Learning Patch Reconstructability for Accelerating Multi-View Stereo

Alex Poms
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA
apoms@cs.cmu.edu

Chenglei Wu, Shoou-I Yu, Yaser Sheikh
Facebook Reality Labs, Pittsburgh
4420 Bayard Street, Pittsburgh, PA
{chenglei.wu,shoou-i.yu,yaser.sheikh}@fb.com

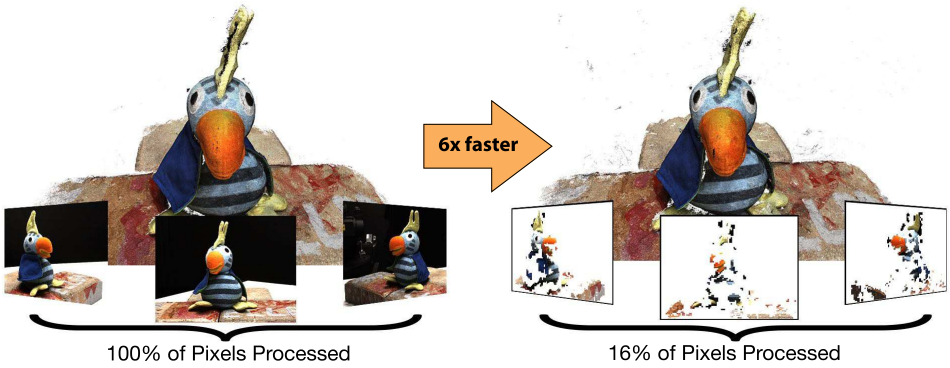

Figure 1: Left: A reconstruction performed using all pixels from the input. Right: Our method produces a similar reconstruction using a fraction of the pixels (white regions in the input images are unused).

Abstract

We present an approach to accelerate multi-view stereo (MVS) by prioritizing computation on image patches that are likely to produce accurate 3D surface reconstructions. Our key insight is that the accuracy of the surface reconstruction from a given image patch can be predicted significantly faster than performing the actual stereo matching. The intuition is that non-specular, fronto-parallel, in-focus patches are more likely to produce accurate surface reconstructions than highly specular, slanted, blurry patches — and that these properties can be reliably predicted from the image itself. By prioritizing stereo matching on a subset of patches that are highly reconstructable and also cover the 3D surface, we are able to accelerate MVS with minimal reduction in accuracy and completeness. To predict the reconstructability score of an image patch from a single view, we train an image-to-reconstructability neural network: the I2RNet. This reconstructability score enables us to efficiently identify image patches that are likely to provide the most accurate surface estimates before performing stereo matching. We demonstrate that the I2RNet, when trained on the ScanNet dataset, generalizes to the DTU and Tanks & Temples MVS datasets. By using our I2RNet with an existing MVS implementation, we show that our method can achieve more than a 30× speed-up over the baseline with only an minimal loss in completeness.

1. Introduction

Using a large number of calibrated high-resolution RGB images, very high quality surface geometry can be reconstructed using Multi-View Stereo (MVS) [31, 3, 19]. The high-level pipeline used by many state-of-the-art MVS methods first estimates the surface, i.e., the depth and normal, for each pixel in each view and then fuses estimates from all views together to create the final surface reconstruction. To estimate the depth and normal of a pixel (the reference pixel), MVS selects a window around the reference pixel, which we call the reference patch, and attempts to find a matching patch in each neighboring image. Once the reference patch has been matched, the match combined with the known geometric calibration of the views enables the MVS algorithm to compute the depth and normal of the 3D point.

Unfortunately, if this matching process is performed for every pixel in the image against every neighboring image, the running time scales with $\mathcal{O}(v^2p)$ where v is the number of images and p is the number of pixels per image, since every pixel in every image (vp) must be matched against every other image (v). This time complexity means that as we add more images with higher resolution, the time required can become prohibitively long. In order to leverage the increasing resolution and proliferation of cameras, it is critical that we develop *scalable* MVS algorithms that can still produce highly accurate surfaces while avoiding performing matching for every pixel against every image.

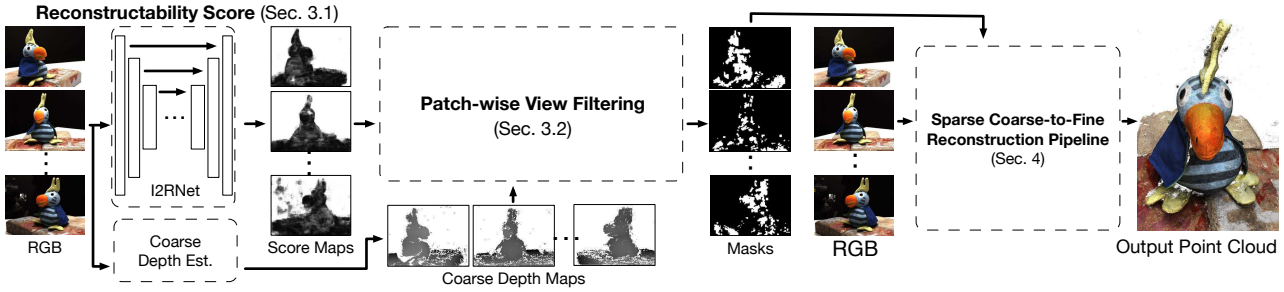

Figure 2: **Method overview:** Given a scan, we first run a neural network (I2RNet) to predict the reconstructability score of each image patch. Then, using these scores and coarse depth maps, the patch-wise view filtering framework will identify a mask for each image indicating where stereo matching should be run. Finally, a coarse-to-fine reconstruction pipeline takes images and their masks as input and generates the final point cloud.

Existing methods have attempted to scale MVS by using two main techniques: *view filtering*, which only estimates depth and normals for a subset of images [7], and *neighbor selection*, which only looks at a limited set of neighboring views when matching a patch instead of all of them [15, 33, 28]. The challenge of employing these techniques is ensuring that they do not significantly impact the accuracy or completeness of the reconstruction. For example, view filtering has been exploited by discarding entire images which redundantly view the same area and selecting the smallest set of images that still cover the surface being reconstructed [7]. However, filtering an entire view based on redundancy might not be perfectly aligned with our final goal: high quality 3D reconstructions, which we have observed are highly dependent on the "reconstructability" of *local patches* of an image. For example, diffuse and in-focus patches usually produce more accurate surface estimates than specular and blurry patches. Since each image can contain a mixture of reconstructable and non-reconstructable patches, coarsely filtering by entire images may discard image patches which would produce highly accurate surfaces.

Instead of image-wise view filtering, we propose a method to perform *patch-wise* view filtering, which selects patches that are more likely to produce high quality surface reconstructions. Specifically, we first learn to identify highly reconstructable image patches directly from the image content. Then, we identify the top N most reconstructable image patches corresponding to each 3D surface region we want to reconstruct and only compute surface estimates from those patches. Since we choose image patches which provide highly accurate surface estimates, we process only a fraction of the total pixels across a set of images (on the order of 2-16%) and still produce highly accurate and complete reconstructions, as shown by Figure 1. There were several contributions required to make this feasible:

Learning Patch Reconstructability: Our key insight is that the accuracy of the surface reconstruction from a given image patch can be predicted significantly faster than performing the actual stereo matching. The prediction is performed by a fully convolutional deep neural network. Then, when given an unseen multi-view scan, the deep network takes each image as input and regresses for each region its *reconstructability*, which serves as a proxy for the accuracy of the surface estimate that would be produced by the patch. For a given patch, its ground truth reconstructability score is computed as the difference between the depth computed from a single view and the ground truth depth, which if absent can be substituted by the result of a high quality MVS algorithm. In our evaluations, we show that our deep regressor, which is trained on a large database of scans (ScanNet [5]), can generalize to other MVS datasets, such as DTU [19, 1] and Tanks and Temples [22]. This demonstrates that the reconstructability of a patch can be estimated to some extent *based on a single image alone*. In terms of run time, the prediction by the deep network for an entire image takes on the order of 100 milliseconds, while the actual matching might take tens of seconds, thus demonstrating that the accuracy of surface reconstruction can be predicted significantly faster than actual stereo matching.

Patch Filtering: Given the reconstructability scores for each patch, only a subset of patches are selected as reference patches to reduce computation time. Choosing this subset by simple methods such as thresholding by the reconstructability score does not take into account coverage of the imaged 3D surface, and can result in an incomplete reconstruction. As a solution, we propose a patch selection framework that ensures coverage of the 3D surface while maximizing the scores of the selected patches.

Sparse Coarse-to-fine MVS: Unfortunately, sparsely computing surface estimates in an image may have adverse effects on MVS algorithms which rely on dense surface estimation to regularize and improve the accuracy of individual pixel estimates [24, 10, 28]. To combat this, we propose a coarse-to-fine surface estimation strategy which ensures that the surface estimates around a selected patch are coarsely initialized by estimates from the previous scale. This coarse-to-fine approach mitigates the sparsity issue and

also further accelerates the MVS algorithm up to 3x by requiring less processing at the finest scale.

Combining these contributions together produces a pipeline which can drastically accelerate high-quality MVS. The pipeline is shown in Figure 2. Experiments on the DTU Robot Image Dataset [19, 1] and Tanks and Temples [22] show that using our combined method can accelerate the reconstruction process by $10-30\times$ for a small decrease in completeness.

2. Related Work

Multi-View Stereo: We provide a high-level review of MVS techniques here and direct the reader to the MVS tutorial by Furukawa *et al*. [8] for a more detailed reference. Common MVS techniques include region growing methods, volumetric methods and depth map based methods. Region growing methods, such as PMVS [9] and MVE [6, 15], find a set of surface patches around discriminative interest points and progressively grow a region of depth samples around these points. Volumetric approaches [30, 23] perform reconstruction directly in a 3D representation, such as a voxel grid. Depth map based methods [10, 12] compute individual depth maps for each view followed by a fusion step to merge the depth maps into a final 3D representation. In the following paragraphs, we focus more on depth map based methods as our method falls into this category.

View Selection for Scalable MVS: As mentioned in the introduction, the core techniques for achieving scalability in large-scale MVS are *view filtering* and *neighbor selection*. These techniques can reduce the problem from scaling quadratically to linearly in the number of images. Furukawa *et al*. [7] perform simultaneous view filtering and neighbor selection by structuring their image subsets as small, potentially overlapping clusters of images where the images in a cluster are chosen to maximize coverage and minimize redundancy with other images. Goesele *et al*. [15] performs neighbor selection by choosing the best neighboring views to match against for each image using a two-level selection scheme: a global selection scheme selects a set of neighbors with good triangulation angle and similar scale and then a local selection scheme chooses views from this set that are both diverse and match well against the reference image. Zheng *et al*. [33] also performs neighbor selection but does so by solving depth estimation and neighbor selection as a joint problem. Schönberger *et al*. [28] extends Zheng *et al*.'s work by performing neighbor selection at the per-pixel level.

Learning Patch Fitness: At a more abstract level, learning the reconstructability of an image patch is an instance of the more general idea that it is possible to predict the "fitness" of an image patch for a specific task. This idea has previously been employed in the context of interest point matching for Structure-from-Motion by Hartmann *et al*. [16]. Hartmann *et al*. learned to predict which interest points will have a high chance of a successful match so that they can discard points with a low probabiliy of matching. Penate-Sanchez *et al*. [25] also used this idea in the context of predicting the matchability of templates.

Learning for MVS: Several works have been proposed which attempt to integrate learning into an MVS pipeline. Galliani *et al*. [11] propose to learn to predict surfaces normals for "bad" image regions by learning from the image regions which were well reconstructed. Ji *et al*. [20] propose an end-to-end neural network architecture for MVS, though their performance does not exceed that of traditional methods. In contrast, we propose to learn which portions of an image are good for reconstruction. Our learned reconstructability score could be used as a complementary signal to these existing approaches since they do not attempt to explicitly estimate reconstructability.

Faster Stereo Matching: A broad set of techniques have been explored for accelerating the stereo matching phase of MVS, motivated by the significant compute required for finding precise stereo correspondences. Geiger *et al*. [13, 14] propose using interest points to accelerate matching by constraining the disparity search space. Bleyer *et al*. [24] propose PatchMatch Stereo, which decouples the runtime of their method from the disparity range that must be sampled for locating stereo correspondence via a randomized, iterative algorithm based on PatchMatch [2]. Galliani *et al*. [10] extend this approach to a multi-view setting and propose a highly-parallel plane propagation scheme which exploits modern parallel hardware. Our proposed coarse-to-fine surface estimation scheme could be applied to several of these methods to provide further regularization and reduce processing time.

3. Patch-wise View Filtering

In this section, we will first discuss how we define the reconstructability score, how we can use ground truth data to train a network that can predict the reconstructability score from input images, and how we can generate training data when we do not have the ground truth surface (Section 3.1, Figure 2 top left). We then discuss how we use the scored input images, along with a coarse reconstruction, to perform patch-wise view filtering, which both chooses highly reconstructable image patches and ensures that the final reconstruction is complete (Section 3.2, Figure 2 middle).

3.1. Learning a Reconstructability Score

In order to filter image patches *before* performing stereo matching with minimal effect on the accuracy of the reconstruction, we would like to choose image patches that are likely to produce the most accurate surface estimates. Many factors reduce image quality and thus accurate normal/depth

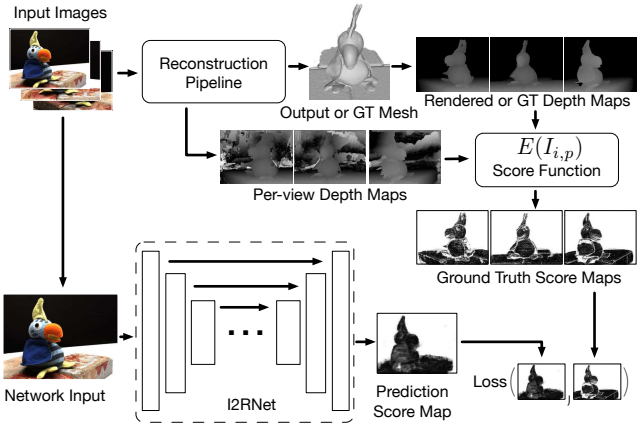

Figure 3: **Reconstructability Score Learning Pipeline:** The labels, i.e., score maps, are produced by taking the per-pixel difference between the computed depth for each view and the true depth from the ground truth. We then use these labels to train a CNN to predict score maps directly from input images.

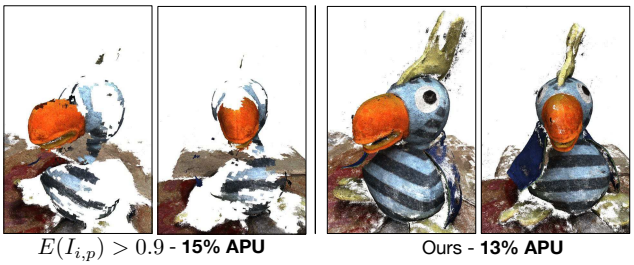

Figure 4: Comparison of two methods to reduce the aggregate pixels used (APU). Left: A fixed threshold is used to remove image patches. Notice the uneven distribution of samples in the point cloud on the left (e.g. the high density on the ground, the low density on parts of the bird). Right: Our patch-wise filtering is used to remove patches, which leads to a significantly more complete reconstruction.

estimation: focus blur from surfaces outside the camera's depth of field, imaging the surface at a highly oblique angle to the imaging plane, depth discontinuities, motion blur, and specular materials. Detecting and quantifying the influence of these factors on the accuracy of the reconstructed surface is challenging. Therefore, instead of trying to model each phenomenon explicitly with handcrafted features, we train a convolutional neural network (CNN) to predict a reconstructability score, which measures how accurate the surface estimates produced by the patch will be. We refer to this CNN as the Image-to-Reconstructability network (I2RNet). Figure 3 shows the training process, which is run offline – when reconstructing a new set of images, the I2RNet does not need additional training.

Training Loss: To train our network, we first need to define our reconstructability score. Let R_i be the surface (normals and depth) estimated by an MVS method for view I_i. Let G be the ground truth surface geometry. Since MVS is primarily concerned with reconstructing accurate 3D points, we use the difference between the positions of the surfaces as an error metric. Specifically, we draw inspiration from the error metric $\mathcal{L}^2$ discussed by Cohen-Steiner *et al.* [4], which quantifies the error in position between two surfaces, to produce the reconstructability score,

$$E(I_{i,p}) = 1 - \min(1, \frac{\|d(R_i,p) - d(G,p)\|}{d(G,p)}) \quad (1)$$

where $I_{i,p}$ indicates pixel p in I_i, and $d(R_i,p)$ is the depth of the surface R_i from the camera of image I_i at pixel p. We normalize by $d(G,p)$ so that the score does not depend on the absolute positions of the surfaces and we cap the normalized difference between the two surfaces to 1 since an error near or larger than 1 suggests the predicted surface is very inaccurate and it is of minimal utility to differentiate between two very inaccurate predictions.

Since it is common to not have the ground truth surface G, we propose to approximate the ground truth when unavailable by the output of a high-quality (expensive) 3D reconstruction pipeline. Since we are able to compute such a surface without human annotation, this allows us to generate labeled images patches $(I_{i,p}, E(I_{i,p}))$ for training our I2RNet from just multi-view input images.

Training Details: The architecture of our I2RNet for learning $E(I_{i,p})$ is a fully-convolutional variant of U-Net [26] augmented with residual connections [17] and is pictured in Figure 3. As in U-Net, the architecture consists of a series of encoding layers followed by a series of decoding layers with connections back to the encoding layers. The final output is the same resolution as the input and represents the reconstructability score for each pixel. We hypothesize that it is possible to predict the reconstructability of a patch using a local support window, so we find a relatively small network with 6 encoding and 6 decoding blocks sufficient.

3.2. Patch-wise View Filtering Framework

Based on the predicted reconstructability scores, the primary challenge in selecting a subset of image patches for MVS reconstruction is that the effect of not reconstructing a single patch is not independent of the other image patches involved in the reconstruction, since it could potentially lead to an incomplete reconstruction if all patches viewing the same surface region are removed. Figure 4 shows two point clouds and the aggregate pixels used (APU) to produce them. We define APU as the percentage of pixels used as reference pixels during stereo matching across all views. For the point cloud on the left of Figure 4, we filtered image patches by removing any patch with a reconstructability score less than 0.9. The resulting point cloud suffers in

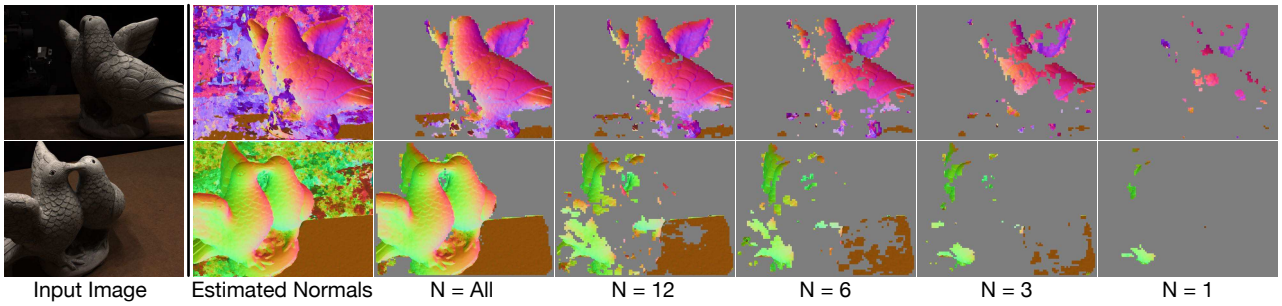

Figure 5: Normal maps produced by our method for a scene from DTU [19] as we vary N, the number of redundant views of a surface region. N decreases from left to right, with the left most normal map being unmasked.

terms of completeness since some surface regions are challenging to reconstruct and never receive a score higher than 0.9. Therefore, we propose a new completeness constrained patch-wise filtering framework, which ensures that the set of patches we choose spans the surface, ensuring completeness, and that the selected patches are of high quality, improving accuracy. The core idea of the algorithm is to use coarse depth estimation, which can be acquired quickly, to group together image patches which observe the same surface region and then select the top N views in each group based on the reconstructability score. Using this approach, which we describe next, the point cloud on the right of Figure 4 achieves a balance of both sparsity in pixels and completeness.

Coarse Depth Estimation: To compute coarse depth, we leverage the MVS algorithm of Galliani *et al.* [10] since it is very efficient (see Section 4). We downsize all input images I_i by a scaling factor S_c and then perform depth estimation to produce coarse depth maps D_i^c. If some of the depth estimates are unreliable, we may falsely establish correspondence between patches which do not actually correspond, so we perform a very conservative geometric consistency check, as is performed by Galliani *et al.* [10], to filter outlier correspondences.

Patch Filtering with a Coarse Voxel Grid: Given coarse depthmaps D_i^c and score maps $E(I_{i,p})$ generated by our I2RNet, we construct a coarse voxel grid with voxels of size W_v to facilitate selection of image patches that provide both completeness and accuracy. The intuition is that for each occupied voxel, we select the top N most reconstructable views and run reconstruction only for the patches from those views to estimate the surface in the voxel. D_i^c determines which image patch falls into which voxel. If we aggregate the selected patches for all voxels, we end up with a mask for each image which indicates the image patches where surface estimation should be performed. Specifically, let M_i denote the mask for I_i, where $M_{i,p} = 1$ when patch p is selected and 0 otherwise. Let $\mathcal{V}_j$ correspond to the set of patches (i, p) which falls into voxel j. The loss function of patch-wise view filtering, which solves for M_i for all images, is then:

$$\begin{aligned} &\underset{M_i,\,\forall i}{\text{maximize}} \quad \sum_{i,p} E(I_{i,p}) M_{i,p} \\ &\text{subject to} \quad \forall j, \sum_{(i,p)\in\mathcal{V}_j} M_{i,p} = min(N, |\mathcal{V}_j|). \end{aligned} \tag{2}$$

where $E(I_{i,p})$ here is the min score over all pixels in the patch. We observe that different $\mathcal{V}_j$'s can be decoupled, so we can solve Equation 2 optimally by greedily selecting the N views with the highest $E(I_{i,p})$ for each voxel. Since we only initialize a voxel if a patch intersects it, the memory requirements are low and the speed is fast because the grid is sparse. Figure 5 shows an example of masked normal maps produced by our approach for varying N, from selecting all patches in a voxel to just one.

4. Sparse Coarse-to-fine Surface Estimation

Sparsely computing surface estimates in an image (Figure 5) may have adverse effects on MVS algorithms which rely on dense surface estimation to regularize and improve the accuracy of individual pixel estimates. For example, Galliani *et al.* [10] initialize each pixel p in the input image I with a random plane parameterized in scene space and these planes are then propagated to neighboring pixels based on a fixed, local propagation scheme over several iterations. This approach performs well despite the large parameter space of planes in 3D because even if only a single pixel attains a roughly correct plane, this plane is then propagated to neighboring pixels and refined for their position on the surface. However, it can easily suffer from local minima, especially when we apply the masks M_i computed by our framework since introducing sparsity results in less random samples for propagation (each pixel provides a single random surface estimate).

We address this issue by introducing a Coarse-to-Fine plane Diffusion strategy (CFD, Figure 6) inspired by Wu *et al.* [32] and Hu *et al.* [18]. We construct an image pyramid with H levels for the input images, normal maps, depth maps, and masks. We iteratively initialize each level of the input image pyramid by resampling the image to half the

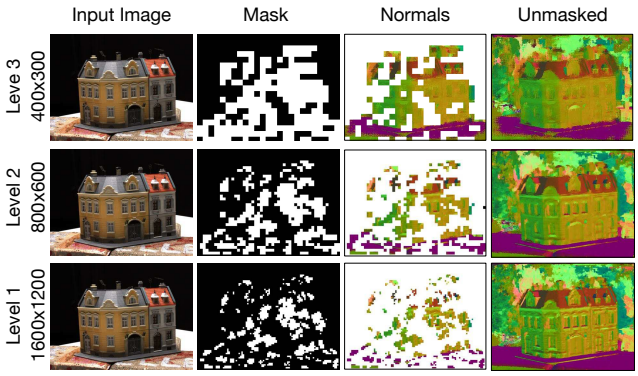

Figure 6: Intermediate outputs for a three level hierarchy of CFD. Notice how the normal map captures finer and finer detail as we move to higher resolutions.

resolution of the previous finest level. For the mask pyramid, we keep the same patch size at each level and conservatively initialize patches by setting $M_{i,p} = 1$ whenever any of the patches that project to the patch from the previous finer level are also set, e.g. the mask of Level 1 is a subset of Level 3 in Figure 6. After pyramid construction, we initialize the plane estimates at the coarsest level and perform the plane propagation as Galliani *et al*. did. We then extract a normal and depth map from the planes at each pixel and upsample them to the resolution of the next, finer level. We convert these upsampled depth and normal maps back into planes and then resume propagation. We iterate this process up to the finest resolution. At each level, we set the number of PatchMatch iterations as $\max(K \cdot 2^{1-h}, 2)$ where K is a base number of iterations and h, $1 \leq h \leq H$ is the current level, where $h = 1$ is the coarsest level. Figure 7 shows an example of the normal maps produced by our approach using varying hierarchy levels.

5. Evaluation

Our evaluation focuses on the accuracy of the I2RNet and its ability to generalize across datasets, the accuracy/completeness/speed tradeoff of our patch-wise selection framework, and the implications of coarse-to-fine surface estimation on the reconstruction results.

Machine Details. In all the following experiments, we use a machine with a 24-core Intel Xeon E5-2680 CPU, a M6000 NVIDIA GPU, and 256 GB of DRAM.

5.1. Datasets

To properly evaluate our method against the general MVS setting, we selected datasets which contain diverse scenes, objects, camera types, viewing angles, lighting conditions, and view overlap:

ScanNet: ScanNet [5] is a RGB-D dataset with over 1500 scans of 707 indoor locations (2.5 million 1296×968 images). Camera calibration and ground truth 3D surface reconstructions are provided. The large collection of images coupled with 3D surfaces make it an ideal dataset for pre-training the I2RNet to produce generalizable features.

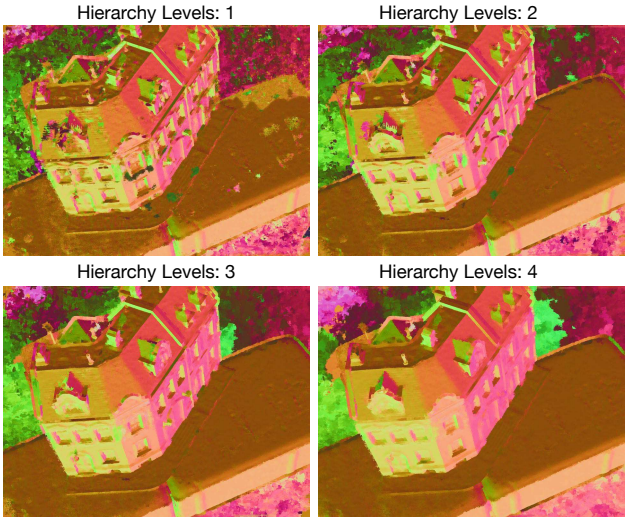

Figure 7: Visualization of final normal maps produced for increasing hierarchy levels H for CFD. Notice how no hierarchy (top left) produces noisy normal maps where as significant hierarchy (bottom right) produces smooth normals. Some very small features, such as the lamps near the doors in the middle, are missed at hierarchy level 4.

DTU: The DTU Dataset [19, 1] consists of 124 different objects captured from the same 49 to 64 camera angles under varying lighting conditions. The images are of moderately high resolution, 1600×1200. Ground truth data is provided via a structured light scanner. Following Galliani *et al*. [10], we use the images with the most diffuse lighting. For testing, the dataset uses 80 of the 124 scans. For training, we use the other 44 scans.

Tanks and Temples (T2): Tanks and Temples [22] is an end-to-end 3D reconstruction dataset that provides 4K videos of various indoor and outdoor man-made scenes. While T2 does provide training and testing datasets, we use the training dataset as the test set since it provides the ground truth point clouds, allowing us to evaluate the performance of our method under various settings without running into the submission limits of the online evaluation server. For the training set, we use the test set by generating ground truth data as described in Section 3.1.

5.2. I2RNet Accuracy and Generalization

We evaluated the I2RNet's ability to generalize from one dataset to another by training a model on one dataset and testing that same model on another dataset. Since we are interested in picking the top N views for a patch, we evaluate the I2RNet based on the ranking of different patches. For each voxel in our coarse voxel grid (Section 3.2), we

	Test Datasets		
	DTU	T2	Mean
ScanNet	0.17	0.06	0.12
DTU	0.18	0.03	0.11
ScanNet + DTU	**0.23**	0.06	0.15
T2	0.08	0.09	0.09
ScanNet + T2	0.14	**0.12**	0.13

Table 1: I2RNet generalization performance on DTU and Tanks and Temples (T2) when trained on the different datasets listed in the first column. The score is the Kendall tau rank correlation coefficient.

compute the ranking of camera views in that voxel based on the ground truth scores and based on the I2RNet predicted scores. We then compare these two rankings by using Kendall's tau (τ) ranking correlation coefficient [21] which produces a value between 1 and -1 indicating the correlation of rankings.

Table 1 shows the accuracy of models trained on ScanNet, DTU, or T2, and their performance on the test set of each dataset. As seen in the first row, the I2RNet generalizes when trained on ScanNet, achieving reasonable performance on both DTU and T2. This shows that the reconstructability of a patch can be estimated to some extent *directly from the image*, and that the I2RNet can rank the reconstructability of image patches in an unseen image. Note that there are still cases where reconstructability is not possible to predict from a single image. For example, if a patch is not observed by neighboring views, then it will be difficult to reconstruct the patch even if the patch itself appears to be highly textured and in-focus. When the ScanNet model is fine-tuned for a specific dataset (ScanNet + X), the I2RNet achieves its best performance on that dataset. This shows that there are still some dataset specific aspects to reconstructability which were not captured by ScanNet.

5.3. MVS Quantitative Evaluation

In this section, we evaluate our method on DTU and T2. In each evaluation, the main parameter we vary is the number of redundant views N selected for each voxel, which is how we influence the total time to compute a reconstruction (lower N values result in faster reconstructions). We evaluate on several settings for N, including ALL and FULL. For ALL, we select all patches in every voxel. For FULL, we disable masking of image patches and process every pixel for comparison with methods which process all pixels. This differs from ALL in that patches can be removed in ALL if the coarse depth estimation filters them out. Since we can not perform a geometric consistency check when $N=1$, we filter patches with $E(I_{i,p}) < 0.75$.

The methods that we compare against in our MVS performance evaluations are: **O-N:** (Ours-N) Our method,

	Acc.	Com.	F_1	APU	Time	
OF-1	0.375	0.253	0.302	4.2	**0.25s**	**262.6x**
OF-2	0.246	0.321	0.278	5.6	0.26s	247.5x
OF-3	0.226	0.306	0.260	8.0	0.29s	221.8x
OF-6	0.213	0.339	0.262	13.8	0.36s	178.7x
OF-12	0.222	0.228	0.225	22.2	0.47s	136.9x
OF-ALL	**0.208**	0.227	0.227	45.2	0.58s	110.9x
OF-FULL	0.211	0.216	0.213	100.0	1.62s	39.7x
O-1	0.281	0.273	0.277	**3.5**	1.31s	49.1x
O-2	0.233	0.273	0.252	6.7	1.91s	33.8x
O-3	0.219	0.259	0.238	9.5	2.28s	28.0x
O-6	0.213	0.234	0.223	16.3	3.27s	19.5x
O-12	0.222	0.184	0.201	26.0	4.82s	13.3x
O-ALL	0.221	0.168	0.191	45.2	6.83s	9.4x
O-FULL	0.223	**0.166**	**0.190**	100.0	15.76s	4.1x
GIPUMAF	0.212	0.296	0.247	100.0	5.17s	12.4x
GIPUMA	0.253	0.191	0.218	100.0	64.34s	1.0x

Table 2: Results on the DTU Robot Image Dataset [19] (lower is better). O-N and OF-N (Ours-N and OursFast-N) are our approach, where N is number of views selected for each surface region, O uses similar settings to GIPUMA, and OF uses similar settings to GIPUMAF (GipumaFast). Acc. and Com. stands for accuracy and completeness.

where N is the number of views selected for each voxel. **OF-N:** (OursFast-N) The same as O-N but with similar modifications as GIPUMAF below. **GIPUMA:** The method introduced by Galliani *et al*. [10] and the one we reimplement in Section 4 to produce O-N. **GIPUMAF:** (GipumaFast) Galliani et al. [10] further propose a faster but less accurate variant of their method, which we denote GIPUMAF. **COLMAP:** A general purpose end-to-end reconstruction pipeline [27, 28] that was state-of-the-art as of publication on Tanks and Temples.

For each dataset, we include a measure of accuracy (or precision) and completeness (or recall). We also consider the $F_1(a,c) = 2\frac{a \cdot c}{a+c}$ score (harmonic mean) where a is accuracy and c is completeness. This provides a single score to measure performance when accuracy and completeness vary. We also provide the average time taken to compute one depth map and the APU (% pixels used across all views), which indicates how many pixels were used as reference pixels.

Parameter Settings. Unless otherwise stated, we use $K=8$ iterations, $H=4$ hierarchy levels, and $S_c=8$ coarse scale factor.

5.3.1 DTU Evaluation

Following the evaluation protocol described in the DTU dataset paper [1], we evaluate on two metrics: accuracy, the distance between points in the MVS point cloud and in the ground truth point cloud, and completeness, the distance between points in the ground truth and the MVS point cloud. For both metrics, a lower score is better since they measure

	Prec.	Recall	F_1	APU	Time
O-1	0.216	0.626	0.306	**6.5**	**2.9s**
O-2	0.274	0.361	0.289	10.7	4.6s
O-3	0.266	0.523	0.330	14.3	5.0s
O-6	0.295	0.610	0.359	19.3	5.9s
O-ALL	0.277	0.672	0.351	31.0	8.6s
O-FULL	0.276	**0.682**	0.354	100.0	30.4s
COLMAP	**0.494**	0.566	**0.512**	100.0	100.3s

Table 3: Results on the Tanks and Temples Dataset [22] (higher is better). O-N is our approach, where N is number of views selected for each surface region. COLMAP [27, 28] is the current state-of-the-art on this dataset.

distance error in millimeters. The median is taken over all point-wise comparisons and then averaged over all 80 scans to produce the final score for the dataset. In addition to the time taken per depth map, we also provide the speedup compared to GIPUMA [10], which is state-of-the-art on the dataset. We use $W_v = 8.0$mm coarse voxel size.

Table 2 shows the final scores of the evaluation. As seen by the time required by the variants of our method, we are faster by up to two orders of magnitude compared to GIPUMA while gracefully degrading in completeness. Even when comparing against the faster baseline of GIPUMAF, we achieve an 11.0x speedup while matching the accuracy and completentess (see OF-12). Our base method, O-FULL, outperforms GIPUMA and achieves the best completeness and F_1 due to the incorporation of CFD. Our completeness is not as good at lower N as there are less views to check against when performing the geometric consistency check, whereas our accuracy does not decrease much because our error metric selects high quality patches. We also note that scans with mostly weakly-textured surfaces tend to reduce more in accuracy for lower N than those which are highly textured because weakly-textured regions rely heavily on neighbors for improving their estimates.

The cost of the patch-wise filtering framework is a minor overhead compared to the time for executing the rest of the pipeline. For example, the high-resolution depth estimation stage takes ~13 minutes in total (all images) for O-FULL. In comparison, the patch filtering stage takes ~20 seconds in all configurations, with the I2RNet taking only 120 milliseconds per view.

5.3.2 Tanks and Temples Evaluation

For Tanks and Temples, we report the precision, recall, and F_1 metrics as described in the T2 paper [22]. Precision is defined as the percentage of points in the computed point cloud that are within some distance threshold from the points in the ground truth point cloud, measuring accuracy. Recall is defined similarly with the distances instead being computed from the ground truth to the computed point cloud, measuring completeness. Higher scores are better.

H	Acc.	Com.	F_1	Time
1	**0.220**	0.175	0.195	46.4s
2	**0.220**	0.169	0.191	31.6s
3	0.223	0.166	**0.190**	25.0s
4	0.223	**0.165**	**0.190**	**14.7s**

Table 4: Comparison of varying levels of Hierarchy H for our method on the DTU dataset (lower is better).

For this dataset, we set $W_v = 0.02$. COLMAP is state-of-the-art on this dataset.

Table 3 shows the results. (We were not able to evaluate GIPUMA on this dataset since it fails due to GPU memory limitations.) As we decrease N, we observe a similar accuracy/completeness/speed tradeoff as with DTU. That is, we achieve significantly faster speeds for a minimal decrease in completeness. Since we base our method on GIPUMA, which uses a less sophisticated neighbor selection scheme than COLMAP, our absolute F_1 score is expected to be lower than that of COLMAP (also observed for GIPUMA by Schöps *et al*. [29] in their evaluation).

5.4. Analysis of CFD

We analyze CFD (Section 4) by evaluating our method with varying hierarchy levels H on the DTU dataset. Table 4 shows the result of running O-FULL with H = 1,2,3,4. By adding levels of hierarchy, the overall completeness improves, thanks to the robustness provided by initializing the finer levels with the coarser levels of the hierarchy. At the same time, since less iterations are needed at the finer scales, the total time taken decreases significantly, despite performing more iterations in aggregate over all the hierarchy levels. We also notice a minor decrease in accuracy (3 microns) at $H = 3,4$ since some very small features are not visible at the coarsest level (see Figure 7).

6. Conclusion

We have presented a framework for accelerating MVS pipelines through learning patch reconstructability. Evaluations show that our I2RNet trained on a large collection of scans is able to predict the reconstructability of patches to some extent based on the image alone. This enables us to efficiently identify reconstructable patches, which, combined with the proposed patch filtering and coarse-to-fine diffusion components, enables us to speed up 3D reconstruction with minimal loss in accuracy and completeness. In this work, we used learning from data to improve a targeted aspect of a high-quality MVS pipeline. Taking this approach, we were able to leverage all the existing theory and algorithms developed for MVS. It is exciting to consider how future data-driven approaches like ours might be used to remove the other assumptions and limitations of the hand-engineered heuristics traditionally used in MVS methods.

References

[1] H. Aanæs, R. R. Jensen, G. Vogiatzis, E. Tola, and A. B. Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision (IJCV)*, 2016. 2, 3, 6, 7

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *ACM Transactions on Graphics (TOG)*, volume 28, 2009. 3

[3] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross. High-quality single-shot capture of facial geometry. In *ACM Transactions on Graphics (ToG)*, volume 29, page 40, 2010. 1

[4] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Transactions on Graphics (ToG)*, 23(3), 2004. 4

[5] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 6

[6] S. Fuhrmann, F. Langguth, and M. Goesele. MVE: A multi-view reconstruction environment. In *Eurographics Workshop on Graphics and Cultural Heritage*, 2014. 3

[7] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2, 3

[8] Y. Furukawa and C. Hernández. Multi-view stereo: A tutorial. *Found. Trends. Comput. Graph. Vis.*, 9(1-2):1–148, June 2015. 3

[9] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(8), 2010. 3

[10] S. Galliani, K. Lasinger, and K. Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *International Conference on Computer Vision (ICCV)*, 2015. 2, 3, 5, 6, 7, 8

[11] S. Galliani and K. Schindler. Just Look at the Image: Viewpoint-specific surface normal prediction for improved multi-view reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[12] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 3

[13] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision*, 2010. 3

[14] A. Geiger, J. Ziegler, and C. Stiller. StereoScan: Dense 3D reconstruction in real-time. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968, 2011. 3

[15] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections. In *International Conference on Computer Vision (ICCV)*. IEEE, 2007. 2, 3

[16] W. Hartmann, M. Havlena, and K. Schindler. Predicting Matchability. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 3

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[18] Y. Hu, R. Song, and Y. Li. Efficient coarse-to-fine patch match for large displacement optical flow. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[19] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs. Large scale multi-view stereopsis evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 2, 3, 5, 6, 7

[20] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang. SurfaceNet: An end-to-end 3D neural network for multiview stereopsis. In *International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[21] M. G. Kendall. Rank correlation methods. 1955. 7

[22] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and Temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4), 2017. 2, 3, 6, 8

[23] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision (IJCV)*, 38(3), 2000. 3

[24] C. R. Michael Bleyer and C. Rother. PatchMatch Stereo - Stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, 2011. 2, 3

[25] A. Penate-Sanchez, L. Porzi, and F. Moreno-Noguer. Matchability prediction for full-search template matching algorithms. In *International Conference on 3D Vision (3DV)*, 2015. 3

[26] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015. 4

[27] J. L. Schönberger and J.-M. Frahm. Structure-from-Motion Revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7, 8

[28] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 2, 3, 7, 8

[29] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8

[30] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision (IJCV)*, 35(2), 1999. 3

[31] E. Tola, C. Strecha, and P. Fua. Efficient large scale multi-view stereo for ultra high resolution image sets. *Machine Vision and Applications*, 2012. In press. 1

[32] C. Wu, M. Zollhöfer, M. Nießner, M. Stamminger, S. Izadi, and C. Theobalt. Real-time shading-based refinement for

consumer depth cameras. *ACM Transactions on Graphics (ToG)*, 33(6), 2014. 5

[33] E. Zheng, E. Dunn, V. Jojic, and J.-M. Frahm. PatchMatch based joint view selection and depthmap estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 2, 3