Generalized Binary Search Network for Highly-Efficient Multi-View Stereo

Zhenxing Mi  Chang Di  Dan Xu
The Hong Kong University of Science and Technology
{zmiaa, dchangac}@connect.ust.hk, danxu@cse.ust.hk

Abstract

Multi-view Stereo (MVS) with known camera parameters is essentially a 1D search problem within a valid depth range. Recent deep learning-based MVS methods typically densely sample depth hypotheses in the depth range, and then construct prohibitively memory-consuming 3D cost volumes for depth prediction. Although coarse-to-fine sampling strategies alleviate this overhead issue to a certain extent, the efficiency of MVS is still an open challenge. In this work, we propose a novel method for highly efficient MVS that remarkably decreases the memory footprint, meanwhile clearly advancing state-of-the-art depth prediction performance. We investigate what a search strategy can be reasonably optimal for MVS taking into account of both efficiency and effectiveness. We first formulate MVS as a binary search problem, and accordingly propose a generalized binary search network for MVS. Specifically, in each step, the depth range is split into 2 bins with extra 1 error tolerance bin on both sides. A classification is performed to identify which bin contains the true depth. We also design three mechanisms to respectively handle classification errors, deal with out-of-range samples and decrease the training memory. The new formulation makes our method only sample a very small number of depth hypotheses in each step, which is highly memory efficient, and also greatly facilitates quick training convergence. Experiments on competitive benchmarks show that our method achieves state-of-the-art accuracy with much less memory. Particularly, our method obtains an overall score of 0.289 on DTU dataset and tops the first place on challenging Tanks and Temples advanced dataset among all the learning-based methods. Our code will be released at https://github.com/MiZhenxing/GBi-Net.

1. Introduction

Multi-view Stereo (MVS) is a long-standing and fundamental topic in computer vision, which aims to reconstruct 3D geometry of a scene from a set of overlapping images [9, 10, 26, 32, 35]. With known camera parameters, MVS matches pixels across images to compute dense correspondences and recover 3D points, which is essentially a 1D search problem [8]. A depth map is widely used as 3D representation due to its regular format. To overcome the issue of coarse matching in previous purely geometry-based methods, recent learning-based MVS methods [14, 21, 41, 42] designed deep networks for dense depth prediction to significantly advance traditional pipelines. For instance, MVSNet [41] and RMVSNet [42] propose to construct 3D cost volumes from 2D image features with dense depth hypotheses. A 3D cost volume is a 5D tensor and is typically regularized by a 3D Convolutional Neural Network (CNN) or a Recurrent Neural Network (RNN) for depth prediction.

The importance of 3D cost volume regularization for accurate depth prediction has been confirmed by other works [4, 5, 11]. However, a severe problem is that 3D cost volumes are highly memory-consuming. Existing works made significant efforts to address this issue via decreasing the resolution of feature maps [41], using a coarse-to-fine strategy that gradually increases resolution of feature maps while decreasing the depth hypothesis number [4, 5, 11], and removing expensive 3D CNN or RNN [33, 40]. Although the memory can be alleviated to some extent, relatively lower accuracy is commonly observed. The size of 3D cost volume, specifically the depth hypothesis number,
plays a dominant role in causing a large memory footprint.

Due to the significance of 3D cost volumes in both model efficiency and effectiveness, a critical question naturally arises: what is a minimum volume size to secure satisfactory accuracy while maintaining as small as possible the memory overhead? In this work, we investigate this question by exploring from a perspective of discrete search strategies, to identify a minimal depth hypotheses number, a key factor in 3D cost volumes. As shown in Fig. 1b, the vanilla MVSNet [41] can be seen as a dense search method that checks all depth hypotheses similar to a linear search in a parallel manner. The coarse-to-fine methods [5, 11] perform a multi-granularity search, which starts from a coarse level and gradually refines the prediction. However, these two types of methods both consider dense search in each stage. We argue that the dense search does not necessarily guarantee better accuracy due to a much larger prediction space and significantly increases model complexity, leading to higher optimization difficulty in model training.

To explore a reasonably optimal search strategy, we first formulate MVS as a binary search problem, which can remarkably reduce the cost volume size to an extremely low bound. It performs comparisons and eliminates half of the search space in each stage (see Fig. 1b), and can convergence quickly to a fine granularity within logarithmic stages. In contrast to regression-based methods, which directly sample depth values from the depth range, we first divide the depth range into 2 bins. In our design, the ‘comparisons’ by the network is to determine which bin contains the true depth value via performing a binary discrete classification. We use center points of the two bins to represent them and construct 3D cost volumes. The binary search offers superior efficiency, while it brings an issue of out-of-bin error. If in some stage the prediction selects a wrong bin, the error will accumulate in later search stages causing unstable optimization gradients and relatively low accuracy.

To tackle this issue, we further design three effective mechanisms, and accordingly propose a generalized binary search deep network, termed as GBi-Net, for highly efficient MVS. The first mechanism is that we pad two error tolerance bins on the two sides to reduce the prediction error of out of bins. The second mechanism is for training. If the network generates an error of prediction out of bins for a pixel at a search stage, we stop the forward pass of this pixel in the next stage, and the gradients at this stage are not used to update the network. Extensive experiments show that the proposed GBi-Net can largely decrease the size of 3D cost volumes for a significantly efficient network, and more importantly, without any trade-off on the depth prediction performance. The third is efficient gradient updating. It updates the network parameters immediately at each search stage without accumulating across different stages as most works do. It can largely reduce the training memory while maintaining the performance. Our method achieves state-of-the-art performance on different competitive datasets including DTU [2] and Tanks & Temples [17]. Notably, on DTU, we achieve an overall score of 0.289 (lower is better), remarkably improving the previous best performing method, and also obtain a memory efficiency improvement of 48.0% compared to UCSNet [5] and 54.1% compared to CasMVSNet [11] (see Fig. 1a).

In summary, our contribution is three-fold:

- We investigate efficient MVS from a perspective of search strategies, and propose a discrete binary search method for MVS (Bi-Net) which can vastly decrease the memory usage of 3D cost volumes.
- We design a highly-efficient generalized binary search network (GBi-Net) via further designing three mechanisms (i.e. padding error tolerance bins, gradients masking, and efficient gradient updating) with the binary search to avoid error accumulation of false network predictions and improve efficiency.
- We evaluate our method on several challenging MVS datasets and significantly advance existing state-of-the-art methods in terms of both the depth prediction accuracy and the memory efficiency.

2. Related Work

We review the most related works in the literature from two aspects, i.e. traditional MVS and learning-based MVS.

**Traditional Multi-View Stereo.** In 3D reconstruction, various 3D representations are used, such as volumetric representation [18, 27], point cloud [9, 19], mesh [6, 16, 29, 30] and depth map [3, 10, 26]. In MVS [23, 26, 35], depth maps have shown advantages in robustness, efficiency. They estimate a depth map of each reference image and fuse them into one 3D point cloud. For instance, COLMAP [26] simultaneously estimates pixel-wise view selection, depth, and surface normal. ACMM [35] leverages a multi-hypothesis joint voting scheme for view selection from different candidates. These existing methods perform modeling of occlusion, illumination across neighboring views for depth estimation. Although stable results can be achieved, high matching noise and poor correspondence localization in complex scenes are still severe limitations. Thus, our method mainly focuses on developing a deep learning-based MVS pipeline to advance the estimation performance.

**Learning-based Multi-View Stereo.** Deep learning-based MVS methods [13, 15, 34, 41] recently have achieved remarkable performance. Deep learning-based methods usually utilize Deep CNNs to estimate a dense depth map. Recently, 3D cost volumes have been widely used in MVS [33, 37, 41, 45]. As a pioneering method, MVSNet [41] constructs the 3D cost volume from feature warping and regularizes the cost volume with 3D CNNs for depth regression.
3. The Proposed Approach

In this section, we introduce the detailed structure of the proposed Generalized Binary Search Network (GBi-Net) for highly-efficient MVS. The overall framework is depicted in Fig. 2. It mainly consists of two parts, i.e. a 2D CNN network for learning visual image representations, and the generalized binary search network for iterative depth estimation. The GBi-Net contains K search stages. In each search stage, we first compute 3D cost volumes by differentiable warping between the reference and source feature map in a specific corresponding scale. Then 3D cost volumes are regularized by 3D CNNs for depth label prediction. The Generalized Binary Search is responsible for initializing and updating depth hypotheses according to the predicted labels iteratively. In every two stages, the networks deal with the same scale of feature maps, and the network parameters are shared. Finally, one-hot labels for training the whole network are computed from ground-truth depth maps. In the next, we first introduce the 2D image encoder in Sec. 3.1 and the 3D cost volume regularization in Sec. 3.2. Then, we elaborate on details about our proposed Binary Search for MVS and Generalized Binary Search for MVS in Sec. 3.3 and Sec. 3.4, respectively. Finally, we present the overall network optimization in Sec. 3.5.

3.1. Image Encoding

The input consists of N images \( \{I_i\}_{i=0}^{N-1} \). \( I_0 \) is a reference image and \( \{I_i\}_{i=1}^{N-1} \) is a set of \( N-1 \) source images. We use Feature Pyramid Network (FPN) [20] as an image encoder to learn generic representation for the images with shared network parameters. From FPN, we obtain a pyramid of feature maps with 4 different scales. To have more powerful representations of the images, one deformable convolutional network (DCN) [7] layer is used as output layer for each scale to more effectively capture scene contexts that are very beneficial for the MVS task.

The main problem of vanilla MVSNet is the large memory consumption of the 3D cost volume. Recurrent MVSNet architectures [34, 38, 42] leverage recurrent networks to regularize cost volumes, which can decrease the memory usage to some extent in the testing phase. However, the major overhead from the 3D cost volumes is not specifically addressed by these existing methods.

To reconstruct high-resolution depth maps meanwhile obtaining a memory-efficient cost volume, cascade-based pipelines are proposed [5, 11, 39], considering a coarse-to-fine dense search strategy to gradually refine the depths. For instance, CasMVSNet [11] utilizes coarse feature maps and depth hypotheses in the first stage for coarse depth prediction, and then upsamples depth maps and narrows the depth range for fine-grained prediction in the next stage. Patchmatchnet [33] learns adaptive propagation and evaluation for depth hypotheses. It removes heavy regularization of 3D cost volumes to achieve an efficient model while it makes a significant trade-off between efficiency and accuracy.

However, these existing works still consider a dense search in each regression stage. The memory overhead on the expensive 3D cost volume is clearly not optimized, while the proposed method targets highly efficient MVS with the designed binary search network, which largely advances the model efficiency, and more importantly, without sacrificing any depth prediction performance.
3.2. Cost Volume Regularization

The construction of 3D cost volumes is a critical step for deep learning-based MVS [41]. We present details about the cost volume construction and regularization for the proposed generalized binary search network. Given $D$ depth hypotheses at the $k$-th search stage, i.e., $\{d_{k,j} | j = 1, \ldots, D\}$, a pixel-wise dense 3D cost volume can be built by differentiable warping on the learned image feature maps [33, 41]. To simplify the description, we ignore the stage index $k$ in the following formulation.

The input of MVS consists of relative camera rotation $R_{F_0 \rightarrow F_i}$ and translation $t_{F_0 \rightarrow F_i}$ from a reference feature map $F_0$ to a source feature map $F_i$. Their corresponding camera intrinsics $K_0, K_i$ are also known. We first construct a set of 2-view cost volumes $\{V_i\}_{i=1}^{N-1}$ from the $N - 1$ source image feature maps by differentiable warping and group-wise correlation [12, 33, 37]. Let $p$ be a pixel in $I_0$, $p'$ be the warped pixel of $p$ in the source image $I_i$ by the $j$-th depth hypothesis, i.e., $d_j$. Then $p'$ can be computed by:

$$p' = K_i \cdot (R_{F_0 \rightarrow F_i} \cdot K_0^{-1} \cdot p + t_{F_0 \rightarrow F_i}),$$  

where the feature maps $F_0$ and $F_i$ all have a channel dimension of $N_c$. Following [12], we divide the channels of the feature maps into $N_g$ groups along the channel dimension, and each feature group therefore has $N_c/N_g$ channels. Let $F_i^g$ be the $g$-th feature group of $F_i$. Then we can compute the $i$-th cost volume $V_i$ from $F_i$ as follows:

$$V_i(j, p, g) = \frac{N_g}{N_c} \langle F_i^g(p), F_i^g(p') \rangle.$$  

Where $\langle \cdot, \cdot \rangle$ denotes a correlation calculation by an inner product operation. The group-wise correlation allows us to more efficiently construct a full cost volume. The initial bin width in the proposed binary search is $2$ equal bins, i.e., $\{B_{k,j} | j = 1, 2\}$ with $B_{k,j}$ denoting a bin. The bin width of $B_{1,j}$ in the first stage is $R/2$. As we cannot directly use discrete bins for warping feature maps, we sample center points of the 2 bins to represent the depth hypotheses of bins, and then construct the cost volume and perform label prediction for the 2 bins. Let the three edges from left to right of the 2 bins be $\{e_{k,j} | m = 1, 2, 3\}$. Then, the two edges of bin $B_{k,j}$ are $e_{k,j}$ and $e_{k,j+1}$. For instance, $e_{k,1}$ and $e_{k,2}$ are edges of $B_{k,1}$. Then the depth hypothesis $d_{k,j}$ for the 2 bins can be computed as follows:

$$d_{k,j} = \frac{e_{k,j} + e_{k,j+1}}{2}, \quad j = 1, 2.$$  

The predicted label of a depth hypothesis indicates whether the true depth value is in the corresponding bin. In the $k$-th search stage, after the network outputs the probability volume $P$, we apply an $\text{argmax}(\cdot)$ operation along the $D$ dimension of $P$, which returns the label $j$ indicates that the true depth value is in the bin $B_{k,j}$. The new 2 bins at the $(k+1)$-th search stage can be further generated by dividing $B_{k,j}$ into two equal-width bins $B_{k+1,1}$ and $B_{k+1,2}$, and the corresponding three edges at this stage can be defined as:

$$e_{k+1,1} = e_{k,j}; e_{k+1,2} = \frac{e_{k,j} + e_{k,j+1}}{2}; e_{k+1,3} = e_{k,j+1}.$$  

Then new depth hypotheses are sampled from the center points of bins $B_{k+1,1}$ and $B_{k+1,2}$ for the $(k+1)$-th stage. The initial bin width in the proposed binary search is $R/2$ and in the $k$-th stage, the bin width is $R/2^k$.

With our proposed binary search strategy, the depth dimension of the 3D cost volume can be decreased to $2$, which pushes the cost volume size to an extremely low bound, and the memory footprint is dramatically decreased. In our experiments, the Binary Search for MVS achieves satisfactory results, outperforming several existing competitive methods, and the memory overhead of the whole MVS network becomes dominated by the 2D image encoder, no longer by the 3D cost volumes. However, as discussed in the Introduction, the issue of the network classification error can cause unstable optimization and degraded accuracy.

3.4. Generalized Binary Search for MVS

To handle the error accumulation and the training issue in the proposed Binary Search for MVS, we extend it
Stage 1
Stage 2
Stage 3

Figure 3. Illustration of Generalized Binary Search. We subdivide the selected bin into two bins according to the label. Then we pad Error Tolerance Bins (ETB) on both sides. We check the gradient mask of this pixel. Only if it is valid, the loss could participate in back-propagation.

There is a Generalized Binary Search for MVS. Specifically, we further design three effective mechanisms which are error tolerance bins, gradient-masked optimization and efficient gradient updating mechanism, making substantial improvement over the Binary Search method.

**Error Tolerance Bins.** After obtaining the selected bin $B_{k, j}$ in the $k$-th search stage, we first divide it into two new bins $B_{k+1, 1}$ and $B_{k+1, 2}$ for the next $(k + 1)$-th stage, as shown in Fig. 3. To make the network have a certain capability of tolerating prediction errors, we propose to respectively add one small bins on the left side of $B_{k+1, 1}$ and on the right side of $B_{k+1, 2}$. This process is termed as Error Tolerance Bins (ETB). More formally, given $D$ ($D$ is an even number and small enough) as the final number of bins, we pad $(D - 2)/2$ more bins to the two sides of the original two bins. After the padding, $D$ new bins, i.e. $\{B_{k+1, j} | j = 1, \ldots, D\}$, are obtained, as well as their corresponding bin edges, i.e. $\{e_{k+1,m} | m = 1, \ldots, D + 1\}$.

We still sample the center points as the depth hypotheses $\{d_{k+1,j} | j = 1, \ldots, D\}$ from these bins with Eq. 4. The error tolerance bins extend the sampling of depth hypotheses to a range out of the two original bins in the binary search, thus enabling the network to correct the predictions and to reduce error accumulation to some extent. Since the depth hypotheses number is now $D$, we also change the initialization of depth hypotheses in the first stage. As the initial depth range $R$ in split into $D$ bins, the initial bin width is $R/D$ and in the $k$-th stage, the bin width is $R/(D \times 2^{k-1})$.

In our network implementation, we pad only 1 ETB on both sides. This leads to a depth hypothesis number of 4, i.e. $D = 4$. In the experiments, we observe dramatically improved depth prediction accuracy while notably, the memory consumption can be the same level as the original binary search, as the memory is still dominated by the 2D image encoder. Fig. 3 shows a real example of our GBi-Net. The hypothesis number $D$ is set to $4$. With the error tolerance bins, the network can predict a correct label of 4 when the true depth is in $B_{3,4}$ at the 3-th search stage, while the original binary search fails.

**Gradient-Masked Optimization.** The proposed GBi-Net is trained in a supervised manner. The ground-truth labels are generated from the ground-truth depth map. In the $k$-th search stage, after we obtain the bins, we calculate which bin is occupied by the ground-truth depth value. Then we can convert the ground truth depth map into a ground truth occupancy volume $G$ with one-hot encoding, which is further used for loss calculation. One problem in the iterative search is that the ground-truth depth values for some pixels may be out of the $D$ bins. In this situation, no valid labels exist and the losses cannot be computed. This is a critical problem in network optimization. The coarse-to-fine methods typically leverage a continuous regression loss, while existing MVS methods with a discrete classification loss [42] widely employ dense space discretization.

In our GBiNet, a designed mechanism to this problem is computing a mask map for each stage, based on the bins and ground-truth depth maps. If the ground-truth depth of a pixel is in the current bins, the pixel is considered as valid. Let the ground-truth depth for a pixel be $d_{gt}$ and the current bin edges be $\{e_m | m = 1, \ldots, D + 1\}$, omitting the stage index for simplicity. Then the pixel is valid only if:

$$e_1 \leq d_{gt} < e_{D+1}. \quad (6)$$

Only the loss gradients from the valid pixels are used to update the parameters in the network. The gradients from all the invalid pixels are not accumulated. With this process, we can train both Bi-Net and GBi-net successfully, as clearly confirmed by our experimental results. The gradient-masked optimization is similar to the popular self-paced learning [25], in which at the very beginning, the network only involves easy samples (i.e. easy pixels) in training, while with the optimization proceeds, the network can predict more accurate labels for hard pixels, and most pixels will eventually participate in the learning process. As can be observed in Fig. 6c in the experiments, a large portion of pixels falls into the the current bins in our GBi-Net.

**3.5. Network Optimization**

**Loss Function.** Our loss function is a standard cross-entropy loss that applies on the probability volume $P$ and a ground truth occupancy volume $G$. A set of the valid pixels $\Omega_q$ is first obtained by the valid mask map and then a mean loss of all valid pixels is computed as follows:

$$Loss = \sum_{q \in \Omega_q} \sum_{j=1}^{D} -G(j, q) \log P(j, q) \quad (7)$$

**Memory-efficient Training.** MVS methods with multiple stages [5, 11] typically average the losses from all the stages.
and back-propagate the gradients together. Nevertheless, this training strategy consumes significant memory because of the gradients accumulation across different stages. In our GBi-Net, we train our network in a more memory-efficient way. Specifically, we compute the loss and back-propagate the gradients immediately after each stage. The gradients are not accumulated across stages, and thus the maximum memory overhead does not exceed the stage with the largest scale. To make the training with multiple stages more stable, we first set the maximum number of search stages as 2, and gradually increase it as the epoch number increases.

4. Experiments

4.1. Datasets

The DTU dataset [2] is an indoor dataset with multi-view images and camera poses. We follow MVSNet [41] for dividing training and testing sets. There are 27097 training samples in total. The BlendedMVS dataset [43] is a large-scale dataset with indoor and outdoor scenes. Following [22, 34, 45], we only use this dataset for training. There are 16904 training samples in total. Tanks and Temples [17] is a large-scale dataset with various outdoor scenes. It contains Intermediate subset and Advanced subset. The evaluation on this benchmark is conducted online by submitting generated point clouds to the official website.

4.2. Implementation Details

Training Details. The proposed GBi-Net is trained on the DTU dataset for DTU benchmarking and trained on BlendedMVS dataset for Tanks and Temples benchmarking, following [22, 34, 45]. We use the high-resolution DTU data provided by the open source code of MVSNet [41]. The original image size is $1200 \times 1600$. We first crop the input images into $1024 \times 1280$ following MVSNet [41]. Different from MVSNet [41] that directly downscale the image to $512 \times 640$, we propose an online random cropping data augmentation. We randomly crop images of $512 \times 640$ from images of $1024 \times 1280$. The motivation is that cropping smaller images from larger images could help to learn better features for larger image scales without increasing the training overhead. When training on BlendedMVS dataset, we use the original resolution of $576 \times 768$. For all the training, $N=5$ input images are used, i.e. 1 reference image and 4 source images. We adopt the robust training strategy proposed in Patchmatchnet [33] for better learning of pixel-wise weights. The maximum stage number is set to 8. For every 2 stages, we share the same feature map scale and the 3D-CNN network parameters. The whole network is optimized by Adam optimizer in Pytorch for 16 epochs with an initial learning rate of 0.0001, which is down-scaled by a factor of 2 after 10, 12, and 14 epochs. The total training batch size is 4 on two NVIDIA RTX 3090 GPUs.

Testing Details. The model trained on DTU training set is used for testing on DTU testing set. The input image number $N$ is set to 5, each with a resolution of $1152 \times 1600$. It takes 0.61 seconds for each testing sample. The model trained on BlendedMVS dataset is used for testing on Tanks and Temples dataset. The image sizes are set to $1024 \times 1920$ or $1024 \times 2048$ to make the images divisible by 64. The input image number $N$ is set to 7. All the testings are conducted on an NVIDIA RTX 3090 GPU. We then filter
Table 3. Point cloud evaluation results on the Advanced and Intermediate subsets of Tanks and Temples dataset [17]. Higher scores are better. The Mean is the average score of all scenes.

<table>
<thead>
<tr>
<th>Method</th>
<th>Advanced</th>
<th>Intermediate</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVSNet [41]</td>
<td>43.48</td>
<td>59.99</td>
</tr>
<tr>
<td>Point-MVSNet [4]</td>
<td>48.27</td>
<td>61.79</td>
</tr>
<tr>
<td>UCSNet [5]</td>
<td>54.83</td>
<td>76.09</td>
</tr>
<tr>
<td>CasMVSNet [11]</td>
<td>56.42</td>
<td>76.36</td>
</tr>
<tr>
<td>PatchmatchNet [33]</td>
<td>53.15</td>
<td>66.99</td>
</tr>
<tr>
<td>BP-MVSNet [45]</td>
<td>57.60</td>
<td>77.31</td>
</tr>
<tr>
<td>Vis-MVSNet [45]</td>
<td>60.03</td>
<td>77.40</td>
</tr>
<tr>
<td>AA-RMVSNet [34]</td>
<td>61.51</td>
<td>77.77</td>
</tr>
<tr>
<td>EPP-MVSNet [22]</td>
<td>61.68</td>
<td>77.86</td>
</tr>
</tbody>
</table>

| Bi-Net (ours)         | 53.41    | 74.93 | 54.37 | 45.09 | 51.86 | 49.09 | 49.56 | 55.76 | 46.67 |
| GBi-Net (ours)        | 47.69    | 58.26 | 56.00 | 51.54 | 56.11 | 56.11 | 56.11 | 56.11 | 47.89 |

and fuse depth maps of a scene into one point cloud, details in the supplemental file. Fig. 4 and Fig. 5 are visualizations of depth maps and point cloud comparison of our method.

4.3. Benchmark Performance

Overall Evaluation on DTU Dataset. We evaluate the results on the DTU testing set by two types of metrics. The first type of metric evaluates point clouds using official evaluation scripts of DTU [2]. It compares the distance between ground-truth point clouds and the produced point clouds. The state-of-the-art comparison results are shown in Table 1. The method CasMVSNet-4 is based on CasMVSNet by changing its sampling number to 4 and stage number to 8. Our two models, GBi-Net and GBi-Net* both significantly improved the best performance on the completeness and the most important overall score (lower is better for both metrics). Our best model improves the overall score from 0.344 of UCSNet [5] to 0.289, while the memory is reduced by 48%. Note that our Bi-Net, i.e., the proposed Binary Search Network can also achieve comparable results to other dense search methods, clearly showing its effectiveness. The second type of metric directly evaluates the accuracy of the predicted depth maps. The depth ranges of DTU dataset are all 510 millimeters. Thus, we compute the depth accuracy, which counts the percentage of pixels whose absolute depth errors are less than a threshold, and 4 thresholds are considered in the evaluation (i.e., 0.125, 0.25, 0.5, 1, with millimeters as a unit). Compared to the depth range of 510 mm, these thresholds are extremely tight and challenging. The results of this type of metric are shown in Table 2. Our GBi-Net also obtains the best results on all the thresholds. The quality of depth maps also explains our best performance on point cloud evaluation.

Overall Evaluation on Tanks and Temples. We train the proposed Bi-Net and GBi-Net on BlendedMVS [43], and testing on Tanks and Temples dataset. We compare our method to state-of-the-art methods. Table 3 shows results on both the Advanced subset and the Intermediate subset. Our GBi-Net achieves the best mean score of 37.32 (higher is better) on Advanced subset compared to all the competitors, and it performs the best on 4 out of the overall 6 scenes. Note that the Advanced subset contains different large-scale outdoor scenes. The results can fully confirm the effectiveness of our method. Table 3 also shows the evaluation results on the Intermediate subset. Our GBi-Net obtains highly comparable results to the state-of-the-art. Notably, with significantly less memory, our mean score is only 0.09 lower than AA-RMVSNet [34] and 0.26 lower than EPP-MVSNet [22]. Moreover, we also obtain state-of-the-art scores on the Family and Francis scenes. Our binary search model Bi-Net also achieves satisfactory performance on both subsets. Our evaluation results on the leaderboard [1] are named as Bi-Net and GBi-Net.

Memory Efficiency Comparison. We compare the memory overhead with several previous best-performing learning-based MVS methods [5, 11, 33, 34, 41, 44, 45] on the DTU testing set. The memory usage evaluation is conducted with an image size of 1152 × 1600. We use pytorch functions¹ to measure the peak allocated memory usage of all the methods. Fig. 1a shows a comparison of the methods regarding memory usage and reconstruction error. Our GBi-Net shows a great improvement in the reconstruction quality while using much less memory. More specifically, the memory footprint is reduced by 77.5% compared to MVSNet [41], by 54.1% compared to CasMVSNet [11], by 82.4% to AA-RMVSNet [34], and by 55.9% to Vis-MVSNet [45]. Although the memory of our method is slightly 479 MB larger than Patchmatchnet [33], shown in Table 1 and 2, our method significantly outperforms it in both the point cloud (0.289 vs. 0.352) and depth map (12.77 vs. 8.113) evaluation by a large margin.

4.4. Model Analysis

Effect of Different Search Strategies. We first conduct a direct comparison on different search strategies as shown in Table 4, including dense linear search by regression (i.e., Dense LS), dense coarse-to-fine search by regression

¹ max_memory_allocated and reset_peak_memory_stats
(i.e. Dense C2F), and our Bi-Net and GBi-Net search by discrete classification. In this comparison, Bi-Net and GBi-Net are trained without using the random cropping data augmentation described in Sec. 4.2. As we can observe from Table 4, our binary search networks achieve significantly better results than Dense LS and Dense C2F on both the depth performance and the memory footprint, fully confirming the effectiveness of the proposed methods.

**Effect of Stage Number.** We analyze the influence of the number of search stages in our method. We test our GBi-Net model on DTU dataset with a maximum stage number of 9. We compare the reconstruction results of Stage 6, 7, 8, 9 with both the point cloud evaluation metrics and the depth map evaluation metrics. As shown in Fig. 6a and Fig. 6b, the reconstruction results improve quickly from Stage 6 to 8 and then convergence, which indicates that our model can converge with a reasonably small stage number.

Table 5. Evaluation on the number of Error Tolerance Bin (ETB), considering the performance of point cloud, depth map, and the memory usage on DTU. 0, 1 and 2 ETBs model are very close. All these results reveal that 1 ETB in our GBi-Net is already sufficient to achieve a good balance between accuracy and memory usage. More importantly, further increasing ETBs may make the model more complex and harder to optimize.

**Effect of memory-efficient Training.** We train a model without our Memory-efficient Training strategy (see Sec. 3.5) on the DTU dataset. This model averages the gradients accumulated from all the different stages and back-propagates them, which is a widely performed training scheme in MVS methods. As shown in Table 6, these two models obtain very similar results on the depth performance, while with the proposed Memory-efficient Training, the memory consumption is largely reduced by 57.1%.

**5. Conclusion**

In this paper, we first presented a binary search network (Bi-Net) design for MVS to significantly reduce the memory footprint of 3D cost volumes. Based on this design, we further proposed a generalized binary search network (GBi-Net) containing three effective mechanisms, i.e. error tolerance bins, gradients masking, and efficient gradient updating. The GBi-Net can greatly improve the accuracy while maintaining the same memory usage as the Bi-Net. Experiments on challenging datasets also showed state-of-the-art depth prediction accuracy, and remarkable memory efficiency of the proposed methods.

**Acknowledgements**

This research is supported in part by the Early Career Scheme of the Research Grants Council (RGC) of the Hong Kong SAR under grant No. 26202321 and HKUST Startup Fund No. R9253.
References


[38] Jianfeng Yan, Zizhuang Wei, Hongwei Yi, Mingyu Ding, Runze Zhang, Yisong Chen, Guoping Wang, and Yu-Wing Tai. Dense hybrid recurrent multi-view stereo net with dynamic consistency checking. In ECCV, 2020. 3


[41] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In ECCV, 2018. 1, 2, 4, 6, 7


