aggregate cost volumes. View aggregation tends to give occluded areas smaller weights and the reweighting map is yielded according to the volume itself. Vis-MVSNet [93] aggregates pair-wise cost volumes by weighted sum, where the weight is negatively related to the uncertainty of depth probability distribution. [39], [40], [94] utilize pixel-wise

view weight networks to learn view weights from the pair-wise cost volumes without supervision.

3.3 Cost Volume Regularization

Usually, the raw cost volume constructed from image features may be noisy and should be incorporated with smoothness constraint for depth estimation [15]. Therefore, cost volume regularization is an important step to refine the raw cost volume and aggregate matching information from a large receptive field.

3.3.1 Online MVS

A 2D encoder-decoder architecture is commonly used to aggregate information. MVDepthNet [25] and GP-MVS [60] concatenate the reference image and cost volume and send them to an encoder-decoder architecture with skip connections. In addition, GP-MVS [60] uses Gaussian process to fuse information from previous views. Neural-RGBD [87] accumulates depth probability volumes over time with a Bayesian filtering framework to effectively reduce depth uncertainty and improve robustness and temporal stability. DeepVideoMVS [30] applies 2D U-Net [71] on the cost volume and adds skip connections between the image encoder and cost volume encoder at all resolutions. It further uses ConvLSTM [95] to propagate past information with a small overhead of computation time and memory consumption. SimpleRecon [34] fuses multi-scale image features, extracted from the pretrained EfficientNetv2 [74], into the cost volume encoder to improve the performance.

3.3.2 Offline MVS

Among most learning-based offline MVS methods that use 4D cost volumes, there are three main categories to perform cost volume regularization: direct 3D CNN, coarse-to-fine and RNN.

Direct 3D CNN: Similar to stereo estimation [62], [63], [64], 3D CNN is widely used for cost volume regularization in MVS [15], [35], [96], [97], [98]. As the blueprint of learning-based offline MVS methods, MVSNet [15] adopts a 3D U-Net [71] to regularize the cost volume, which aggregates context information from a large receptive field with relatively low computation cost. It is found that the 3D regularization can capture better geometry structures, perform photometric matching in 3D space, and alleviate the influence of image distortion caused by perspective transformation and occlusions [97]. However, since 3D CNN is memory and run-time consuming, many offline methods [15], [35], [97], [98] use limited depth hypotheses and estimate depth maps at low resolution.

RNN: Instead of 3D CNN, R-MVSNet [36] sequentially regularizes 2D slices of the cost volume along the depth dimension with a convolutional GRU [99], which is able to gather spatial and uni-directional context information in the depth dimension. D^2 HC-RMVSNet [37] augments R-MVSNet [36] with a complex convolutional LSTM [95]. AA-RMVSNet [77] further introduce an intra-view feature aggregation module for feature extraction and an inter-view cost volume aggregation module to adaptively aggregate cost volumes of different views. With sequential process of

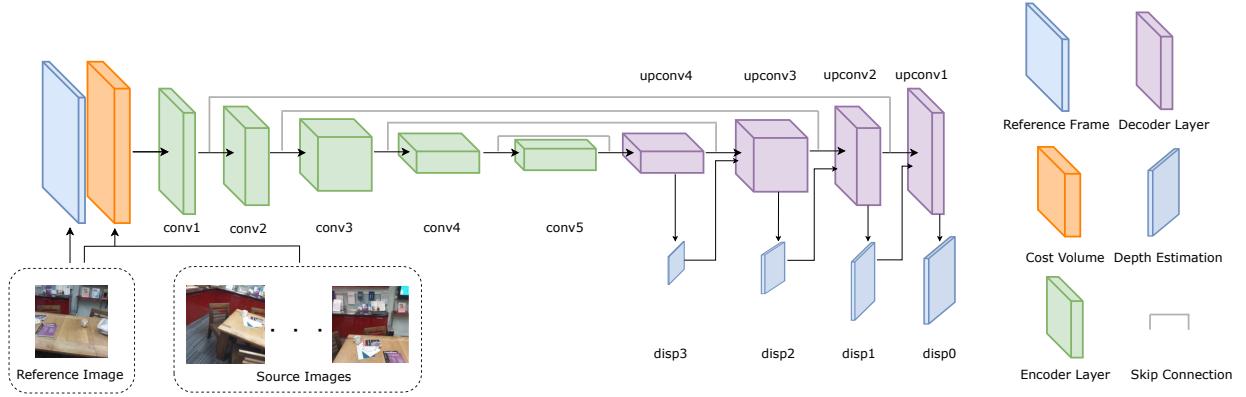


Fig. 3. Pipeline of MVDepthNet [25]. Multiple image frames are encoded in the cost volume. MVDepthNet takes the reference image and the cost volume as the input and outputs inverse depth maps of four different resolutions. Skip connections between the encoder and decoder of the same resolution are used for the per-pixel depth estimation.

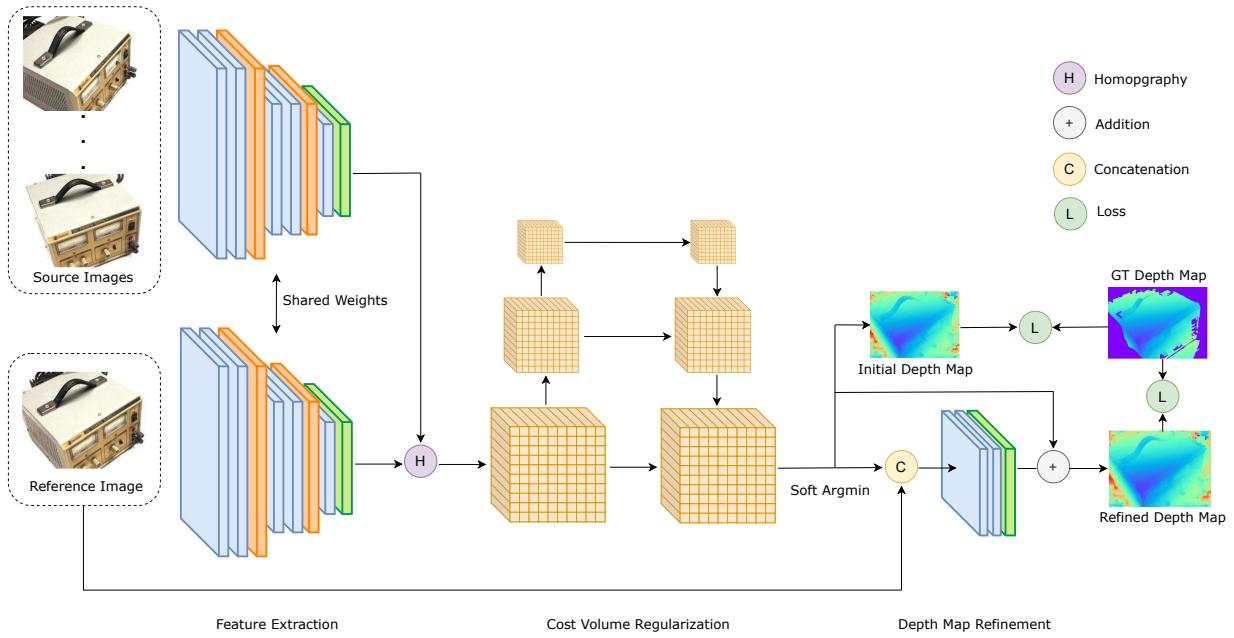


Fig. 4. Pipeline of MVSNet [15]. Reference and source images go through the feature extraction network, followed by the differentiable homograph warping to construct the cost volume. A 3D U-Net is used to regularize the cost volume into a probability volume. The final depth map is estimated from the probability volume and refined with the reference image features.

2D slices, these methods improve the scalability for high-resolution reconstruction as well as large-scale scenes and reduce memory, however, at the cost of run-time.

Coarse-to-Fine: Predicting depth in a coarse-to-fine manner [31], [38], [39], [75], [78], [94], [100] is another solution of reducing both memory consumption and running-time. A coarse depth map is first predicted and then upsampled and refined during finer stages to construct fine details. Cas-MVSNet [31] constructs cascade cost volumes by warping features with reduced depth ranges around the previous coarse maps. Based on Eq. (3), the homography of differential warping at stage $k+1$ is

$$\mathbf{H}_i^{(k+1)}(d^{(k)} + \Delta^{(k+1)}) = (d^{(k)} + \Delta^{(k+1)})\mathbf{K}_0\mathbf{T}_0\mathbf{T}_i^{-1}\mathbf{K}_i^{-1}, \quad (6)$$

where $d^{(k)}$ is the estimated depth value at stage k and $\Delta^{(k+1)}$ is the residual depth to be determined in the current stage. Following MVSNet [15], 3D CNN is used on each stage to regularize the cost volume. UCS-Net [38] estimates

uncertainty from the coarse prediction to adaptively adjust search ranges in finer stages. CVP-MVSNet [75] constructs the cost volume with a proposed optimal depth resolution of half pixel to narrow depth range in finer stages. EPP-MVSNet [101] introduces an epipolar-assembling module to assemble high-resolution information into cost volume and an entropy-based process to adjust depth range. TransMVSNet [79] uses a Feature Matching Transformer for robust long-range global context aggregation within and across images. WT-MVSNet [81] uses a window-based Epipolar Transformer for enhanced patch-to-patch matching, and a window-based Cost Transformer to better aggregate global information.

3.4 Iterative Update

Diverging from conventional approaches, certain methods [39], [40], [102], [103], [104] adopt iterative updates to gradually refine depth maps. By iteratively updating the

depth maps based on successive iterations, these methods can progressively improve the accuracy and consistency of the reconstructed 3D scene. Moreover, the ability to control the number of iterations provides users with the flexibility to prioritize either computational efficiency or reconstruction quality, depending on specific application requirements.

Some methods [39], [102] combine iterative PatchMatch [105] with deep learning. PatchMatch algorithm is widely used in many traditional MVS methods [18], [19], [33], [106]. The PatchMatch algorithm mainly consists of: random initialization, propagation of hypotheses to neighbors, and evaluation for choosing best solutions. After initialization, the approach iterates between propagation and evaluation until convergence. Traditional methods usually design fixed patterns for propagation. Recently, PatchmatchNet [39] proposes learned adaptive propagation and cost aggregation modules, which enables PatchMatch to converge faster and deliver more accurate depth maps. Instead of propagating depth samples naively from a static set of neighbors as done traditionally [18], PatchmatchNet adaptively gathers samples from pixels of the same surface with the guidance of image features. PatchMatch-RL [102], jointly estimates depth, normal and visibility with a coarse-to-fine structure. Considering *argmax* based hard decisions/sampling of PatchMatch is non-differentiable, PatMatch-RL adopts reinforcement learning in training.

Recently, RAFT [107] estimates optical flow by iteratively updating a motion field with GRU. In MVS, several methods [40], [103], [104] also adopted this approach to enhance efficiency and flexibility. IterMVS [40] proposes a lightweight GRU-based probability estimator that encodes the per-pixel probability distribution of depth in its hidden state. CER-MVS [103] and Effi-MVS [104] both embed the RAFT module in a coarse-to-fine structure, which outputs a residual that is added to the previous depth. MaGNet [88] iteratively updates the Gaussian distribution of depth for each pixel with the matching scores. DELS-MVS [108] proposes Epipolar Residual Network to search for the corresponding point in the source image directly along the corresponding epipolar line and follows an iterative manner to narrow down the search space. Based on IterMVS [40], RIAV-MVS [109] iteratively refines index fields with a 3-level GRUs. IGEV-MVS [110] iteratively updates a disparity map regressed from geometry encoding cost volumes and pairs of correlation volumes.

3.5 Depth Estimation

3.5.1 Online MVS

Many methods [25], [30], [34], [60] apply encoder-decoder on the cost volume \mathbf{C} and reduce the feature channel to 1, *i.e.*, $\mathbf{C}' \in \mathbb{R}^{H \times W \times 1}$. Then the *sigmoid* activation σ is applied for normalization. Together with the predefined depth range $[d_{\min}, d_{\max}]$ (mostly manually set), the depth map can be computed. For example, DeepVideoMVS [30] estimates depth as:

$$\hat{d}(\mathbf{p}) = \left(\left(\frac{1}{d_{\min}} - \frac{1}{d_{\max}} \right) \cdot \sigma(\mathbf{C}'(\mathbf{p})) + \frac{1}{d_{\max}} \right)^{-1}, \quad (7)$$

where \mathbf{p} is the pixel coordinate. For other methods, NeuralRGBD [87] directly models the cost volume as probability volume \mathbf{P} and adopts *soft argmax* [62] to predict

depth, Eq. (8). MaGNet [88] takes the mean of Gaussian distribution at last iteration as depth.

3.5.2 Offline MVS

For a cost volume $\mathbf{C} \in \mathbb{R}^{H \times W \times D \times C}$, a probability volume $\mathbf{P} \in \mathbb{R}^{H \times W \times D}$ is usually generated after cost volume regularization, which is then used for depth estimation. Currently, almost all the learning-based MVS methods use exclusively either regression (*soft argmax*) or classification (*argmax*) to predict depth.

Following GCNet [62], MVSNet [15] uses *soft argmax* to regress the depth map with sub-pixel precision. Specifically, the expectation value along the depth direction of probability volume \mathbf{P} is computed as the final prediction:

$$\hat{d}(\mathbf{p}) = \sum_{i=1}^D d_i \cdot \mathbf{P}(i, \mathbf{p}), \quad (8)$$

where \mathbf{p} is the pixel coordinate, d_i is i -th depth sample and $\mathbf{P}(i, \mathbf{p})$ is the predicted depth probability. For coarse-to-fine methods [31], [38], [75], *soft argmax* is applied on each stage to regress the depth maps. On the other hand, some methods [35], [39] compute the expectation value of *inverse* depth samples since this sampling is more suitable for complex and large-scale scenes [35].

In contrast, methods [36], [37], [77] that use RNN for cost volume regularization mainly adopt *argmax* operation. They choose the depth sample with the highest probability as the final prediction, which is similar to classification. Since the *argmax* operation adopted by winner-take-all cannot produce depth estimations with sub-pixel accuracy, the depth map may be refined in post-processing [36].

Recently, Wang *et al.* [40] propose a hybrid strategy to combine regression and classification. Similarly, Peng *et al.* [111] first use classification to get the optimal hypothesis and then regress the proximity for it.

DMVSNet [112] estimates two depth maps on multi-stages and composes the final depth map by alternating between selecting the maximum and minimum predicted depth values, which generates an oscillating depth map for improved geometry.

3.6 Depth Refinement

Given that the raw depth estimation from MVS may be noisy, refinement is usually used to improve the accuracy. R-MVSNet [36] enforces multi-view photo-consistency to alleviate the stair effect and achieve sub-pixel precision. Point-MVSNet [97] uses PointFlow to refine the point cloud iteratively by estimating the residual between the depth of the current iteration and that of the ground truth. Fast-MVSNet [113] adopts an efficient Gauss-Newton layer to optimize the depth map by minimizing the feature residuals. PatchmatchNet [39] refines the final upsampled depth map with a depth residual network [114] and reference image feature. [40], [103], [104] use the mask upsampling module from RAFT [107] to upsample and refine the depth map to full resolution by computing the weighted sum of depth values in a window based on the mask. Based on a coarse depth map, RayMVSNet [115], [116] aggregates multi-view image features with an epipolar transformer and use a 1D implicit field to estimate the SDF of the sampled points and

the location of the zero-crossing point. GeoMVSNet [76] filters the depth map by geometry enhancement in the frequency domain. EPNet [117] uses a hierarchical edge-preserving residual learning module to refine multi-scale depth estimation with image context features.

3.7 Confidence Estimation

As discussed in Sec. 2.4, photometric confidence is important to filter out unreliable estimations during depth fusion. Following MVSNet [15], most offline MVS methods take the probability of the estimation [36], [37], [77] or the probability sum over several samples near the estimation [15], [31], [39] from the probability volume as confidence. In stereo matching, some methods learn confidence from disparity [118], RGB image [119], [120] or matching costs [121] and obtain confidence scores in $[0, 1]$ interval. Motivated by this, some traditional MVS methods [122], [123] based on classical PatchMatch [106] propose to estimate the confidence with deep learning and use this to refine the results from PatchMatch. Recently, IterMVS [40] estimates confidence from the hidden state of a convolutional GRU. A 2D CNN followed by a *sigmoid* is applied to the hidden state to predict the confidence. DELS-MVS [108] feeds pixel-wise entropy of the partition probabilities, which is computed for each source image, into a confidence network to learn confidence. The confidence is used to guide the fusion of multiple depth maps.

3.8 Loss Function

3.8.1 Online MVS

Many methods [25], [30], [60] compute the regression loss of estimated inverse depth maps for training:

$$L = \sum_{\mathbf{p}} \left\| \frac{1}{d(\mathbf{p})} - \frac{1}{\hat{d}(\mathbf{p})} \right\|_1, \quad (9)$$

where \mathbf{p} denotes the pixel coordinate, $d(\mathbf{p})$ denotes the ground truth depth, $\hat{d}(\mathbf{p})$ denotes the estimation and $\|\cdot\|_1$ denotes the L_1 loss. For other methods, Neural-RGBD [87] uses Negative-Log Likelihood (NLL) over the depth with its depth probability volume. Similarly, MaGNet [88] uses NLL loss since the per-pixel depth is modeled as Gaussian distribution. SimpleRecon [34] computes the regression loss with log-depth. To improve performance, SimpleRecon further uses gradient loss on depth, normal loss where normal is computed with depth and intrinsics, and multi-view regression loss.

3.8.2 Offline MVS

Based on the depth estimation strategy as discussed in Sec. 3.5, loss functions can be mainly categorized into regression and classification. For methods [15], [35] that predict depth with *soft argmax* [62], (smooth) L_1 loss is usually adopted as the loss function, which is stated as:

$$L = \sum_{\mathbf{p}} \|d(\mathbf{p}) - \hat{d}(\mathbf{p})\|_1. \quad (10)$$

For coarse-to-fine methods [31], [38], [39], [75], the L_1 loss is computed on each stage for multi-stage supervision.

For methods [36], [37], [77] that predict depth with an *argmax* operation, cross entropy loss is commonly used for the loss function since the problem is multi-class classification. The cross entropy loss function is defined as:

$$L = \sum_{\mathbf{p}} \left(\sum_{i=1}^D -G(i, \mathbf{p}) \cdot \log[P(i, \mathbf{p})] \right), \quad (11)$$

where $G(i, \mathbf{p})$ is the ground truth one-hot vector of depth at pixel \mathbf{p} , and $P(i, \mathbf{p})$ is the predicted depth probability.

For methods that predict depth with a hybrid strategy of classification and regression, Wang *et al.* [40] adopt both L_1 loss and cross entropy loss to supervise the regression and classification respectively, while Peng *et al.* [111] use focal loss [124].

4 UNSUPERVISED & SEMI-SUPERVISED METHODS WITH DEPTH ESTIMATION

In this section, we introduce unsupervised learning-based MVS methods [41], [42], [43], [125], [126] and semi-supervised method [45]. Supervised MVS methods mentioned in Sec. 3 depend extensively on the availability of accurate ground truth depth maps obtained through depth-sensing equipment. To make MVS practical in more general real-world scenarios, it is vital to consider alternative unsupervised learning-based methods that can provide competitive accuracy compared to the supervised ones without any ground truth depth. Existing unsupervised methods are built upon the assumption of photometric consistency (Sec. 4.1), and are categorized into end-to-end (Sec. 4.2) and multi-stage (Sec. 4.3). SGT-MVSNet [45] is the only semi-supervised method so far, and we introduce it in Sec. 4.4.

4.1 Photometric Consistency Assumption

In the realm of unsupervised depth map prediction, extant methods [43], [127], [128], [129] endeavor to establish photometric consistency between reference and source perspectives. This pivotal notion revolves around the augmentation of similarity between the reference image \mathbf{I}_0 and individual source images \mathbf{I}_i after their warping to align with the reference view.

In relation to the depth estimation denoted as \hat{d} for the initial image \mathbf{I}_0 , the process involves the projection of reference pixels into the subsequent image \mathbf{I}_i using the formulation presented in Eq. (4).

Subsequently, the distorted version of the source image denoted as $\hat{\mathbf{I}}_0^i$ is created by interpolating the RGB values of the source image at the displaced pixel positions through bilinear sampling. This interpolation is carried out at the locations where the pixels have been transformed due to the warping process. Additionally, alongside the warped image $\hat{\mathbf{I}}_0^i$, a binary mask M_i is commonly generated. This mask is employed to exclude pixels that have been projected beyond the boundaries of the image and are, thus, considered invalid.

The photometric consistency loss L_{PC} can be written as:

$$L_{PC} = \sum_{i=1}^{N-1} \frac{1}{\|M_i\|_1} \left(\left\| (\hat{\mathbf{I}}_0^i - \mathbf{I}_0) \odot M_i \right\|_2 + \left\| (\nabla \hat{\mathbf{I}}_0^i - \nabla \mathbf{I}_0) \odot M_i \right\|_2 \right), \quad (12)$$

where ∇ denotes the gradient at the pixel level, while \odot symbolizes element-wise Hadamard multiplication. In most cases [41], [42], [43], [127], [128], the incorporation of structural similarity loss and depth smoothness loss into the computation is commonplace. This practice aims to enhance the training process's stability and speed up the convergence.

The computation of the structural similarity loss occurs between a synthesized image and the initial reference image, aiming to uphold contextual congruity. More precisely, the assessment of contextual similarity often involves the Structural Similarity Index (SSIM) proposed [130], defined as:

$$L_{\text{SSIM}} = \sum_{i=1}^{N-1} \left[1 - \text{SSIM}(\mathbf{I}_0, \hat{\mathbf{I}}_0^i) \right] \odot M_i, \quad (13)$$

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \quad (14)$$

where μ, σ^2 represent the mean and variance of the images, c_1, c_2 are constants to avoid numerical issues.

The incorporation of a smoothness loss term serves to promote the continuity of depth information within the context of image and depth disparity alignment. This continuity is evaluated based on the color intensity gradient present in the input reference image. The computation of the smoothness loss, L_{SM} , is defined as follows:

$$L_{\text{SM}} = \sum_{\mathbf{x}} \left| \nabla_u \tilde{d}(\mathbf{x}) \right| e^{-|\nabla_u \mathbf{I}_0(\mathbf{x})|} + \left| \nabla_v \tilde{d}(\mathbf{x}) \right| e^{-|\nabla_v \mathbf{I}_0(\mathbf{x})|}, \quad (15)$$

where ∇_u and ∇_v refer to the gradient along x and y axis, $\tilde{d} = d/\bar{d}$ is the mean-normalized inverse depth, and M represents the set of valid pixels in the reference image.

4.2 End-to-end Unsupervised Methods

End-to-end methods [41], [42], [127], [131], [132] train from scratch with the same input as supervised methods (Sec. 3) but without ground truth depth. They are based on photometric consistency, structural similarity, and smoothness constraints for loss terms.

Unsup_MVS [127] adopts view synthesis supervision and dynamically selects the X best (lowest loss) values out of Y loss maps. However, photometric correspondences can be inaccurate due to non-Lambertian surfaces, camera exposure variations, and occlusions, leading to ambiguous supervision (Sec. 4.1). JDACS [41] introduces semantic consistency to address these challenges, using a pre-trained network to generate semantic maps and enforcing cross-view segmentation consistency. However, this approach can struggle to converge and lacks detailed information. RC-MVSNet [42] enhances supervision through neural rendering by combining Neural Radiance Fields and cost volumes. It uses a CasMVSNet backbone to generate depth maps supervised by photometric consistency, along with rendering-consistency and view synthesis losses to handle occlusions and varying lighting. ElasticMVS [131] addresses issues in photometric loss-based geometry by introducing a part-aware patch-match framework, using an elastic part representation to guide the depth map prediction. The network is

optimized with contrastive and spatial concentration losses to promote pixel isolation. CL-MVSNet [132] improves proximity between positive pairs through contrastive consistency between a regular CasMVSNet branch and two contrastive branches. It also introduces $L0.5$ photometric consistency to improve the precision of the reconstruction.

These end-to-end methods train from scratch without pre-processing, reducing training time and complexity in real-world applications.

4.3 Multi-stage Unsupervised Methods

Multi-stage methods require pre-training or data pre-processing and are based on pseudo-label generation.

Self-supervised CVP-MVSNet [128] generates pseudo ground truth depth using photometric consistency, refining depth maps with cross-view consistency checks and point cloud fusion [15]. The meshes are then reconstructed for further training iterations. U-MVS [43] pre-trains an optical-flow network, PWC-Net [133], and uses flow-depth consistency to generate pseudo labels, reducing supervision ambiguity in foregrounds and backgrounds with an uncertainty-aware self-training consistency. KD-MVS [44] employs knowledge distillation to increase performance. A teacher MVS model generates pseudo ground-truth labels through photometric and feature-metric consistency loss, with uncertainty encoding. These labels are then used to train the student models.

4.4 Semi-supervised Methods

SGT-MVSNet [45] uses sparse 3D points to estimate depth maps. A 3D point consistency loss minimizes differences between back-projected 3D points and ground truth. A coarse-to-fine depth propagation module improves accuracy at edges and boundaries.

5 LEARNING-BASED MVS WITHOUT DEPTH ESTIMATION

Though the learning-based methods that predict individual depth maps with plane-sweep are the main family of learning-based multi-view stereo, there are many methods of other families that achieve impressive 3D reconstruction quality in the recent years. In this section, We discuss four main families: voxel-based, NeRF-based, 3D Gaussian Splatting-based and large feed-forward methods.

5.1 Voxel-based Methods

These methods [46], [47], [134], [135] estimate the scene geometry with volumetric representation by leveraging implicit function, *e.g.*, SDF. Specifically, Atlas [46] and NeuralRecon [47] attempt to predict the TSDF volume from the 3D feature volume constructed by lifting 2D image features. Atlas uses 3D CNN to regress the TSDF volume based on the feature volume accumulated from all images of the scene, which exhibits great completeness of reconstruction. The TSDF reconstruction is supervised using L_1 loss to the ground truth TSDF values. Since dense feature volume brings lots of computational overhead, NeuralRecon further improves the efficiency by incrementally reconstructing the

scene in a fragment-wise and coarse-to-fine manner. The 3D features from different fragments are passed through the whole incremental reconstruction process with a RNN. TransformerFusion [134] fuses coarse and fine image features in a voxel grid with two transformers and then predicts an occupancy field to represent the scene geometry. VoRTX [135] uses a similar design to TransformerFusion and the scene geometry is obtained by passing three different-level features output from transformers through a 3D CNN.

5.2 NeRF-based Methods

In novel view synthesis, Neural Radiance Field (NeRF) [27] has kicked off a new emerging representation of 3D, which offers a differentiable volume-rendering scheme to supervise a 3D radiance-based representation with 2D image-level losses. NeRF employs multi-layer perceptron (MLP) to map a position (x, y, z) and the normalized view direction (θ, ϕ) to the corresponding color \mathbf{c} and volume density σ . For a specific ray at a novel viewpoint, NeRF uses approximated numerical volume rendering to compute the accumulated color as:

$$\mathbf{C} = \sum_{i=1}^N T_i (1 - \exp(-\sigma_i \delta_i)) \mathbf{c}_i, \quad (16)$$

where i is the index of sample, $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ is the accumulated transmittance, and $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples. The model is trained by minimizing the loss between the predicted and ground truth color:

$$\mathcal{L}_{\text{color}} = \mathbb{E}[\|\mathbf{C} - \mathbf{C}_{gt}\|^2]. \quad (17)$$

Many subsequent endeavors [136], [137], [138], [139], [140], [141], [142], [143], [144] further improve NeRF in quality, fast training, memory efficiency and real-time rendering.

Though the initial purpose of NeRF is to perform novel view synthesis, VolSDF [48] and NeuS [49] integrate NeRF with SDF for surface reconstruction. The SDF, denoted as f , is transformed into the density σ for volume rendering. Specifically, for a point $\mathbf{p}(t)$, VolSDF computes the volume density $\sigma(\mathbf{p}(t))$ from the signed distance $f(\mathbf{p}(t))$ as:

$$\sigma(\mathbf{p}(t)) = \frac{1}{\beta} \Psi_\beta(-f(\mathbf{p}(t))), \quad (18)$$

where Ψ_β denotes the Cumulative Distribution Function (CDF) of a zero-mean Laplace distribution with learnable scale parameter $\beta > 0$. NeuS computes the density in a different way as:

$$\sigma(\mathbf{p}(t)) = \max\left(\frac{-f'(\mathbf{p}(t))\Phi'_s(f(\mathbf{p}(t)))}{\Phi_s(f(\mathbf{p}(t)))}, 0\right), \quad (19)$$

where Φ_s is the sigmoid function with learnable scale parameter s . After training, the mesh can be extracted from the SDF field with Marching Cubes [67].

Note that without ground truth geometry supervision, e.g., depth map or TSDF volume, these methods are trained in a self-supervised way as NeRF, Eq. (17). Since training with rendering loss only like NeRF has ambiguity in geometry [145], some following methods improve the reconstruction with explicit geometry supervision, e.g., monocular depth/normal priors [146], [147] and sparse SfM point

cloud [50]. Motivated by Instant-NGP [138] that accelerates training with hash grids, other methods [148], [149], [150], [151] use hash grids to speed up training and improve surface details. In addition, to overcome the drawback that NeRF typically needs to be trained for each new scene, some recent methods [152], [153], [154] propose generalizable pipelines for implicit reconstruction even under sparse-view settings. Specifically, these methods project 3D points on the image planes and aggregate the corresponding image features as a feature volume like Atlas [46], which is used to estimate the surface location.

Recall that the depth map-based MVS methods predict depth with photometric consistency across multiple views. However, this assumption fails for glossy surfaces with reflections and thus they cannot reconstruct them accurately. In contrast, some NeRF methods can handle reflections well. Recently, Ref-NeRF [155] reparameterizes the appearance with separate diffuse and reflective components by using the reflected view direction, which improves the rendering of specular surfaces. Therefore, recent methods [151], [156], [157], [158], [159] adopt this representation in reconstruction and can successfully reconstruct specular surfaces. Specifically, based on VolSDF/NeuS, these methods mainly replace the view direction with the reflected view direction following Ref-NeRF.

5.3 3D Gaussian Splatting-based Methods

Unlike implicit representations with a coordinate-based MLP such as NeRF [27], 3D Gaussian Splatting [160] (3DGS) explicitly represents the scene with point primitives, each of which is parameterized as a scaled Gaussian with 3D covariance matrix Σ and mean μ :

$$G(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu)}, \quad (20)$$

where \mathbf{x} is an arbitrary position. Σ is formulated with a scaling matrix \mathbf{S} and rotation matrix \mathbf{R} as:

$$\Sigma = \mathbf{R} \mathbf{S} \mathbf{S}^T \mathbf{R}^T. \quad (21)$$

In addition, each Gaussian contains the color \mathbf{c} modeled by Spherical Harmonics and an opacity α . Different from NeRF that uses volume rendering, 3DGS efficiently renders the scene via tile-based rasterization. After projecting 3D Gaussian $G(\mathbf{x})$ into the 2D Gaussian $G'(x)$ on the image plane [160], a tile-based rasterizer efficiently sorts the 2D Gaussians and employs α -blending for rendering:

$$\mathbf{C}(x) = \sum_{i \in N} \mathbf{c}_i \sigma_i \prod_{j=1}^{i-1} (1 - \sigma_j), \quad \sigma_i = \alpha_i G'(x), \quad (22)$$

where x is the pixel location, N is the number of sorted 2D Gaussians. During training, 3DGS minimizes the rendering loss like NeRF, as in Eq. (17). 3DGS can be initialized with SfM [17], [161] or MVS [33], which performs better than random initialization [160].

Motivated by the NeRF-based MVS methods, researchers try to adapt 3DGS for reconstruction. Though 3DGS achieves high-quality novel-view synthesis, it is challenging to recover high quality geometry since no explicit geometry constraint is used and 3D Gaussians do not correspond well to the actual surface because of the 3D

covariance [51]. SuGaR [51] introduces a geometry regularization term encouraging the 3D Gaussians to be well-aligned over the scene surfaces so that the Gaussians can contribute to better scene geometry. To reconstruct the mesh, SuGaR samples 3D points on a level set of the density computed from 3D Gaussians and then runs Poisson Reconstruction [162] on these points. NeuSG [163] jointly trains Neuralangelo [149], a NeRF-based method, and 3DGs. During training, the normals from the SDF field of Neuralangelo regularize the rotation of 3D Gaussians, while the SDF field is regularized to ensure that the SDF values at 3D Gaussians' positions are close to zero. 2DGs [52] further improves surface accuracy by modeling the 3D primitives as surfels (tangent plane clipped) and fuses rendered depth maps into mesh with TSDF fusion. PGSR [53] elicits unbiased depth maps as strong geometry prior from rendered distances and rendered normals, and enforces multi-view geometric consistency by regularizing such a prior with local plane assumption and edge constraint. RaDe-GS [164] also improves the quality of geometry by precisely modeling the depth and normal, instead of the error-prone approximate projection in the original 3DGs. However, retrieving good geometry from Gaussian-based representations still needs to be explored.

5.4 Large Feed-forward Methods

In 3D reconstruction and 3D generation, a recent prominent trend is to directly learn the 3D representation from large-scale 3D datasets, *e.g.* Objaverse [29]. These methods typically adopt large-scale transformers [165] to generate a 3D representation based on various inputs, including single image [54], [166], [167], textual description [168], [169], posed multi-view images [168], [170], [171], [172], [173], and un-posed multi-view images [55], [174]. LRM [54] uses a transformer-based feed-forward network to regress the 3D object from a single image. Specifically, it predicts features of a tri-plane representation [175], which is subsequently converted into a radiance field similar to NeRF. Large feed-forward models can also be combined with 3D Gaussian Splatting, where they directly generate 3D Gaussian points from posed multi-view inputs [168], [170], [171], [172], [173], [176]. Instead of predicting 3D Gaussian locations directly, which is a highly ill-posed task, the common practice is to use per-pixel aligned Plücker ray embeddings and predict the depth for each pixel [176] of input views. As such, this design requires the input views to cover a large range of the object. Recently, DUST3R [55] predicts pixel-aligned point coordinates directly from uncalibrated image pairs. After extracting image token representations with Vision Transformer [177], two transformer decoders exchange information of the token representations via cross-attention and then regress the per-pixel 3D points in the coordinate frame of the first image. Many downstream tasks can be performed with the point clouds, *e.g.*, point matching, localization, intrinsic and relative pose estimation. Directly extended from DUST3R, MAST3R [56] focuses on the specific application of feature description and matching across multi-view images. It leverages an additional matching head on top of DUST3R's architecture to boost the performance of feature-based matching. With the dense correspondences established, MAST3R manages to achieve a comparable reconstruction quality as MVSNet [15]. Spann3R [178] manages

an external spatial memory to keep track of all previous relevant 3D information yielded by DUST3R and then queries this spatial memory to predict the 3D structure.

6 DISCUSSIONS

In this section, we summarize and discuss the performance of learning-based MVS methods, including supervised online methods with depth estimation (Sec. 6.1), supervised offline methods with depth estimation (Sec. 6.2), unsupervised methods with depth estimation (Sec. 6.3) and methods without depth estimation (Sec. 6.4). Moreover, we discuss potential directions for future research (Sec. 6.5).

6.1 Supervised Online MVS with Depth Estimation

Typically, online MVS methods are trained on ScanNet [26] and then evaluated on ScanNet [26] and 7-Scenes [179]. We summarize the quantitative results of online MVS methods in Tab. 1. By incorporating temporal information in depth estimation, [30], [60], [188] achieve better performance than MVDepthNet [25]. However, this usually increase the network complexity. Instead of temporal fusion, SimpleRecon [34] injects cheaply available metadata into the cost volume and achieves state-of-the-art performance on both ScanNet and 7-Scenes.

6.2 Supervised Offline MVS with Depth Estimation

Offline MVS methods are usually trained on DTU [21] and then evaluated on DTU [21], Tanks and Temples [22] and ETH3D [23]. Recently, many methods further finetune the DTU-pretrained model on BlendedMVS [24] before evaluating on Tanks and Temples and ETH3D since it contains large-scale scenes.

6.2.1 Benchmark Performance

Tab. 3 summarizes quantitative results of offline MVS methods. Compared with those methods that use direct 3D CNN for regularization, the methods that use RNN and coarse-to-fine regularization perform much better. Since RNN based methods have bad time efficiency, we can find that coarse-to-fine methods become main stream of the community because of their impressive performance and high efficiency in both memory and run-time. In addition, iterative methods based on traditional PatchMatch [39], [102] or RAFT [40], [103], [104] also become popular since they achieve comparable performance as state-of-the-art methods with more lightweight structure. For benchmarks, most learning-based methods perform explicitly better than traditional methods on DTU and Tanks and Temples. However, on ETH3D, traditional methods [123], [189] perform better than all existing learning-based methods. We conjecture that the strong viewpoint variations and large textureless regions in human-made environments still make learning-based methods struggle. Therefore, it is valuable to pay more attention to ETH3D to evaluate the robustness of learning-based methods in real-world scenes.

TABLE 1
Quantitative results of supervised online MVS methods on ScanNet [26] and 7-Scenes [179].

Methods	ScanNet [26]			7-Scenes [179]		
	abs ↓	abs-rel ↓	$\eta < 1.25 \uparrow$	abs ↓	abs-rel ↓	$\eta < 1.25 \uparrow$
MVDepthNet [25]	0.167	0.087	0.925	0.201	0.117	0.877
DPSNet [96]	0.219	0.119	0.868	0.249	0.149	0.826
Neural-RGBD [87]	0.236	0.122	0.850	0.214	0.131	0.865
GP-MVS [60]	0.149	0.076	0.940	0.174	0.100	0.903
DeepVideoMVS [30]	0.119	0.060	0.965	0.145	0.038	0.938
MaGNet [88]	0.147	0.081	0.930	0.213	0.126	0.855
RIAV-MVS [109]	0.139	0.075	0.938	0.178	0.100	0.897
SimpleRecon [34]	0.089	0.043	0.981	0.105	0.058	0.974
DoubleTake [89]	0.077	0.037	0.984	0.099	0.053	0.970

TABLE 2
Quantitative results of point cloud evaluation on MVS benchmarks [21], [22], [23] for unsupervised depth map-based MVS methods. “Finetuned with BlendedMVS” denotes whether the methods are finetuned on BlendedMVS [24] before evaluating on Tanks and Temples [22].

Methods	DTU [21]			Finetuned with BlendedMVS [24]	Tanks and Temples [22]	
	Acc. ↓	Comp. ↓	Overall ↓		Intermediate $F_1 \uparrow$	Advanced $F_1 \uparrow$
End-to-end	Unsup_MVSNet [127]	0.881	1.073	0.977	✗	-
	MVS ² [125]	0.760	0.515	0.637	✗	37.21
	M ³ VSN [126]	0.636	0.531	0.583	✗	37.67
	JDACS-MS [41]	0.398	0.318	0.358	✗	45.48
	RC-MVSNet [42]	0.396	0.295	0.345	✗	55.08
	ElasticMVS [131]	0.374	0.325	0.349	✗	57.88
	CL-MVSNet [132]	0.375	0.283	0.329	✗	59.39
Multi-stage	Self-supervised CVP-MVSNet [128]	0.308	0.418	0.363	✗	43.48
	U-MVS [43]	0.354	0.354	0.354	✗	57.15
	KD-MVS [44]	0.359	0.295	0.327	✓	64.14
						30.82
						37.81
						37.03
						-
						30.97
						37.96

6.2.2 Memory Consumption and Run-time

Low run-time and memory consumption are crucial in most industrial applications with limited computational power and storage, *e.g.*, robotics and AR/VR. Because of the low-resolution input and the simplicity of network structures, online MVS methods [25], [30], [34] usually achieve high efficiency in both memory and run-time. In contrast, offline MVS methods focus on high-resolution images and usually use computationally expensive network modules, *e.g.*, 3D convolutions, to improve reconstruction quality. These explicitly increase run-time and GPU memory. Recently, many researchers tried to improve the efficiency while maintaining the reconstruction accuracy as other offline MVS methods. We compare the efficiency of state-of-the-art offline MVS methods [31], [39], [40], [76], [79], [86], [100], [104], [111] in Fig. 5. All these methods share similar coarse-to-fine structures to improve efficiency. However, we observe that some methods [31], [76], [79], [111] still require lots of computation resource and are not efficient in either memory or run-time. To further reduce run-time and memory, [39], [40], [100], [104] carefully simplify the architecture, *e.g.*, replacing costly 3D convolution with 2D convolution. Moreover, these methods still achieve competitive performance when compared with the top-performing methods. Therefore, it is promising to further improve both the performance and efficiency for practical applications.

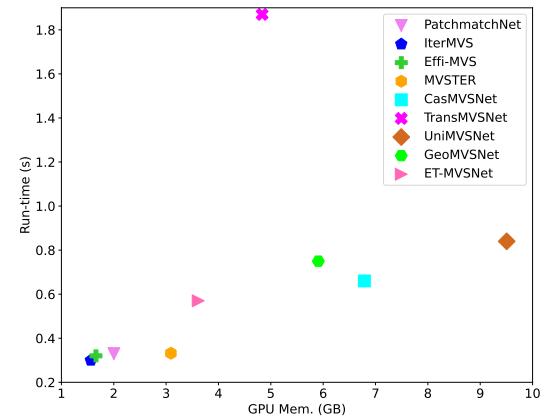


Fig. 5. GPU memory consumption and run-time of state-of-the-art offline methods [31], [39], [40], [76], [79], [86], [100], [104], [111] on DTU dataset [21]. The image resolution and the number of images are set to 1600×1152 and 5 respectively. For fair comparison, experiments are done on one workstation with a NVIDIA 2080 Ti GPU.

6.3 Unsupervised MVS with Depth Estimation

We summarize the results of unsupervised MVS methods in Tab. 2. Based on the conventional photometric consistency loss (Sec. 4.1), unsupervised MVS methods may have difficulties with many challenging situations, *e.g.*, varying lighting conditions or occlusions, and suffer from downgraded

TABLE 3

Quantitative results of point cloud evaluation on MVS benchmarks [21], [22], [23] for supervised offline MVS methods. “Finetuned with BlendedMVS” denotes whether methods are finetuned on BlendedMVS dataset [24] before evaluation on Tanks and Temples [22] and ETH3D [23].

Methods		DTU [21]			Finetuned with BlendedMVS [24]	Tanks and Temples [22]		ETH3D [23]	
		Acc. ↓	Comp. ↓	Overall ↓		Intermediate $F_1 \uparrow$	Advanced $F_1 \uparrow$	Training $F_1 \uparrow$	Test $F_1 \uparrow$
Direct 3D CNN	MVSNet [15]	0.396	0.527	0.462	✗	43.48	-	-	-
	P-MVSNet [180]	0.406	0.434	0.420	✗	55.62	-	-	-
	CIDER [35]	0.417	0.437	0.427	✗	46.76	23.12	-	-
	PointMVSNet [97]	0.342	0.411	0.376	-	-	-	-	-
	PVA-MVSNet [91]	0.379	0.336	0.357	✗	54.46	-	-	-
	Fast-MVSNet [113]	0.336	0.403	0.370	✗	47.39	-	-	-
RNN	R-MVSNet [36]	0.383	0.452	0.417	✗	48.40	24.91	-	-
	D^2 HC-RMVSNet [37]	0.395	0.378	0.386	✗	59.20	-	-	-
	AA-RMVSNet [77]	0.376	0.339	0.357	✓	61.51	-	-	-
	BH-RMVSNet [181]	0.368	0.303	0.335	✓	61.96	34.81	-	79.61
Coarse-to-fine	CasMVSNet [31]	0.325	0.385	0.355	✗	56.84	-	-	-
	CVP-MVSNet [75]	0.296	0.406	0.351	✗	54.03	-	-	-
	UCS-Net [38]	0.338	0.349	0.344	✗	54.83	-	-	-
	AttMVS [182]	0.383	0.329	0.356	✗	60.05	37.34	-	-
	Vis-MVSNet [93]	0.369	0.361	0.365	✓	60.03	-	-	-
	EPP-MVSNet [101]	0.413	0.296	0.355	✓	61.68	35.72	74.00	83.40
	CDS-MVSNet [80]	0.351	0.278	0.315	✓	61.58	-	-	-
	TransMVSNet [79]	0.321	0.289	0.305	✓	63.52	37.00	-	-
	GBi-Net [78]	0.315	0.262	0.289	✓	61.42	37.32	-	-
	UniMVSNet [111]	0.352	0.278	0.315	✓	64.36	38.96	-	-
	NP-CVP-MVSNet [183]	0.356	0.275	0.315	✓	59.64	-	-	-
	MVSTER [100]	0.340	0.266	0.303	✓	60.92	37.53	72.06	79.01
	PVSNNet [94]	0.337	0.315	0.326	✓	59.11	35.51	76.57	82.62
	MVSFormer [82]	0.327	0.251	0.289	✓	66.37	40.87	-	-
	IS-MVSNet [184]	0.351	0.359	0.355	✓	62.82	34.87	73.33	83.15
	HR-MVSNet [185]	0.332	0.310	0.321	✓	63.12	34.27	-	-
	EPNet [117]	0.299	0.323	0.313	✓	63.68	40.52	79.08	83.72
	GeoMVSNet [76]	0.331	0.259	0.295	✓	65.89	41.52	-	-
	RA-MVSNet [186]	0.326	0.268	0.297	✓	65.72	39.93	-	-
	DMVSNNet [112]	0.338	0.272	0.305	✓	64.66	41.17	-	-
	ET-MVSNet [86]	0.329	0.253	0.291	✓	65.49	40.41	-	-
	GoMVS [187]	0.347	0.227	0.287	✓	66.64	43.07	79.16	85.91
Iterative update	PatchmatchNet [39]	0.427	0.277	0.352	✗	53.15	32.31	64.21	73.12
	PatchMatch-RL [102]	-	-	-	✓	51.80	31.80	67.80	72.40
	IterMVS [40]	0.373	0.354	0.363	✓	56.94	34.17	71.69	80.09
	Effi-MVS [104]	0.321	0.313	0.317	✗	56.88	34.39	-	-
	CER-MVS [103]	0.359	0.305	0.332	✓	64.82	40.19	-	-
	IGEV-MVS [110]	0.331	0.326	0.324	-	-	-	-	-

reconstruction results. Therefore, many methods propose to employ different strategies to improve the robustness, *e.g.*, segmentation consistency [41], optical flow consistency [43], neural rendering consistency [42], pseudo-label generation with geometric filtering [44], [128], multi-iteration training [44], [128] and featuremetric loss [44]. It is inspiring that KD-MVS [44] already outperforms its backbone, CasMVSNet [31], and many other fully supervised methods. However, since unsupervised methods mainly test on relatively simple scenes [21], [22], their scalability to more complex and large-scale scenes, *e.g.*, ETH3D [23], remains a question. This good generalization capability across various scenes and datasets is crucial for practical applications. Therefore, evaluating unsupervised MVS methods on various scenes is an important task for future research. Most unsupervised methods use simple MVS backbones [15], [31], [75] and mainly focus on the training strategy. Employing state-of-

the-art architectures is also a potential direction that can further boost the performance.

6.4 Learning-based MVS without Depth Estimation

6.4.1 Voxel-based Methods

We summarize the quantitative results on ScanNet [26] in Tab. 4. Since the voxel representation increases the computation overhead explicitly when the scale of scene increases, voxel-based methods mainly focus on indoor scenes, *e.g.*, ScanNet [26]. We find that depth map-based MVS methods, *e.g.*, SimpleRecon [34], fuse depth maps with traditional TSDF fusion [65] and can achieve comparable performance as these voxel-based methods with lower complexity. This further shows the advantages of depth map-based methods over voxel-based methods.

TABLE 4
Quantitative results of voxel-based MVS methods on ScanNet [26].

Methods	Prec. \uparrow	Recall \uparrow	F-score \uparrow
Atlas [46]	0.675	0.605	0.636
NeuralRecon [47]	0.630	0.612	0.619
TransformerFusion [134]	0.728	0.600	0.655
VoRTX [135]	0.767	0.651	0.703

6.4.2 NeRF-based Methods

The quantitative results of nerf-based methods on DTU dataset [21] are summarized in Tab. 5. The most outstanding features of NeuS [49] and VolSDF [48] are that they are self-supervised, *i.e.*, without depth supervision, and produce smooth and complete mesh surfaces. However, many recent methods [50], [146], [147] find that incorporating more explicit geometric supervision explicitly improves the reconstruction quality. Compared with depth map-based MVS methods [15], [25], the main drawback of NeRF-based methods is that they are mostly per-scene optimization problems and need lots of time to train the model on each new scene. Recently, some methods [152], [153], [154] try to make the pipeline generalizable so that there is no need to train the model for new scenes. However, they perform worse than depth map-based methods [15]. Another way to improve training time efficiency is to employ hash grids [138], which also improves surface details. However, these methods [148], [149], [150], [151] need lots of GPU memory and space to train and store the large model. In addition, the scalability of NeRF-based methods remains a problem when attempting to apply on large-scale scenes [23].

6.4.3 3D Gaussian Splatting-based Methods

Because of its efficient rasterization, 3D Gaussian Splatting [160] becomes popular in novel view synthesis in the last half year. However, there are many challenges to use reconstruct surfaces with 3DGS. For example, the 3D Gaussians do not correspond well to the actual surface of the scene since the 3D Gaussian has a 3D covariance matrix [51]. Moreover, the surface may contain noisy undulations [51]. However, it is worthwhile to further explore in this direction since 3DGS is much faster to train than those NeRF-based MVS methods. Moreover, instead of directly extracting the surface from 3D space [51], rendering accurate depth maps and then fuse them like depth map-based MVS methods is a promising direction [52], [53]. Additionally, generalizable feed-forward 3DGS methods [194], [195] are interesting since they are proven to improve depth accuracy with rendering loss and do not be trained for each scene.

6.4.4 Large Feed-forward Methods

As a new trend of 3D reconstruction, large feed-forward methods demonstrate impressive performance in challenging settings that other MVS methods cannot handle, *e.g.*, single image input and un-posed sparse images. Employing a large model that is able to get scaled up, these methods manage to learn strong priors of multi-view geometry from large datasets [29]. Since these large feed-forward methods [54], [55], [174] usually do not enforce geometric constraint explicitly, *i.e.*, epipolar geometry, the reconstructed

geometry is by default placed in a local coordinate system without absolute scales, making it not feasible for applications requiring precise measurement. Also, these methods reconstruct the target under a globally shared representation, while depth-based MVS is typically performed without global data dependency, making conventional MVS still more suitable for large-scale reconstruction and parallel processing. The massive and costly computation of large feed-forward methods also limit the feasibility of applications.

Benefited from the deep prior rooted in the large datasets, large feed-forwards methods are able to handle some of the typical corner cases, *e.g.*, non-Lambertian surfaces. However it also makes their generalization ability questionable since they learn priors of multi-view geometry completely from the training dataset, while the depth map-based methods utilize feature matching via plane sweep algorithm from traditional methods that are more explainable.

6.5 Future Research Directions

6.5.1 Datasets and Benchmarks

For learning-based MVS methods, ScanNet [26] and DTU [21] are two main training datasets, while ScanNet [26], DTU [21], Tanks & Temples [22] and ETH3D [23] are the main evaluation benchmarks.

For training, the scene scale of ScanNet and DTU is relatively small (room-scale for ScanNet and object-scale for DTU) and their quality is not satisfactory. For example, the camera calibration of ScanNet is not very accurate. For DTU, the ground truth depth maps are rendered from mesh, which contain some outliers and holes. This is because the mesh is reconstructed from the sparse ground truth point cloud with Screened Poisson surface reconstruction [196] and thus contains incomplete regions [182]. Therefore, improving the scalability and quality of training datasets is an important research direction.

Recently, there are many researchers trying to solve this problem. TartanAir dataset [197] is a large-scale synthetic dataset that is collected in photo-realistic simulation environments. BlendedMVS [24] introduces more large-scale scenes and improves the performance of many MVS methods on real-world scenes, shown in Tab. 3. ArKitScenes [198] consists of 5,048 RGB-D sequences, which is more than three times the size of the current largest available indoor dataset, ScanNet. As an extension of ScanNet, ScanNet++ [199] is a large-scale dataset that captures high-quality geometry and color of indoor scenes with high-end laser scanner, DSLR camera, and RGB-D streams from an iPhone. Motivated by ImageNet [200] that drives a remarkable trend of learning from large-scale data in 2D visual tasks, Yu *et al.* [201] propose MVImgNet, a large-scale dataset of multi-view images collected by shooting videos of real-world objects. MVImgNet demonstrates the potential of various 3D visual tasks, including radiance field reconstruction, multi-view stereo, and view-consistent image understanding. Similarly, Objaverse [29] is a large dataset of objects with 800K+ (and growing) 3D models with descriptive captions, tags, and animations. Both MVImgNet [200] and Objaverse [29] can be used to train large feed-forward models, *e.g.*, LRM [54], for reconstruction, since these models need massive data to learn the strong prior. However, both of them mainly focus

TABLE 5
Quantitative results of NeRF-based MVS methods on DTU [21].

Methods	VolSDF [48]	NeuS [49]	NeuralWarp [129]	HF-NeuS [190]	RegSDF [191]	PET-NeuS [192]
Overall ↓	0.86	0.87	0.68	0.77	0.72	0.71
Methods	Geo-NeuS [50]	Voxurf [193]	NeuS2 [148]	PermutoSDF [150]	Neuralangelo [149]	UniSDF [151]
Overall ↓	0.51	0.72	0.70	0.68	0.61	0.64

on object-level scenes. Large datasets for larger-scale scenes, *e.g.*, rooms, are valuable to be explored.

For evaluation, the main benchmarks [21], [22], [23], [26] are all introduced before 2018. Some of these benchmarks are currently saturated and the methods are difficult to further improve the performance on them. Therefore, it is meaningful to introduce new benchmarks to evaluate the robustness and performance of learning-based methods in challenging real-world scenes. For example, LaMAR [202] is a recent large-scale dataset captured using multiple modern AR devices in diverse environments. It contains challenging short-term appearance and structural changes and high quality LiDAR point cloud ground truth. These make LaMAR a potential benchmark for MVS.

6.5.2 View Selection

As discussed in Sec. 2.2, view selection is crucial for triangulation quality. Picking neighboring views that are suitable for triangulation can not only improve the reconstruction accuracy but also reduce useless computation for the bad views, *e.g.*, views with strong occlusions. However, view selection is often overlooked and not well studied. For example, all offline depth map-based MVS methods [31], [39], [79], [100] follow MVSNet [15] and use the same simple heuristic strategy to compute scores for neighboring views and sort them. Online depth map-based MVS methods [30], [34], [60] and voxel-based methods [47] adopt heuristic strategies to choose views with enough pose-distance. Though it is intractable to incorporate non-differentiable view selection into deep learning, it is worth exploring new view selection strategies since it may improve the reconstruction without changing the model design. Current view selection strategies simply pick the same set of neighboring views for all the pixels in the reference image, and it is probable that different reference pixels have different optimal choices of neighboring views. The learned pixel-wise view weight [39], [40], [94] is one solution for this since it weights the source views differently for each pixel based on its visibility across neighboring views.

6.5.3 Depth Fusion

As discussed in Sec. 2.4, both online and offline MVS methods with plane-sweep use traditional TSDF fusion or depth filtering methods to reconstruct mesh or point cloud from the estimated depth maps. However, TSDF fusion [65], [66] is not robust enough to the outliers in the depth maps and may have memory issues due to the dense volumetric representation. Depth filtering following Galliani *et al.* [18] introduces lots of hyper-parameters. Researchers may carefully finetune these hyper-parameters for each scene since it may have a great impact on the evaluation metrics. Therefore, it is valuable to improve the depth fusion step

so that the reconstruction quality can be further improved. Recently, there have been some learning-based depth map fusion methods [203], [204]. However, they are limited to small scenes due to the dense volume representation and large memory consumption.

6.5.4 Features

One not well-studied topic is what kind of feature extractors are suitable for MVS, as mentioned in Sec. 3.1. So far, most of the offline depth map-based methods [15], [31], [78] mainly apply simple 2D CNN and FPN [72] structure as a feature extractor. To improve the receptive fields flexibly, Deformation Convolution [205] is used [39], [77], [78], [79]). The attention mechanism is also applied in feature learning, where works such as [77], [79], [91] use intra-attention or inter-attention to capture long-range feature dependencies. Furthermore, MVSFormer [82] digs deeper into using patch-wise ViT [177] as a feature extractor in MVS. The insight is that ViT works better to formulate global feature correlations, and FPN can learn detailed ones. In contrast, online depth map-based methods [30], [34] usually adopt efficient pretrained backbones [70], [73], [74]. Recently, large feed-forward methods [54], [55], [166] adopt large pretrained feature encoders, *e.g.*, ViT [177], DINO [83], to learn strong prior from large-amount of data. However, using a powerful feature extraction network usually increases the computation overhead and reduces the efficiency. Finding a good balance between performance and efficiency is an interesting direction, *e.g.*, distilling the large ViT into a relatively small transformer [206].

6.5.5 Real-time and Memory Efficient Models

Though reconstruction accuracy is considered the most important factor in MVS, it is also critical to have models that can run in near real-time and with low memory, *e.g.*, AR/VR and autonomous driving. Comparatively, depth map-based methods are most suitable for achieving efficiency because of their conciseness, flexibility, and scalability. Though online MVS methods can achieve real-time estimation with images of low resolutions, they may have issues with memory consumption since many large backbones are usually used for feature extraction [30], [34]. In addition, the efficiency in both run-time and memory will drop when image resolution increases. For offline MVS methods, as discussed in Sec. 6.2.2, some recent methods try to improve efficiency with carefully simplified network architectures. However, this usually limits the performance when compared with state-of-the-art methods. There are many other directions for further improvement. For example, using model compression techniques and knowledge distillation are potential directions to not only keep the good performance and also improve efficiency.

6.5.6 Prior Assistance

MVS mainly relies on assessing local photometric consistency to find the optimal matching across reference and source images. Accordingly, it usually encounters difficulties when estimating the geometry for regions where the photometric measurement becomes unreliable or invalid, *e.g.*, textureless areas and non-Lambertian surfaces, which are common in human-made scenes [23]. Therefore, using prior information to guide the MVS algorithm in these challenging regions is a promising research direction. We elaborate 3 typical examples of prior assistance as follows.

Surface Normal: As a non-local representation of the geometry compared with the depth map, surface normal has been proven effective in improving reconstruction performance in recent works. To enforce the constraints of normal maps in depth estimation, Kusupati *et al.* [207] integrate a multi-view normal estimation network into the MVS pipeline. Long *et al.* [208] introduce a Combined Normal Map, estimated with PlaneCNN [209] to enforce the normal consistency on the depth map. Liu *et al.* [210] apply graph-based depth map optimization as post-processing. Since monocular normal network [211] is trained on large-scale dataset and can provide high-quality priors, [146], [187] use monocular normal priors to improve the surface accuracy.

Shape Prior: For indoor scenes where common textureless areas, *e.g.*, walls, planes are suitable choices of the geometric primitives and are exploited in traditional methods [20]. Recently, PlaneMVS [212] and PlanarRecon [213] explicitly estimate plane parameters for depth refinement or holistic reconstruction. In addition, predicting object-level attributes simultaneously when estimating depth [214] is also a feasible solution for specific applications like urban modeling.

Semantic Segmentation: Intuitively, points assigned with the same semantic labels may be more likely to lie on the same 3D plane. Some attempts [215], [216] employ a rule-based protocol to utilize semantic segmentation for better matching and depth quality. Manhattan-SDF [217] relies on 2D segmentation results to enforce Manhattan world priors to handle low-textured planar regions. Similarly, Shvets *et al.* [218] also employ SAM (Segment Anything Model) encoder to generate semantic features to enhance MVS.

7 CONCLUSION

In this survey, we have provided a comprehensive review of the learning-based MVS methods, which are categorized into: depth map-based, voxel-based, NeRF-based, 3D Gaussian Splatting-based and large feed-forward methods. Particularly, we have devoted significant attention to depth map-based methods because of their conciseness, flexibility and scalability. We explain key aspects such as datasets utilized, general working pipelines, and algorithmic intricacies. Furthermore, we have summarized and compared the quantitative performance of these methods across popular benchmarks, providing valuable insights into their efficacy and applicability. Finally, our discourse extends to an in-depth exploration of potential research directions for the future of MVS, highlighting avenues for further investigation and innovation in the field.

REFERENCES

- [1] Y. Furukawa and C. Hernández, "Multi-view stereo: A tutorial," *Foundations and Trends® in Computer Graphics and Vision*, 2015.
- [2] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *IROS*, 2012.
- [3] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *IEEE TRO*, 2013.
- [4] T. Schops, T. Sattler, and M. Pollefeys, "Bad slam: Bundle adjusted direct rgb-d slam," in *CVPR*, 2019.
- [5] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "Mulls: Versatile lidar slam via multi-metric linear least square," in *ICRA*, 2021.
- [6] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *ACM UIST*, 2011.
- [7] M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt *et al.*, "Real-time non-rigid reconstruction using an rgb-d camera," *ACM ToG*, 2014.
- [8] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *CVPR*, 2015.
- [9] S. M. Seitz and C. R. Dyer, "Photorealistic scene reconstruction by voxel coloring," *IJCV*, 1999.
- [10] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *IJCV*, 2000.
- [11] I. Kostrikov, E. Horbert, and B. Leibe, "Probabilistic labeling cost for high-accuracy multi-view reconstruction," in *CVPR*, 2014.
- [12] A. O. Ulusoy, M. J. Black, and A. Geiger, "Semantic multi-view stereo: Jointly estimating objects and voxels," in *CVPR*, 2017.
- [13] M. Lhuillier and L. Quan, "A quasi-dense approach to surface reconstruction from uncalibrated images," *IEEE TPAMI*, 2005.
- [14] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE TPAMI*, 2009.
- [15] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in *ECCV*, 2018.
- [16] R. Yang and M. Pollefeys, "Multi-resolution real-time stereo on commodity graphics hardware," in *CVPR*, 2003.
- [17] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *CVPR*, 2016.
- [18] S. Galliani, K. Lasinger, and K. Schindler, "Massively parallel multiview stereopsis by surface normal diffusion," in *ICCV*, 2015.
- [19] Q. Xu and W. Tao, "Multi-scale geometric consistency guided multi-view stereo," in *CVPR*, 2019.
- [20] ———, "Planar prior assisted patchmatch multi-view stereo," in *AAAI*, 2020.
- [21] H. Aanaes, R. R. Jensen, G. Vogiatzis, E. Tola, and A. B. Dahl, "Large-scale data for multiple-view stereopsis," *IJCV*, 2016.
- [22] A. Knapsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM ToG*, 2017.
- [23] 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.
- [24] 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.
- [25] K. Wang and S. Shen, "Mvdepthnet: Real-time multiview depth estimation neural network," in *3DV*, 2018.
- [26] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *CVPR*, 2017.
- [27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.
- [28] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM ToG*, 2023.
- [29] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. Van der Bilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi, "Objaverse: A universe of annotated 3d objects," in *CVPR*, 2023.
- [30] A. Duzceker, S. Galliani, C. Vogel, P. Speciale, M. Dusmanu, and M. Pollefeys, "Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion," in *CVPR*, 2021.

- [31] 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.
- [32] R. T. Collins, "A space-sweep approach to true multi-image matching," in *CVPR*, 1996.
- [33] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys, "Pixelwise view selection for unstructured multi-view stereo," in *ECCV*, 2016.
- [34] M. Sayed, J. Gibson, J. Watson, V. Prisacariu, M. Firman, and C. Godard, "Simplerecon: 3d reconstruction without 3d convolutions," in *ECCV*, 2022.
- [35] Q. Xu and W. Tao, "Learning inverse depth regression for multi-view stereo with correlation cost volume," in *AAAI*, 2020.
- [36] 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.
- [37] 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*, 2020.
- [38] 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.
- [39] F. Wang, S. Galliani, C. Vogel, P. Speciale, and M. Pollefeys, "Patchmatchnet: Learned multi-view patchmatch stereo," in *CVPR*, 2021.
- [40] F. Wang, S. Galliani, C. Vogel, and M. Pollefeys, "Itermvs: Iterative probability estimation for efficient multi-view stereo," 2022.
- [41] H. Xu, Z. Zhou, Y. Qiao, W. Kang, and Q. Wu, "Self-supervised multi-view stereo via effective co-segmentation and data-augmentation," in *AAAI*, 2021.
- [42] D. Chang, A. Božić, T. Zhang, Q. Yan, Y. Chen, S. Süstrunk, and M. Nießner, "Rc-mvsnet: unsupervised multi-view stereo with neural rendering," in *ECCV*, 2022.
- [43] H. Xu, Z. Zhou, Y. Wang, W. Kang, B. Sun, H. Li, and Y. Qiao, "Digging into uncertainty in self-supervised multi-view stereo," in *ICCV*, 2021.
- [44] Y. Ding, Q. Zhu, X. Liu, W. Yuan, H. Zhang, and C. Zhang, "Kdmvs: Knowledge distillation based self-supervised learning for multi-view stereo," in *ECCV*, 2022.
- [45] T. Kim, J. Choi, S. Choi, D. Jung, and C. Kim, "Just a few points are all you need for multi-view stereo: A novel semi-supervised learning method for multi-view stereo," in *ICCV*, 2021.
- [46] Z. Murez, T. v. As, J. Bartolozzi, A. Sinha, V. Badrinarayanan, and A. Rabinovich, "Atlas: End-to-end 3d scene reconstruction from posed images," in *ECCV*, 2020.
- [47] J. Sun, Y. Xie, L. Chen, X. Zhou, and H. Bao, "Neuralrecon: Real-time coherent 3d reconstruction from monocular video," in *CVPR*, 2021.
- [48] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, "Volume rendering of neural implicit surfaces," in *NeurIPS*, 2021.
- [49] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," in *NeurIPS*, 2021.
- [50] Q. Fu, Q. Xu, Y. S. Ong, and W. Tao, "Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction," in *NeurIPS*, 2022.
- [51] A. Guédon and V. Lepetit, "Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering," in *CVPR*, 2024.
- [52] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2d gaussian splatting for geometrically accurate radiance fields," in *ACM SIGGRAPH*, 2024.
- [53] D. Chen, H. Li, W. Ye, Y. Wang, W. Xie, S. Zhai, N. Wang, H. Liu, H. Bao, and G. Zhang, "Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction," *arXiv preprint*, 2024.
- [54] Y. Hong, K. Zhang, J. Gu, S. Bi, Y. Zhou, D. Liu, F. Liu, K. Sunkavalli, T. Bui, and H. Tan, "Lrm: Large reconstruction model for single image to 3d," *ICLR*, 2024.
- [55] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, "Dust3r: Geometric 3d vision made easy," in *CVPR*, 2024.
- [56] V. Leroy, Y. Cabon, and J. Revaud, "Grounding image matching in 3d with mast3r," *arXiv preprint*, 2024.
- [57] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [58] P. Moulon, P. Monasse, R. Perrot, and R. Marlet, "Openmvg: Open multiple view geometry," in *International Workshop on Reproducible Research in Pattern Recognition*, 2016.
- [59] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM ToG*, 2017.
- [60] Y. Hou, J. Kannala, and A. Solin, "Multi-view stereo by temporal nonparametric fusion," in *ICCV*, 2019.
- [61] R. Zhang, S. Li, T. Fang, S. Zhu, and L. Quan, "Joint camera clustering and surface segmentation for large-scale multi-view stereo," in *ICCV*, 2015.
- [62] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," in *ICCV*, 2017.
- [63] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *CVPR*, 2018.
- [64] X. Guo, K. Yang, W. Yang, X. Wang, and H. Li, "Group-wise correlation stereo network," in *CVPR*, 2019.
- [65] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *ACM SIGGRAPH*, 1996.
- [66] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molnyea, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *ISMAR*, 2011.
- [67] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *Seminal graphics: pioneering efforts that shaped the field*, 1998.
- [68] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *NeurIPS*, 2014.
- [69] A. Sinha, Z. Murez, J. Bartolozzi, V. Badrinarayanan, and A. Rabinovich, "Deltas: Depth estimation by learning triangulation and densification of sparse points," in *ECCV*, 2020.
- [70] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.
- [71] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.
- [72] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017.
- [73] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "Mnasnet: Platform-aware neural architecture search for mobile," in *CVPR*, 2019.
- [74] M. Tan and Q. Le, "Efficientnetv2: Smaller models and faster training," in *ICML*, 2021.
- [75] J. Yang, W. Mao, J. M. Alvarez, and M. Liu, "Cost volume pyramid based depth inference for multi-view stereo," in *CVPR*, 2020.
- [76] Z. Zhang, R. Peng, Y. Hu, and R. Wang, "Geomvsnet: Learning multi-view stereo with geometry perception," in *CVPR*, 2023.
- [77] Z. Wei, Q. Zhu, C. Min, Y. Chen, and G. Wang, "Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network," in *ICCV*, 2021.
- [78] Z. Mi, D. Chang, and D. Xu, "Generalized binary search network for highly-efficient multi-view stereo," in *CVPR*, 2022.
- [79] 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.
- [80] K. T. Giang, S. Song, and S. Jo, "Curvature-guided dynamic scale networks for multi-view stereo," *arXiv preprint*, 2021.
- [81] J. Liao, Y. Ding, Y. Shavit, D. Huang, S. Ren, J. Guo, W. Feng, and K. Zhang, "Wt-mvsnet: window-based transformers for multi-view stereo," *NeurIPS*, 2022.
- [82] C. Cao, X. Ren, and Y. Fu, "Mvsformer: Learning robust image representations via transformers and temperature-based depth for multi-view stereo," *arXiv preprint*, 2022.
- [83] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *ICCV*, 2021.
- [84] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, "Twins: Revisiting the design of spatial attention in vision transformers," *NeurIPS*, 2021.
- [85] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.
- [86] 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.

- [87] C. Liu, J. Gu, K. Kim, S. G. Narasimhan, and J. Kautz, "Neural rgb (r) d sensing: Depth and uncertainty from a video camera," in *CVPR*, 2019.
- [88] G. Bae, I. Budvytis, and R. Cipolla, "Multi-view depth estimation by fusing single-view depth probability with multi-view geometry," in *CVPR*, 2022.
- [89] M. Sayed, F. Aleotti, J. Watson, Z. Qureshi, G. Garcia-Hernando, G. Brostow, S. Vicente, and M. Firman, "Doubletake: Geometry guided depth estimation," *arXiv preprint*, 2024.
- [90] X. Guo, K. Yang, W. Yang, X. Wang, and H. Li, "Group-wise correlation stereo network," in *CVPR*, 2019.
- [91] H. Yi, Z. Wei, M. Ding, R. Zhang, Y. Chen, G. Wang, and Y.-W. Tai, "Pyramid multi-view stereo net with self-adaptive view aggregation," in *ECCV*, 2020.
- [92] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *ICCV*, 2019.
- [93] J. Zhang, Y. Yao, S. Li, Z. Luo, and T. Fang, "Visibility-aware multi-view stereo network," *BMVC*, 2020.
- [94] Q. Xu, W. Su, Y. Qi, W. Tao, and M. Pollefeys, "Learning inverse depth regression for pixelwise visibility-aware multi-view stereo networks," *IJCV*, 2022.
- [95] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *NeurIPS*, 2015.
- [96] S. Im, H.-G. Jeon, S. Lin, and I. S. Kweon, "Dpsnet: End-to-end deep plane sweep stereo," in *ICLR*, 2018.
- [97] R. Chen, S. Han, J. Xu, and H. Su, "Point-based multi-view stereo network," in *ICCV*, 2019.
- [98] K. Luo, T. Guan, L. Ju, H. Huang, and Y. Luo, "P-MVSNet: Learning patch-wise matching confidence aggregation for multi-view stereo," in *ICCV*, 2019.
- [99] 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*, 2014.
- [100] 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.
- [101] 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.
- [102] J. Y. Lee, J. DeGol, C. Zou, and D. Hoiem, "Patchmatch-rl: Deep mvs with pixelwise depth, normal, and visibility," in *ICCV*, 2021.
- [103] Z. Ma, Z. Teed, and J. Deng, "Multiview stereo with cascaded epipolar raft," *arXiv preprint*, 2022.
- [104] S. Wang, B. Li, and Y. Dai, "Efficient multi-view stereo by iterative dynamic cost volume," in *CVPR*, 2022.
- [105] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM ToG*, 2009.
- [106] M. Bleyer, C. Rhemann, and C. Rother, "Patchmatch stereo - stereo matching with slanted support windows," in *BMVC*, 2011.
- [107] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *ECCV*, 2020.
- [108] C. Sormann, E. Santellani, M. Rossi, A. Kuhn, and F. Fraundorfer, "Dels-mvs: Deep epipolar line search for multi-view stereo," in *WACV*, 2023.
- [109] C. Cai, P. Ji, Q. Yan, and Y. Xu, "Riav-mvs: Recurrent-indexing an asymmetric volume for multi-view stereo," in *CVPR*, 2023.
- [110] G. Xu, X. Wang, X. Ding, and X. Yang, "Iterative geometry encoding volume for stereo matching," in *CVPR*, 2023.
- [111] R. Peng, R. Wang, Z. Wang, Y. Lai, and R. Wang, "Rethinking depth estimation for multi-view stereo: A unified representation," in *CVPR*, 2022.
- [112] X. Ye, W. Zhao, T. Liu, Z. Huang, Z. Cao, and X. Li, "Constraining depth map geometry for multi-view stereo: A dual-depth approach with saddle-shaped depth cells," in *ICCV*, 2023.
- [113] Z. Yu and S. Gao, "Fast-mvsnet: Sparse-to-dense multi-view stereo with learned propagation and gauss-newton refinement," in *CVPR*, 2020.
- [114] T.-W. Hui, C. C. Loy, and X. Tang, "Depth map super-resolution by deep multi-scale guidance," in *ECCV*, 2016.
- [115] J. Xi, Y. Shi, Y. Wang, Y. Guo, and K. Xu, "Raymvsnet: Learning ray-based 1d implicit fields for accurate multi-view stereo," in *CVPR*, 2022.
- [116] Y. Shi, J. Xi, D. Hu, Z. Cai, and K. Xu, "Raymvsnet++: learning ray-based 1d implicit fields for accurate multi-view stereo," *IEEE TPAMI*, 2023.
- [117] W. Su and W. Tao, "Efficient edge-preserving multi-view stereo network for depth estimation," in *AAAI*, 2023.
- [118] M. Poggi and S. Mattoccia, "Learning from scratch a confidence measure," in *BMVC*, 2016.
- [119] Z. Fu, M. Ardabiliyan, and G. Stern, "Stereo matching confidence learning based on multi-modal convolution neural networks," in *International Workshop on Representations, Analysis and Recognition of Shape and Motion From Imaging Data*, 2017.
- [120] F. Tosi, M. Poggi, A. Benincasa, and S. Mattoccia, "Beyond local reasoning for stereo confidence estimation with deep learning," in *ECCV*, 2018.
- [121] S. Kim, S. Kim, D. Min, and K. Sohn, "Laf-net: Locally adaptive fusion networks for stereo confidence estimation," in *CVPR*, 2019.
- [122] Z. Li, W. Zuo, Z. Wang, and L. Zhang, "Confidence-based large-scale dense multi-view stereo," *IEEE TIP*, 2020.
- [123] A. Kuhn, C. Sormann, M. Rossi, O. Erdler, and F. Fraundorfer, "Deeepc-mvs: Deep confidence prediction for multi-view stereo reconstruction," in *3DV*, 2020.
- [124] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *ICCV*, 2017.
- [125] Y. Dai, Z. Zhu, Z. Rao, and B. Li, "Mvs2: Deep unsupervised multi-view stereo with multi-view symmetry," in *3DV*, 2019.
- [126] B. Huang, H. Yi, C. Huang, Y. He, J. Liu, and X. Liu, "M3vsnet: Unsupervised multi-metric multi-view stereo network," in *ICIP*, 2021.
- [127] T. Khot, S. Agrawal, S. Tulsiani, C. Mertz, S. Lucey, and M. Hebert, "Learning unsupervised multi-view stereopsis via robust photometric consistency," *arXiv preprint*, 2019.
- [128] J. Yang, J. M. Alvarez, and M. Liu, "Self-supervised learning of depth inference for multi-view stereo," in *CVPR*, 2021.
- [129] F. Darmon, B. Bascl, J.-C. Devaux, P. Monasse, and M. Aubry, "Deep multi-view stereo gone wild," in *3DV*, 2021.
- [130] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE TIP*, 2004.
- [131] J. Zhang, R. Tang, Z. Cao, J. Xiao, R. Huang, and L. Fang, "Elasticmvs: Learning elastic part representation for self-supervised multi-view stereopsis," *NeurIPS*, 2022.
- [132] K. Xiong, R. Peng, Z. Zhang, T. Feng, J. Jiao, F. Gao, and R. Wang, "Cl-mvsnet: Unsupervised multi-view stereo with dual-level contrastive learning," in *ICCV*, 2023.
- [133] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *CVPR*, 2018.
- [134] A. Bozic, P. Palafox, J. Thies, A. Dai, and M. Nießner, "Transformerfusion: Monocular rgb scene reconstruction using transformers," *NeurIPS*, 2021.
- [135] N. Stier, A. Rich, P. Sen, and T. Höllerer, "Vortx: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion," in *3DV*, 2021.
- [136] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *CVPR*, 2022.
- [137] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "Tensorrf: Tensorial radiance fields," in *ECCV*, 2022.
- [138] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM ToG*, 2022.
- [139] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," in *CVPR*, 2022.
- [140] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *CVPR*, 2022.
- [141] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi, "Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures," in *CVPR*, 2023.
- [142] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Zip-nerf: Anti-aliased grid-based neural radiance fields," in *ICCV*, 2023.
- [143] C. Reiser, R. Szeliski, D. Verbin, P. Srinivasan, B. Mildenhall, A. Geiger, J. Barron, and P. Hedman, "Merf: Memory-efficient ra-

- diance fields for real-time view synthesis in unbounded scenes," *ACM TOG*, 2023.
- [144] Q. Gao, Q. Xu, H. Su, U. Neumann, and Z. Xu, "Strivec: Sparse tri-vector radiance fields," in *ICCV*, 2023.
- [145] Y. Wei, S. Liu, Y. Rao, W. Zhao, J. Lu, and J. Zhou, "Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo," in *ICCV*, 2021.
- [146] Z. Yu, S. Peng, M. Niemeyer, T. Sattler, and A. Geiger, "Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction," *NeurIPS*, 2022.
- [147] J. Wang, P. Wang, X. Long, C. Theobalt, T. Komura, L. Liu, and W. Wang, "Neuris: Neural reconstruction of indoor scenes using normal priors," in *ECCV*, 2022.
- [148] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, "Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction," in *ICCV*, 2023.
- [149] Z. Li, T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin, "Neuralangelo: High-fidelity neural surface reconstruction," in *CVPR*, 2023.
- [150] R. A. Rosu and S. Behnke, "Permutosdf: Fast multi-view reconstruction with implicit surfaces using permutohedral lattices," in *CVPR*, 2023.
- [151] F. Wang, M.-J. Rakotosaona, M. Niemeyer, R. Szeliski, M. Pollefeys, and F. Tombari, "Unisdf: Unifying neural representations for high-fidelity 3d reconstruction of complex scenes with reflections," *arXiv preprint*, 2023.
- [152] X. Long, C. Lin, P. Wang, T. Komura, and W. Wang, "Sparseneus: Fast generalizable neural surface reconstruction from sparse views," in *ECCV*, 2022.
- [153] Y. Ren, F. Wang, T. Zhang, M. Pollefeys, and S. Süstrunk, "Volrecon: Volume rendering of signed ray distance functions for generalizable multi-view reconstruction," in *CVPR*, 2023.
- [154] Y. Liang, H. He, and Y.-C. Chen, "Retr: Modeling rendering via transformer for generalizable neural surface reconstruction," in *NeurIPS*, 2023.
- [155] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-nerf: Structured view-dependent appearance for neural radiance fields," in *CVPR*, 2022.
- [156] L. Yariv, P. Hedman, C. Reiser, D. Verbin, P. P. Srinivasan, R. Szeliski, J. T. Barron, and B. Mildenhall, "Bakedsdf: Meshing neural sdf's for real-time view synthesis," in *ACM SIGGRAPH*, 2023.
- [157] Y. Liu, P. Wang, C. Lin, X. Long, J. Wang, L. Liu, T. Komura, and W. Wang, "Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images," *arXiv preprint*, 2023.
- [158] R. Liang, H. Chen, C. Li, F. Chen, S. Panneer, and N. Vijaykumar, "Envindr: Implicit differentiable renderer with neural environment lighting," in *ICCV*, 2023.
- [159] W. Ge, T. Hu, H. Zhao, S. Liu, and Y.-C. Chen, "Ref-neus: Ambiguity-reduced neural implicit surface learning for multi-view reconstruction with reflection," in *ICCV*, 2023.
- [160] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM ToG*, 2023.
- [161] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," in *ACM SIGGRAPH year=2006*.
- [162] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *SGP*, 2006.
- [163] H. Chen, C. Li, and G. H. Lee, "Neusg: Neural implicit surface reconstruction with 3d gaussian splatting guidance," *arXiv preprint*, 2023.
- [164] B. Zhang, C. Fang, R. Shrestha, Y. Liang, X. Long, and P. Tan, "Rade-gs: Rasterizing depth in gaussian splatting," *arXiv preprint*, 2024.
- [165] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, 2017.
- [166] J. Han, F. Kokkinos, and P. Torr, "Vfusion3d: Learning scalable 3d generative models from video diffusion models," *arXiv preprint*, 2024.
- [167] D. Tochilkin, D. Pankratz, Z. Liu, Z. Huang, A. Letts, Y. Li, D. Liang, C. Laforte, V. Jampani, and Y.-P. Cao, "Triposr: Fast 3d object reconstruction from a single image," *arXiv preprint*, 2024.
- [168] Y. Xu, H. Tan, F. Luan, S. Bi, P. Wang, J. Li, Z. Shi, K. Sunkavalli, G. Wetzstein, Z. Xu *et al.*, "Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model," *ICLR*, 2024.
- [169] J. Li, H. Tan, K. Zhang, Z. Xu, F. Luan, Y. Xu, Y. Hong, K. Sunkavalli, G. Shakhnarovich, and S. Bi, "Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model," *ICLR*, 2024.
- [170] A. Chen, H. Xu, S. Esposito, S. Tang, and A. Geiger, "Lara: Efficient large-baseline radiance fields," in *ECCV*, 2024.
- [171] K. Zhang, S. Bi, H. Tan, Y. Xiangli, N. Zhao, K. Sunkavalli, and Z. Xu, "Gs-lrm: Large reconstruction model for 3d gaussian splatting," *arXiv preprint*, 2024.
- [172] Y. Xu, Z. Shi, W. Yifan, H. Chen, C. Yang, S. Peng, Y. Shen, and G. Wetzstein, "Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation," *arXiv preprint*, 2024.
- [173] J. Tang, Z. Chen, X. Chen, T. Wang, G. Zeng, and Z. Liu, "Lgm: Large multi-view gaussian model for high-resolution 3d content creation," *arXiv preprint*, 2024.
- [174] P. Wang, H. Tan, S. Bi, Y. Xu, F. Luan, K. Sunkavalli, W. Wang, Z. Xu, and K. Zhang, "Pf-lrm: Pose-free large reconstruction model for joint pose and shape prediction," *ICLR*, 2024.
- [175] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. J. Guibas, J. Tremblay, S. Khamis *et al.*, "Efficient geometry-aware 3d generative adversarial networks," in *CVPR*, 2022.
- [176] S. Szymanowicz, C. Rupprecht, and A. Vedaldi, "Splatter image: Ultra-fast single-view 3d reconstruction," in *CVPR*, 2024.
- [177] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.
- [178] H. Wang and L. Agapito, "3d reconstruction with spatial memory," *arXiv preprint*, 2024.
- [179] B. Glocker, S. Izadi, J. Shotton, and A. Criminisi, "Real-time rgbd camera relocalization," in *ISMAR*, 2013.
- [180] K. Luo, T. Guan, L. Ju, H. Huang, and Y. Luo, "P-mvsnet: Learning patch-wise matching confidence aggregation for multi-view stereo," in *ICCV*, 2019.
- [181] Z. Wei, Q. Zhu, C. Min, Y. Chen, and G. Wang, "Bidirectional hybrid lstm based recurrent neural network for multi-view stereo," *IEEE TVCG*, 2022.
- [182] K. Luo, T. Guan, L. Ju, Y. Wang, Z. Chen, and Y. Luo, "Attention-aware multi-view stereo," in *CVPR*, 2020.
- [183] J. Yang, J. M. Alvarez, and M. Liu, "Non-parametric depth distribution modelling based depth inference for multi-view stereo," in *CVPR*, 2022.
- [184] L. Wang, Y. Gong, X. Ma, Q. Wang, K. Zhou, and L. Chen, "Is-mvsnet: Importance sampling-based mvsnet," in *ECCV*, 2022.
- [185] Q. Zhu, Z. Wei, Z. Wang, Y. Chen, and G. Wang, "Hybrid cost volume regularization for memory-efficient multi-view stereo networks," in *BMVC*, 2022.
- [186] Y. Zhang, J. Zhu, and L. Lin, "Multi-view stereo representation revist: Region-aware mvsnet," in *CVPR*, 2023.
- [187] J. Wu, R. Li, H. Xu, W. Zhao, Y. Zhu, J. Sun, and Y. Zhang, "Gomvs: Geometrically consistent cost aggregation for multi-view stereo," in *CVPR*, 2024.
- [188] X. Long, L. Liu, W. Li, C. Theobalt, and W. Wang, "Multi-view depth estimation using epipolar spatio-temporal networks," in *CVPR*, 2021.
- [189] Q. Xu, W. Kong, W. Tao, and M. Pollefeys, "Multi-scale geometric consistency guided and planar prior assisted multi-view stereo," *IEEE TPAMI*, 2022.
- [190] Y. Wang, I. Skorokhodov, and P. Wonka, "Hf-neus: Improved surface reconstruction using high-frequency details," *NeurIPS*, 2022.
- [191] J. Zhang, Y. Yao, S. Li, T. Fang, D. McKinnon, Y. Tsin, and L. Quan, "Critical regularizations for neural surface reconstruction in the wild," in *CVPR*, 2022.
- [192] Y. Wang, I. Skorokhodov, and P. Wonka, "Pet-neus: Positional encoding tri-planes for neural surfaces," in *CVPR*, 2023.
- [193] T. Wu, J. Wang, X. Pan, X. Xu, C. Theobalt, Z. Liu, and D. Lin, "Voxurf: Voxel-based efficient and accurate neural surface reconstruction," *arXiv preprint*, 2022.
- [194] D. Charatan, S. Li, A. Tagliasacchi, and V. Sitzmann, "pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction," in *CVPR*, 2024.
- [195] Y. Chen, H. Xu, C. Zheng, B. Zhuang, M. Pollefeys, A. Geiger, T.-J. Cham, and J. Cai, "Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images," in *ECCV*, 2024.

- [196] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM ToG*, 2013.
- [197] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *IROS*, 2020.
- [198] G. Baruch, Z. Chen, A. Dehghan, T. Dimry, Y. Feigin, P. Fu, T. Gebauer, B. Joffe, D. Kurz, A. Schwartz *et al.*, "Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgbd data," *arXiv preprint*, 2021.
- [199] C. Yeshwanth, Y.-C. Liu, M. Nießner, and A. Dai, "Scannet++: A high-fidelity dataset of 3d indoor scenes," in *ICCV*, 2023.
- [200] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.
- [201] X. Yu, M. Xu, Y. Zhang, H. Liu, C. Ye, Y. Wu, Z. Yan, C. Zhu, Z. Xiong, T. Liang *et al.*, "Mvimgnet: A large-scale dataset of multi-view images," in *CVPR*, 2023.
- [202] P.-E. Sarlin, M. Dusmanu, J. L. Schönberger, P. Speciale, L. Gruber, V. Larsson, O. Miksik, and M. Pollefeys, "Lamar: Benchmarking localization and mapping for augmented reality," in *ECCV*, 2022.
- [203] S. Weder, J. Schönberger, M. Pollefeys, and M. R. Oswald, "Routefusion: Learning real-time depth map fusion," in *CVPR*, 2020.
- [204] S. Weder, J. L. Schönberger, M. Pollefeys, and M. R. Oswald, "Neuralfusion: Online depth fusion in latent space," in *CVPR*, 2021.
- [205] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *ICCV*, 2017.
- [206] K. Wu, J. Zhang, H. Peng, M. Liu, B. Xiao, J. Fu, and L. Yuan, "Tinyvit: Fast pretraining distillation for small vision transformers," in *ECCV*, 2022.
- [207] U. Kusupati, S. Cheng, R. Chen, and H. Su, "Normal assisted stereo depth estimation," in *CVPR*, 2020.
- [208] X. Long, L. Liu, C. Theobalt, and W. Wang, "Occlusion-aware depth estimation with adaptive normal constraints," in *ECCV*, 2020.
- [209] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, "Planercnn: 3d plane detection and reconstruction from a single image," in *CVPR*, 2019.
- [210] H. Liu, X. Tang, and S. Shen, "Depth-map completion for large indoor scene reconstruction," *Pattern Recognition*, 2020.
- [211] A. Eftekhar, A. Sax, J. Malik, and A. Zamir, "Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans," in *ICCV*, 2021.
- [212] J. Liu, P. Ji, N. Bansal, C. Cai, Q. Yan, X. Huang, and Y. Xu, "Planemvs: 3d plane reconstruction from multi-view stereo," in *CVPR*, 2022.
- [213] Y. Xie, M. Gadelha, F. Yang, X. Zhou, and H. Jiang, "Planarrecon: Real-time 3d plane detection and reconstruction from posed monocular videos," in *CVPR*, 2022.
- [214] A. Osman Ulusoy, M. J. Black, and A. Geiger, "Semantic multi-view stereo: Jointly estimating objects and voxels," in *CVPR*, 2017.
- [215] E.-K. Stathopoulou and F. Remondino, "Semantic photogrammetry-boosting image-based 3d reconstruction with semantic labeling," *ISPRS Archives*, 2019.
- [216] E. K. Stathopoulou, R. Battisti, D. Cernea, F. Remondino, and A. Georgopoulos, "Semantically derived geometric constraints for mvs reconstruction of textureless areas," *Remote Sensing*, 2021.
- [217] H. Guo, S. Peng, H. Lin, Q. Wang, G. Zhang, H. Bao, and X. Zhou, "Neural 3d scene reconstruction with the manhattan-world assumption," in *CVPR*, 2022.
- [218] M. Shvets, D. Zhao, M. Niethammer, R. Sengupta, and A. C. Berg, "Joint depth prediction and semantic segmentation with multi-view sam," in *WACV*, 2024.



Fangjinhua Wang is currently pursuing the PhD degree in Computer Science at Computer Vision and Geometry Group, ETH Zurich. He previously received the MSc degree in Robotics at ETH Zurich. He has a broad interest in 3D computer vision and deep learning, including 3D reconstruction, novel view synthesis, and visual localization and mapping.



Qingtian Zhu is a PhD student at the Graduate School of Information Science and Technology, The University of Tokyo, Japan. Prior to this, he received the MSc degree from the School of Computer Science, Peking University, China. His fields of research are computer vision and computer graphics, including 3D vision, photogrammetry, and implicit representations.



Di Chang is currently a PhD student in the Thomas Lord Department of Computer Science and the Institute of Creative Technologies at University of Southern California. His current research interests lie in human-centric computer vision, 3D reconstruction, and generative models.



Quankai Gao is currently a PhD student in computer science department at University of Southern California. Before that, he earned his MSc degree in computer science at the University of Southern California. His research interests lie at computer vision and computer graphics, including rendering, 3D scene understanding, reconstruction and generation.



Junlin Han is currently a PhD student in the Department of Engineering at the University of Oxford. Before that, he obtained his BSc from the Australian National University. His research interests lie in computer vision and deep learning, including 3D reconstruction, generation, and editing.



Tong Zhang received the B.S. and M.S. degree from Beihang University, Beijing, China and New York University, New York, United States in 2011 and 2014 respectively, and he received the Ph.D. degree from the Australian National University, Canberra, Australia in 2020. He is working as a postdoctoral researcher at EPFL. He was awarded the ACCV 2016 Best Student Paper Honorable Mention and the CVPR 2020 Paper Award Nominee. His research interests include subspace clustering, representation learning and 3D vision.



Richard Hartley is a Distinguished Professor Emeritus at the Australian National University, where he has worked since 2001. He also led the Autonomous Systems and Sensor Technology Program at National ICT Australia (now Data61, CSIRO). He is an author of the book Multiple View Geometry in Computer Vision. He is a Fellow of the Royal Society, the Australian Academy of Science, the Australian Mathematical Society, and the IEEE.



Marc Pollefeys is a Prof. of Computer Science at ETH Zurich and Director of Science at Microsoft. He is best known for his work in 3D computer vision, but also for works on robotics, graphics, machine learning, and camera-based self-driving cars and drones. He received a M.Sc. and a PhD from the KU Leuven in Belgium in 1994 and 1999, respectively. He became an assistant professor at the University of North Carolina in Chapel Hill in 2002 and joined ETH Zurich as a full professor in 2007.

8 PRELIMINARIES

8.1 Plane Sweep

We take a toy example of three cameras viewing the same object as illustrated in Fig. 6. Plane sweep is then performed on the frustum of cam_x (termed as the reference camera) by creating a series of fronto-parallel hypothesized planes within a given range (typically $[d_{\min}, d_{\max}]$), with each plane corresponding to a depth value *w.r.t.* cam_x . Let's then examine two 3D points, M and M' , as examples of occupied and unoccupied positions on hypothesized planes, respectively. For M , all of the three cameras capture the identical point lying on the geometry surface with photometric consistency. In contrast, the photometric consistency of the observations of M' are poor, indicating that M' is an invalid hypothesis. To evaluate the similarity of the observations towards M and M' , the images of cam_1 and cam_2 (termed as the source cameras) are warped to each plane by homography. These warped images are then compared against the scaled images of cam_x . Depth hypotheses with high similarity measures are considered reliable.

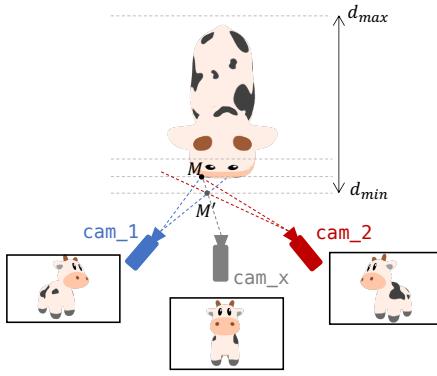


Fig. 6. Illustration of plane sweep algorithm [32]. To estimate the depth map for reference image (cam_x), neighboring source images (cam_1 , cam_2) are projected with homography to fronto-parallel planes of the reference view frustum.

Regarding selecting D depth hypotheses from the triangulated depth range $[d_{\min}, d_{\max}]$, there are two main schemes, namely the forward depth sampling and the inverse depth sampling. The naive forward sampling divides the depth range into $D - 1$ depth intervals with identical lengths. Given a depth index k , we have

$$d_k = d_{\min} + \frac{k}{D-1}(d_{\max} - d_{\min}), k = 0, \dots, D-1, \quad (23)$$

where the depth hypotheses distribute uniformly between the two ends. While for the inverse sampling scheme [35], we sample uniformly in the multiplicative inverse of d , such that

$$\frac{1}{d_k} = \frac{1}{d_{\max}} + \frac{k}{D-1}\left(\frac{1}{d_{\min}} - \frac{1}{d_{\max}}\right), k = 0, \dots, D-1. \quad (24)$$

In this way, the distribution of sampled depth values becomes sparser with d approaching to d_{\max} . When reconstructing unbounded outdoor scenes, where the depth range is rather large, the inverse sampling will be a reasonable choice since it samples more densely at the foreground. In addition, as revealed in [94], the inverse sampling leads to a

uniform sampling on the projected epipolar lines of source images.

8.2 Depth Fusion

For photometric consistency filtering, a per-pixel confidence is estimated to measure the confidence of depth estimation, *i.e.*, probability that the ground truth depth is within a small range near the estimation. Then a threshold can be set to filter depth values with low confidence. For geometric consistency filtering, the consistency of depth estimations are measured among multiple views. For a pixel \mathbf{p} in the reference view 0, we project it to pixel \mathbf{p}' in its i -th neighboring view through its depth prediction $d_0(\mathbf{p})$. After looking up the depth for \mathbf{p}' , $d_i(\mathbf{p}')$, we re-project \mathbf{p}' back to the reference view at pixel \mathbf{p}'' and look up its depth $D_0(\mathbf{p}'')$. We consider pixel \mathbf{p} and its depth as consistent to the i -th neighboring view, if the distances, in image space and depth, between the original estimate and its re-projection satisfy:

$$\xi_d = \|\mathbf{p} - \mathbf{p}''\|_2 \leq \tau_1, \quad (25)$$

$$\xi_p = \frac{\|d_0(\mathbf{p}'') - d_0(\mathbf{p})\|}{d_0(\mathbf{p})} \leq \tau_2, \quad (26)$$

where τ_1 and τ_2 are thresholds. The pixels are considered to be reliable estimations if they are consistent in at least N neighboring views. Recently, instead of using predefined thresholds for τ_1 and τ_2 , dynamic consistency checking [37] is proposed to dynamically aggregate geometric matching error among all views and improve the robustness of reconstruction. Specifically, the dynamic multi-view geometric consistency $C_{\text{geo}}(\mathbf{p})$ is computed as:

$$C_{\text{geo}}(\mathbf{p}) = \sum_i \exp(-(\xi_p + \lambda \xi_d)), \quad (27)$$

where λ is a weight to balance the reprojection error in two metrics. Then the outliers with $C_{\text{geo}}(\mathbf{p})$ smaller than a threshold are filtered.

8.3 Datasets and Benchmarks

Tab. 6 is a brief summary of commonly used public MVS datasets and benchmarks for training and evaluation.

ScanNet: ScanNet [26] is a large RGB-D dataset that contains 1613 indoor scenes with ground-truth camera poses, depth maps, surface reconstruction and semantic segmentation labels. Online MVS methods [25], [30], [34] mainly use ScanNet for training and testing.

7-Scenes: 7-Scenes dataset [179] is a RGB-D dataset captured in indoor scenes with a handheld Kinect RGB-D camera. Since it is relatively small, 7-Scenes is usually used to test the generalization performance of the models trained on ScanNet [26] without finetuning.

DTU: DTU dataset [21] is an object-centric MVS dataset collected under well-controlled laboratory conditions with known accurate camera trajectory. It contains 128 scans with 49 or 64 views under 7 different lighting conditions. Since DTU dataset officially provides scanned ground truth point clouds instead of depth maps, it is required to generate mesh models with surface reconstruction, *e.g.*, screened Poisson surface reconstruction algorithm [196], and then render depth maps [15] for training.

TABLE 6
An overview of commonly used datasets and benchmarks for learning-based Multi-View Stereo.

Dataset	Provided Ground Truth ¹				Synthetic	Online Benchmark	Evaluation Target
	Camera pose	Depth Map	Point Cloud	Mesh			
ScanNet [26]	✓	✓		✓			Depth Map / Mesh
7-Scenes [179]	✓	✓					Depth Map
DTU [21]	✓		✓				Point Cloud
Tanks and Temples [22]			✓			✓	Point Cloud
ETH3D [23]	✓		✓			✓	Point Cloud
BlendedMVS [24]	✓	✓			✓		Depth Map

¹ For datasets with online benchmark, the point cloud ground truth of test set is not released.

Tanks and Temples: Tanks and Temples [22] is a large-scale benchmark captured in more complex real indoor and outdoor scenarios. It is divided into intermediate and advanced sets. Different scenes have different scales, surface reflection and exposure conditions. It is usually used to test the generalization performance. Note that Tanks and Temples does not provide ground truth camera parameters, which are usually estimated with Structure-from-Motion methods such as COLMAP [17] or OpenMVG [58].

ETH3D: ETH3D benchmark [23] contains 25 large-scale scenes, including indoor and outdoor, with high-resolution RGB images. The scenes typically contain many low-textured regions and non-Lambertian surfaces, e.g., white walls and reflective floor. In addition, the images are usually sparse and have strong viewpoint variations and occlusions. Therefore, ETH3D is considered as a very challenging benchmark. Similar to Tanks and Temples, ETH3D is usually used to test the generalization performance.

BlendedMVS: BlendedMVS dataset [24] is a recently introduced large-scale synthetic dataset for MVS training that contains a variety of scenes, such as cities, sculptures and shoes. The dataset consists of over 17k high-resolution images rendered with reconstructed models and is split into 106 training scenes and 7 validation scenes. Since images are rendered through virtual cameras, the camera parameters are accurate enough for training.

8.4 Evaluation Metrics

8.4.1 2D Metrics

We list the commonly used metrics as follows: mean absolute depth error (Abs), mean absolute relative depth error (Abs Rel) and inlier ratio with threshold 1.25 ($\delta < 1.25$):

$$\begin{aligned} \text{Abs} &= \frac{1}{N} \sum_{\mathbf{p}} |d(\mathbf{p}) - \hat{d}(\mathbf{p})|, \\ \text{Abs Rel} &= \frac{1}{N} \sum_{\mathbf{p}} \frac{|d(\mathbf{p}) - \hat{d}(\mathbf{p})|}{d(\mathbf{p})}, \\ \text{Inlier Ratio} &= \frac{1}{N} \sum_{\mathbf{p}} \mathbb{1} \left[\frac{d(\mathbf{p})}{1.25} < \hat{d}(\mathbf{p}) < 1.25d(\mathbf{p}) \right], \end{aligned} \quad (28)$$

where $d(\mathbf{p})$ denotes the ground truth depth, $\hat{d}(\mathbf{p})$ denotes the estimation, N denotes the number of pixels with valid depth measurements and $\mathbb{1}[\cdot]$ denotes the indicator function.

8.4.2 3D Metrics

Precision/Accuracy: Precision/Accuracy measures the percentage of predicted points that can be matched to the ground truth point cloud. Considering a point \mathbf{P}_p in the predicted point cloud, it is considered to have a good match in the ground truth point cloud $\{\mathbf{P}_g\}$ if

$$\|\mathbf{P}_p - \arg \min_{\mathbf{P} \in \{\mathbf{P}_g\}} \|\mathbf{P} - \mathbf{P}_p\|_2\|_2 \leq \tau, \quad (29)$$

where τ is a scene-dependent threshold assigned by datasets, usually set to a large value for large-scale scenes. Note that in some datasets, instead of being measured by percentage [22], [23], precision/accuracy is measured by mean or median absolute distance [21].

Recall/Completeness: Recall/Completeness measures the percentage of ground truth points that can be matched to the predicted point cloud. For a point \mathbf{P}_g in the ground truth point cloud, it is considered a good match in the predicted point cloud $\{\mathbf{P}_p\}$ if

$$\|\mathbf{P}_g - \arg \min_{\mathbf{P} \in \{\mathbf{P}_p\}} \|\mathbf{P} - \mathbf{P}_g\|_2\|_2 \leq \tau. \quad (30)$$

Similar to precision/accuracy, recall/completeness can be measured by the percentage of points [22], [23] or measured by the absolute distance [21], similar to Chamfer distance.

F-Score: The two aforementioned metrics measure the accuracy and completeness of predicted point clouds. However, each of these metrics alone cannot present the overall performance since different MVS methods adopt different assumptions. A stronger assumption usually leads to higher accuracy but lower completeness. If only precision/accuracy is reported, it would favor MVS algorithms that only include estimated points of high certainty. On the other hand, if only recall/completeness is reported it would favor MVS algorithms that include everything, regardless of accuracy. Therefore, F-score, an integrated metric, is introduced [22], [23]. F-score is the harmonic mean of precision and recall. It is sensitive to extremely small values and tends to get more affected by smaller values, which means that F-score does not encourage imbalanced results. However, in most cases, F-score still suffers from unfairness due to the limitations of ground truth, e.g., sparse and incomplete point clouds may penalize for filling in the areas that are not present in the ground truth [46].

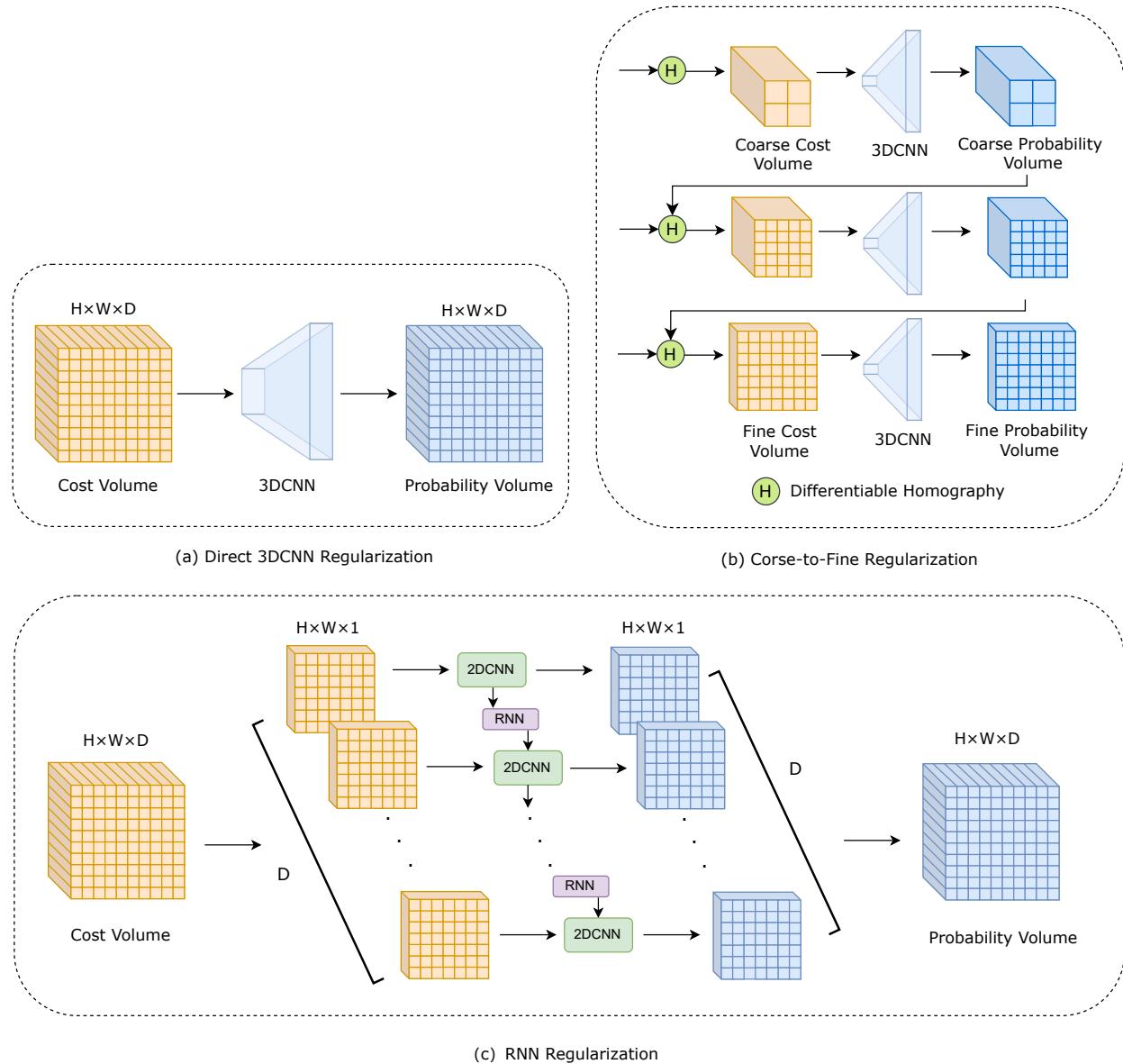


Fig. 7. Illustration of typical cost regularization schemes for offline MVS methods that use 4D cost volumes. (a) direct 3D CNN regularization [15], [35], [97], [98] applies a 3D CNN to aggregate contextual information; (b) coarse-to-fine regularization [31], [38], [75], [94] constructs multi-scale cost volumes based on coarse prediction and uses 3D CNN regularization on each scale; (c) RNN regularization [36], [37], [77] sequentially regularizes 2D slices of the cost volume to reduce memory consumption.

9 SUPERVISED METHOD WITH DEPTH ESTIMATION

9.1 Cost Volume Regularization

In Fig. 7, we visualize the three main categories to perform cost volume regularization for offline MVS methods: direct 3D CNN, coarse-to-fine and RNN.