Multi-View Depth Map Estimation With Cross-View Consistency

Jian Wei
jian.wei@graphics.uni-tuebingen.de
Benjamin Resch
benjamin.resch@uni-tuebingen.de
Hendrik P. A. Lensch
hendrik.lensch@uni-tuebingen.de

Computer Graphics
Tübingen University
72076 Tübingen
Germany

Abstract

To increase robustness of depth-map-based multi-view stereo approaches, we combine several techniques: Depth estimates are propagated in parallel in the local neighborhood to efficiently spread reliable depth information into regions without prominent structures. A faster coarse-to-fine strategy fills in larger holes. Most importantly, a novel cross-view filtering stage based on free-space constraints and variance filtering, enforces consistency among the depth maps of different views. Our algorithm alternates between correlation and consistency optimization. This way, noisy patches and spikes are excluded so that the task for the subsequent depth map fusion algorithms becomes easier. Combining improved propagation, hierarchical estimation, and iterative multi-view consistency optimization, our method increases the estimation speed, generates dense depth maps with desirable global consistency, and yields convincing 3D reconstruction results.

1 Introduction

Multi View Stereo (MVS) aims to establish 3D models from multiple calibrated images. Based on the taxonomy of Seitz *et al*. [34], MVS algorithms can be loosely divided into: 3D volumetric approaches [22, 35], surface evolution techniques [37, 41], feature extraction and growing methods [2, 11, 17, 18], and the algorithms based on depth maps [6, 14, 16, 30]. The MVS evaluation website [1] shows that the last two occupy most of the top performers. Some methods employ region growing in the depth map creation stage of the last category, followed by depth map fusion, among which are the works of Goesele *et al*. [17] and Bailer *et al*. [2]. We base our spatial depth propagation on these two methods tackling their drawbacks, and propose a valid improvement for the whole depth-map-based MVS system.

Goesele *et al*. [17] grow regions with a priority queue of matching candidates, by employing robust statistics and adaptive view selection, but they only estimate depth in reliable regions. Bailer *et al*. [2] alternate between left/rightward and up/downward propagations along quite long scanlines. Though they achieve dense and fast estimation by exploiting GPU parallelism, the iterative long propagation steps slow down the entire estimation.

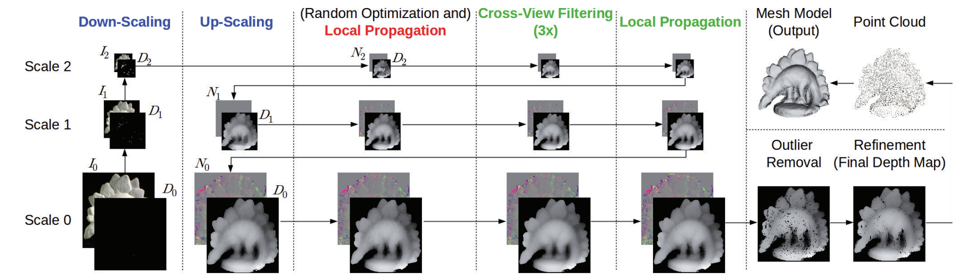

Figure 1: Our processing pipeline for one view of Dino dataset. I_k, D_k, and N_k are the image, depth map, and normal map at scale k. I_0 is the input image, and D_0 is initialized from bundler [36]. Our key steps (main differences from the work [2]) are highlighted: hierarchical framework (blue), local propagation (red), and cross-view filtering with an additional propagation pass (green). Backgrounds are removed by thresholding.

A more crucial drawback of these methods is that they only estimate depth map for each view independently, so inconsistent outliers may exist and probably grow during propagation, producing unstable estimates across views. This leads to a large amount of estimates removed in the depth map merging stage after consistency checking, and diminishes the reconstruction quality. This also applies to most of other depth-map-based methods, considering that they only focus on the correlation measure in their depth map estimation.

In this paper, we propose an iterative hierarchical method for depth map estimation (see Fig. 1 for our workflow), combining both correlation certainty and multi-view consistency:

1) A parallel, local propagation scheme grows depth along short scanlines. To get dense estimates in poorly textured regions, we spread the sparse estimates to neighbors at a coarse scale by down-scaling, and then depth estimates grow at finer scale using the up-scaled results from lower resolution. This also leads to a significant speed-up of our estimation.

2) Our main objective is to produce globally consistent depth maps. Recent work [27] on optical flow estimation obtains temporal consistency by filtering along motion paths, which simultaneously uses and estimates per-pixel results. Based on this idea, we impose a cross-view filtering stage after each view-dependent propagation pass, and use a second propagation to spread the optimized results. Thereby, we can determine the consistent depths earlier so that the consistency can be balanced against the non-robust correlation measure. This is not possible in the typical setup of individual depth map creation plus subsequent multi-view fusion. Moreover, filtering across views also can fill holes using the depth candidates projected from other views. The superior consistency and density of our depth maps puts less stress on the fusion steps producing denser output with less outliers.

2 Related Work

Region Growing. Some methods [11, 18, 20] propagate sparse patches in 3D space to avoid computations, cleaning up, and merging of noisy or redundant depth maps. Goesele *et al*. [17] prioritize the pixel traversal in image space by matching confidence. The PatchMatch stereo works [4, 5] use a top-left to bottom-right order in odd iterations, and the reversed order in even iterations. These methods [4, 5, 17] though achieving high accuracy are difficult to parallelize. Bailer *et al.* [2] propose a more GPU friendly propagation method with long parallel horizontal and vertical scanlines. Because such propagation does not make full use of the GPU, we employ a more effective local traversal scheme for speed improvement.

Dense Reconstruction. Many works have been presented to improve the reconstruction density. They adopt multi-baseline matching [30], dynamic programming [31, 32], silhouette and stereo fusion [8], scaled window matching [6], or identification of dominant plane orientations [12]. Unlike these methods, Goesele *et al*. [16, 17] only deal with those points with strong multi-view correlation. For dense estimation, we and Bailer *et al*. [2] both traverse all pixels in the region growing. Our hierarchical propagation and cross-view filtering can fill in otherwise unrecoverable regions producing depth maps of higher coverage.

Consistent Depth Estimation. An early work [23] gets consistent estimates using global energy minimization via graph cuts. They treat input images symmetrically and encode visibility constraint properly. Campbell *et al*. [7] introduce an unknown state for those pixels with inconsistent depth values and use a subsequent global regularization step to recover the surfaces in these places. Furukawa and Ponce [11] enforce global visibility consistency and a weak form of regularization to eliminate incorrect 3D patches after each patch expansion.

In the optical flow estimation area, Lang *et al*. [27] introduce a temporal smoothness assumption for consistent results. Beginning with the sparse estimates and an edge-aware filtering in image space, they filter the results in temporal domain by following motion vectors. Then the estimates are updated by iterating between spatial and temporal filtering passes.

Since most depth-map-based MVS works do not use a consistency optimization process during depth estimation, unreliable values may affect the accuracy of all following steps. We base our approach on the work [27] by performing a cross-view filtering in addition to independent propagation. Our method iteratively produces and optimizes consistent results, and thus improves the final consistency of the depth maps.

Depth Map Merging. To reconstruct the 3D models many depth map fusion and mesh creation approaches [9, 10, 19, 28, 29, 33, 42] have been proposed. They typically reject lots of points using noisy and inconsistent depth maps, and thus induce a loss of detail in the final mesh. Instead, we improve both accuracy and consistency of each individual depth map in the depth map estimation stage. This reduces the burden in the merging and 3D model polishing steps, so more consistent geometries with high coverage are reconstructed.

3 Depth Map Estimation

Like Bailer *et al*. [2], our depth map estimation is based on the PatchMatch work [3], but uses the workflow in Fig. 1 with the highlighted novel contributions. After initializing the sparse depth map and selecting secondary images for each view, all images and depth maps are down-scaled. Next, we start our hierarchical estimation. Concretely, at each scale we use a propagation-filtering-propagation approach. The propagation traverses pixels along short paths. The cross-view filtering step removes inconsistent outliers and propagates reliable information to different views. Then the depth and normal maps are up-scaled for the estimation at the next consecutive scale. Finally, inconsistent values are rejected by consistency checking, and an edge-aware filter is used for refinement. Each view selects at most 6 secondary images using the scale robust method [2]. We use three scales with scaling factor of 2. At each scale, three cross-view filtering passes are performed, and before the first propagation step, randomly shifted depth estimates and random normals are assigned if smaller matching errors are obtained (the random optimization step in Fig. 1).

Initialization. Let D_k, N_k, and E_k be the depth, normal, and matching error maps of a reference view r at scale k. For each pixel p, we initialize its finest-scale depth $D_0(p)$ from bundler [36] if p is feature point; otherwise $D_0(p) = 0$. Its normal $N_k(p) = \{N_k^x(p), N_k^y(p)\}$ includes the gradients of the tangent plane in x and y directions. The surface normals are initialized fronto-parallel at the coarsest scale, *i.e.* $N_2(p) = \{0,0\}$. Before the estimation at each scale, E_k is initialized using the existing depth and normal estimates.

Matching Error Calculation. We calculate the matching error using the self weighted average: 1 minus the Normalized Cross Correlation (NCC) scores [2]:

$$E_k(p, D_k(p), N_k(p)) = \frac{\sum_{i \in \Phi_r} 1 - NCC_{r,i}(p, D_k(p), N_k(p)) \cdot \frac{1}{1 - NCC_{r,i}(p, D_k(p), N_k(p))}}{\sum_{i \in \Phi_r} \frac{1}{1 - NCC_{r,i}(p, D_k(p), N_k(p))}} \tag{1}$$

where Φ_r is the secondary views of the reference view r. The calculation of the NCC score $NCC_{r,i}(p, D_k(p), N_k(p))$ for tilted patches can be seen in the work [11].

This section introduces our key steps: the local propagation, the hierarchical framework, and the cross-view filtering. We also describe how the depth maps are cleaned and refined.

3.1 Local Propagation

In the propagation, good estimates are dispersed into the neighborhoods by traversing all pixels if the propagated value improves the correlation measure. Specifically, for a pixel $p = (x, y)$, loop over each secondary pixel $q = (x', y')$ with $D_k(q) > 0$. Figure 2 shows an example of downward propagation from three secondary pixels $(x-1, y-1)$, $(x, y-1)$, and $(x+1, y-1)$. Since the depths are not equal in a tilted patch, a depth hypothesis $d = D_k(q) \cdot (1 - 2^k \cdot (x'-x) \cdot N_k^x(q) - 2^k \cdot (y'-y) \cdot N_k^y(q))$ is calculated depending on the patch normal by modifying the depth in both x and y directions, where 2^k considers the effect of image scaling. If p has no estimate or we get a matching error $e < E_k(p)$ for p using d and $N_k(q)$, we update its depth, normal, and matching error.

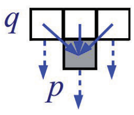

Figure 2: An example of downward propagation.

The work [2] traverses pixels following parallel scanlines with length of half the image height (width) in the vertical (horizontal) propagation. They scan downward and upward (leftward and rightward) on GPU simultaneously, and achieves faster estimation compared to other CPU-based methods [4, 5, 17]. However, their speed is still limited by too few scanlines to keep the GPU busy. So we shorten the traversal distance such that more GPU threads can be assigned. In the vertical (horizontal) propagation, we select an eighth of the image height (width) as the length of each scanline. In every other iteration vertical and horizontal propagations are applied alternately.

3.2 Hierarchical Framework

For textureless regions with few initial estimates, one propagation alone at the original scale is insufficient due to the locality of short scanlines. We solve this problem by down-scaling the depth map and spreading the sparse data into the neighborhoods. This way, one propagation at the coarsest scale can fill most of the holes. Then the estimates are used for the consecutive finer scale by up-scaling. Another benefit is the reduced overall time of depth estimation, since the scaling process is negligible compared with the speed-up of propagation. In practice, we also down-scale the images and up-scale the estimated normal maps.

We base our up-scaling procedure on the joint bilateral upsampler [24], which guides data interpolation by considering the information from the high-resolution image. Let I_k be the image at scale k, and S_k be its depth map D_k or normal map N_k^x or N_k^y. We up-scale S_{k+1} to

S_k using Eq. 2a, where N is a neighborhood of pixel p. q may be a pixel at finer-scale k or the corresponding pixel at coarser-scale $k+1$. The weight $\omega(p,q) = s(||p-q||) \cdot r(I_k(p) - I_k(q))$ depends on the Gaussian functions s and r with $\sigma_s = 0.4$ and $\sigma_r = 0.2$, decreasing with larger spatial distance and intensity difference. Similarly, we down-scale images and depth maps using Eq. 2b, where $\omega(p,q) = s(||p-q||) \cdot r(I_{k+1}(p) - I_{k+1}(q))$.

$$S_k(p) = \frac{\sum_{q \in N} \omega(p,q) \cdot S_{k+1}(q)}{\sum_{q \in N} \omega(p,q)} \quad (2a) \qquad S_{k+1}(p) = \frac{\sum_{q \in N} \omega(p,q) \cdot S_k(q)}{\sum_{q \in N} \omega(p,q)} \quad (2b)$$

Because the sparse depth data often belong to isolated pixels in the initial depth maps, most of the depth data in the first round are just copied into their neighbors during down-scaling, except in the case that two spread regions interfere.

3.3 Cross-View Filtering

So far, our propagation is per view, which may lead to global instabilities of depths. Inspired by the temporally consistent optical flow estimation [27], after propagation of all views at the coarsest scale, we perform a cross-view filtering for each reference view to improve the depth consistency. Then a second propagation spreads the optimized estimates. The results are updated in each propagation-filtering-propagation iteration at finer scales.

The temporal filtering of Lang *et al.* [27] follows optical flow vectors to avoid averaging across object boundaries. Similarly, our cross-view filtering considers the projection relationships of pixels between views using the depth information. For each depth value, we find the corresponding pixels in the secondary views, and project them back into the reference view obtaining new depth candidates. These candidates are weighted by the depth difference between the reference and secondary views to get an optimized depth. In some cases, this depth projection from secondary views can even fill holes in the reference, spawning further, more consistent propagation.

In the following, we first describe how to obtain the depth candidates. Then based on the visibility analysis of Merrell *et al.* [29], a depth value is judged to be consistent or conflicted with other views. According to these constraints, we describe two coherence measurements, which are utilized in the cross-view filtering introduced subsequently, the next subsection for post-processing, and our result evaluations.

Depth Candidates. Let D_r and D_i be the depth maps of the reference view r and the view $i \in \Phi_r$. As shown in Fig. 3, for each pixel p in the view r one can obtain two depth candidates $D_{i\to r}(p)$ and $D_{r\to i\to r}(p)$ per secondary image. To obtain $D_{i\to r}(p)$, we project all depth values from the view i to r and pick the least distant depth at position p. To obtain $D_{r\to i\to r}(p)$, we project the estimate $D_r(p)$ into the view i, obtaining a projected depth $D_{r\to i}(p)$ and a position $p_{r\to i}$ on the image plane of the view i. We project its depth $D_i(p_{r\to i})$ back into the view r to obtain the new candidate.

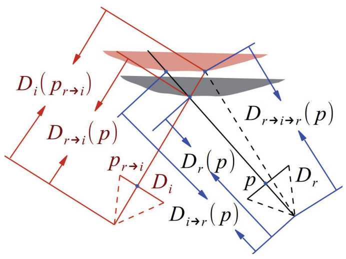

Figure 3: The two depth candidates obtained from a secondary view i for the reference view pixel p.

Visibility Constraints. If the views r and i see the same surface, *i.e.* $D_r(p)$ and $D_{i\to r}(p)$ are close enough, $D_r(p)$ is ***consistent***; otherwise, if $D_{i\to r}(p) < D_r(p)$, the estimated surface of p is ***occluded***. If $D_{r\to i}(p) < D_i(p_{r\to i})$, we say that the surface of $p_{r\to i}$ is ***violated*** by $D_r(p)$.

Coherence Measurements. We use two coherence measurements for the depth map pixels:

Consistency: The consistency rating [2] $M_r(p)$ for $D_r(p)$ is defined by Eq. 3a, increased by ***consistent*** secondary views (B_i) and decreased by ***violation*** case (A_i). The difference of violation and ***occlusion*** cases (C_i) is used as penalty because it is impossible that a point is both too near and too far. Given Eq. 3b which is used in the following outlier removal step, $D_r(p)$ is the more consistent, the higher $M_r(p)$ is.

$$M_r(p) = \sum_{i\in\Phi_r} 2\mathbf{B}_i(p) - \left|\sum_{i\in\Phi_r} \mathbf{A}_i(p) - \sum_{i\in\Phi_r} \mathbf{C}_i(p)\right| \tag{3a}$$

$$M_r(p) \geq \begin{cases} 0 & \text{if } |\Phi_r| < 2 \\ 2 & \text{if } |\Phi_r| > 5 \\ 1 & \textit{else} \end{cases} \tag{3b}$$

Variance: If $D_r(p)$ is not conflicted with other views, the relative depth difference $\Delta D_{i\to r}(p) = \frac{D_r(p)-D_{i\to r}(p)}{D_r(p)}$ should be close to 0 for each view i. That means, all the values of $\{D_{i\to r}(p)\}$ should have a low variance relative to $D_r(p)$. We also consider the relative depth difference $\Delta D_{r\to i}(p) = \frac{D_i(p_{r\to i})-D_{r\to i}(p)}{D_i(p_{r\to i})}$, since in this case $D_{r\to i}(p)$ should also have a close value to the depth $D_i(p_{r\to i})$. We define the variance $V_r(p)$ for $D_r(p)$ in Eq. 4, considering the minimum value of $|\Delta D_{i\to r}(p)|^2$ and $|\Delta D_{r\to i}(p)|^2$ for each secondary view i, and we say that $D_r(p)$ is sufficiently consistent if $V_r(p) \leq \tau_1$.

$$V_r(p) = \frac{1}{|\Phi_r|} \sum_{i\in\Phi_r} \min\left(|\Delta D_{i\to r}(p)|^2,\ |\Delta D_{r\to i}(p)|^2\right) \tag{4}$$

The variance is more precise for measuring coherence, because the consistency rating can only be integer and depends on the number of secondary views. Therefore, we use the variance in our cross-view filtering while adopting the consistency for outlier removal.

Cross-View Filtering. In Eq. 5 we calculate a new depth $\tilde{D}_r(p)$ for each pixel p as a weighted average of $D_r(p)$ as well as the candidates $D_{i\to r}(p)$ and $D_{r\to i\to r}(p)$ projected from each view i. The weights ω are defined in Eq. 6 with $\tau_2 > \tau_1$ being a looser threshold.

$$\tilde{D}_r(p) = \frac{\omega_r(p)\cdot D_r(p) + \sum_{i\in\Phi_r}(\omega_{i\to r}(p)\cdot D_{i\to r}(p) + \omega_{r\to i\to r}(p)\cdot D_{r\to i\to r}(p))}{\omega_r(p) + \sum_{i\in\Phi_r}(\omega_{i\to r}(p) + \omega_{r\to i\to r}(p))} \tag{5}$$

$$\omega_r(p) = 0, \quad \omega_{i\to r}(p) = 1, \quad \omega_{r\to i\to r}(p) = 0, \qquad \text{if } D_r(p) = 0 \text{ or } V_r(p) > \tau_2 \tag{6a}$$

$$\left.\begin{array}{c} \omega_r(p) = 1, \quad \omega_{i\to r}(p) = \exp\left(-\frac{|\Delta D_{i\to r}(p)|^2}{\sigma_d(p)^2}\right), \\ \omega_{r\to i\to r}(p) = \exp\left(-\frac{|\Delta D_{r\to i}(p)|\cdot|\Delta D_{r\to i\to r}(p)|}{\sigma_d(p)^2}\right), \quad \text{with } \sigma_d(p) = \gamma\cdot V_r(p) \end{array}\right\} \text{else} \tag{6b}$$

In Eq. 6a, if pixel p has no estimate, *i.e.* $D_r(p) = 0$, or its depth $D_r(p)$ is significantly inconsistent, *i.e.* $V_r(p) > \tau_2$, the projected depths $\{D_{r\to i\to r}(p)\}$ are unavailable or unreliable, so $\tilde{D}_r(p)$ is obtained simply using the average of the candidates $\{D_{i\to r}(p)\}$. In this case, cross-view filtering fills the hole here or corrects $D_r(p)$ with a consistent $\tilde{D}_r(p)$ value.

Otherwise, in Eq. 6b, we simply set the weight $\omega_r(p) = 1$ for $D_r(p)$, and define the weights $\omega_{i\to r}(p)$ and $\omega_{r\to i\to r}(p)$ as Gaussian functions, considering the relative depth differences: $\Delta D_{i\to r}(p)$, $\Delta D_{r\to i}(p)$, and $\Delta D_{r\to i\to r}(p)$. Here, $\Delta D_{r\to i\to r}(p) = \frac{D_r(p)-D_{r\to i\to r}(p)}{D_r(p)}$ measures the difference between $D_r(p)$ and the candidate $D_{r\to i\to r}(p)$ projected from view i.

The parameter $\sigma_d(p)$ controls the resulting smoothness. A smaller $\sigma_d(p)$ makes $\tilde{D}_r(p)$ gain less support from different depths, and a larger $\sigma_d(p)$ produces more smooth depth map but also increases the possibility of spreading depths across the object boundaries. The calculation of $\sigma_d(p)$ in Eq. 6b takes the coherence of $D_r(p)$ into account, enforcing more filtering on the $D_r(p)$ with a higher $V_r(p)$, and γ controls the overall effect of the filtering.

The above filtering may lead to slight shifting for some inliers which were accurate before. To avoid this, after obtaining the new depth $\tilde{D}_r(p)$, we perform three random shiftings

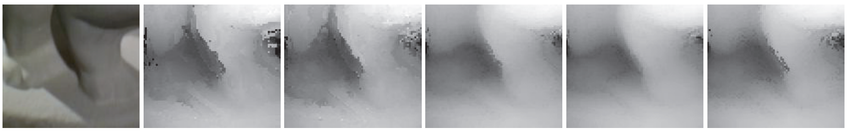
Image region
Without cross-view filtering
$\gamma = 0.5$ with random checking
$\gamma = 2.0$ with random checking
$\gamma = 10.0$ with random checking
$\gamma = 2.0$ without random checking

Figure 4: The first two are a region and its depth at scale 1 without our cross-view filtering. The next three show the filtering results using γ =0.5, 2.0, and 10.0, all with the subsequent random checking. The last one is the result using $\gamma = 2.0$ but without the random checking.

Figure 5: Sample images of the datasets.

Name	Images	Image Size	Patch Size	γ	λ
Fountain-P11 [38]	11	3072×2048	7×7	2.0	0.3
Dino [1]	363	640×480	7×7	2.0	0.1
Temple [1]	312	640×480	3×3	1.0	0.3
Sofa [2]	82	1853×1236	7×7	2.0	0.3

Table 1: Dataset characteristics.

(see [2] for more details) around $\tilde{D}_r(p)$ to get more depths and calculate the matching error (described in Subsection 3.1) for each. If the smallest error is lower than a threshold λ, we update $D_k(p)$ with the corresponding depth since the correlation measure is more robust in this case. Otherwise, the pixel p is likely located within a textureless region, so we depend more on the cross-view filtering and update $D_k(p) \leftarrow \tilde{D}_r(p)$.

Figure 4 presents the results for Dino dataset after cross-view filtering using different γ with and without the random checking. Obviously, a smaller γ makes less filtering effect, while a larger γ tends to homogeneously smooth the depths and leads to difficulty in finding back the details even if using additional random checking. In the case of adopting the trade-off value of γ with the random checking, we get better alignment between depth discontinuities and object boundaries compared to the result without in the last subfigure.

Coherence Constraint in Depth Propagation. A naive depth propagation may decrease the strong coherence achieved by the previous cross-view filtering step, due to the non-robust correlation measure. Therefore, in each depth propagation pass, for the consistent estimates with a variance lower than $1.5 \cdot \tau_1$, we only update the data if the matching error of the new depth and normal is 1.2 times smaller than that of the former estimate.

3.4 Outlier Removal and Refinement

Inconsistent outliers with a low consistency conflicting Eq. 3b are filtered out from the resulting depth maps. Finally we use a domain transform filter [15] with $\sigma_s = 30$ and $\sigma_r = 0.03$ to refine the results by filling the holes and then filtering the noise. We perform three complete filtering passes (*i.e.* horizontal+vertical 1D passes for each) for both tasks. In the hole filling the existing depths are not altered. To avoid involving inconsistent values, we also remove the outliers before each complete filtering pass and after the last pass.

4 Results

We used four datasets with the sample images shown in Fig. 5. Table 1 lists the number and size of their images. Our procedures introduced in Section 3 are all implemented on GPU. Results are also obtained using the GPU-based work of Bailer *et al.* [2]. We use the minimum least square algorithm [13] for point selection and optimization, and the Delaunay triangulation [26] for 3D reconstruction.

Table 2: Timings of each step using Bailer *et al*. [2] and different combinations of our individual processing steps, when reconstructing all views of Fountain-P11. The results are not finally refined in the approach of Bailer *et al*. [2].

Step		Bailer *et al*. [2]	Only LP	LP+HF	LP+CVF	LP+HF+CVF
Downscaling				8.4s		8.4s
1st	Propagation	174.3s	142.2s	13.6s	142.0s	13.6s
	Cross-View Filtering				151.2s	10.8s
	Propagation				226.9s	16.0s
	Upscaling			0.4s		0.4s
2nd	Propagation	1126.6s	880.3s	224.5s	1000.7s	250.0s
	Cross-View Filtering				193.3s	49.2s
	Propagation				951.9s	228.7s
	Upscaling			2.1s		2.1s
3rd	Propagation	418.0s	410.2s	417.4s	450.6s	458.2s
	Cross-View Filtering				204.1s	214.0s
	Propagation				279.9s	280.6s
Outlier removal		42.2s	41.2s	44.2s	48.1s	51.1s
Refinement			121.7s	144.1s	151.3s	189.5s
Overall		1984.4s	1866.8s	1079.1s	4020.3s	1844.1s

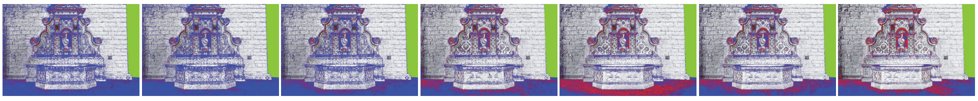

Figure 6: Relative error maps for the center view of Fountain-P11 after outlier removal, but not refined. LP+HF+CVF^1 uses cross-view filtering only for post-processing, and LP+HF+CVF^2 uses propagation-filtering at each scale without the second propagation. The blue pixels have no depth, the red have an error larger than 0.003, the green have no ground truth data, and the pixels with an error between 0 and 0.003 are marked gray 255 to 0.

Performance of the individual processing steps is evaluated on Fountain-P11 with the ground truth depth maps provided in [39], referring to Table 2 for timings, Fig. 6 for relative error maps of the center-view depth maps, as well as Table 3 for comparisons of statistics. The relative error evaluates depth accuracy between the depth estimates D and ground truth D_{gt}: $e = |D - D_{gt}|/D_{gt}$. The completeness relates the number of recovered pixels to the image size. The consistency (Eq. 3a) and variance (Eq. 4) measure the multi-view coherence.

Parameter Selection. The patch size for calculating the matching error, the filtering parameter γ, and the matching error threshold λ are sensitive to the image size and content of the datasets. Their values for different datasets are given in Table 1. For the Temple scene with low image resolution, sufficient texture, and frequent occlusions, we adopt small patch size for reliable matching and small γ value to preserve the accurate estimate from patch matching. For Fountain-P11 as well as Sofa with high image resolution, and Dino which lacks enough texture, we utilize large patches for more information in patch matching and large γ values for less inconsistent outliers. We set $\lambda = 0.1$ for Dino to depend more on cross-view filtering instead of non-robust patch matching due to poor texture, while larger λ values are used for others. The two thresholds τ_1 and τ_2 of the variance are independent of the datasets, so they are fixed in our experiments: $\tau_1 = 4 \times 10^{-6}$ and $\tau_2 = 36 \times 10^{-6}$.

Local Propagation (LP). Table 2 shows that local propagation with shorter scanlines is faster but might result in more holes if applied alone as shown in Figs. 6 and 7, as well as Table 3. It has little effect on the accuracy though.

Hierarchical Framework (HF). Table 2 also shows that our hierarchical estimation achieves a remarkable speed-up at coarse scales, and is faster than the work [2] and Only LP if a cross-view filtering and a second propagation are used (the time for scaling is almost negligible). Furthermore, comparing Only LP and LP+HF in Table 3 the hierarchical strategy improves the depth accuracy and density. Though the holes are filled regardless of accuracy by our coarse-to-fine strategy, the subsequent random optimization and propagation steps improve the estimates. Its hole-filling effect is also illustrated in Figs. 1 and 7.

Cross-View Filtering (CVF). Comparing the first three columns and the columns LP+CVF

Measurement	Bailer *et al.* [2]	Only LP	LP+HF	LP+CVF	LP+HF+CVF	LP+HF+CVF^1	LP+HF+CVF^2
Mean Rel. Error ($\times 10^{-3}$) $\downarrow$	1.663	1.414	1.236	2.407	1.732	1.505	2.062
Completeness (%) $\uparrow$	64.0	63.9	66.9	74.6	79.6	75.9	80.5
Mean Consistency $\uparrow$	9.083	9.019	9.124	9.611	9.556	9.253	10.090
Mean Variance ($\times 10^{-6}$) $\downarrow$	1.790	1.722	1.626	1.602	1.092	1.179	1.099
Mean Rel. Error of LP+HF+CVF on Pixels of Other Methods ($\times 10^{-3}$) $\downarrow$	1.102	1.068	1.142	1.292		1.319	1.368

Table 3: Statistical comparisons obtained by the methods in Fig. 6. The first row measures their relative errors on all reconstructed pixels. The last row measures the relative errors of LP+HF+CVF only on the pixels reconstructed by other methods. The arrows indicate preferred directions. Our approach LP+HF+CVF significantly improves all measures.

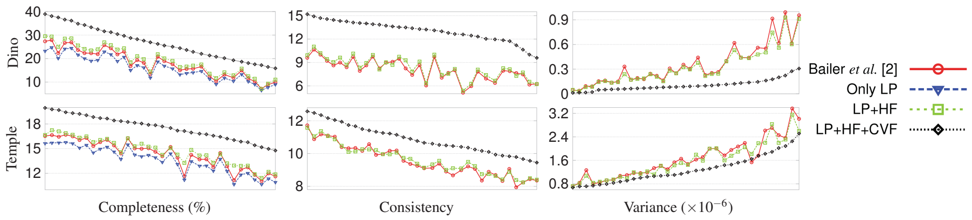

Figure 7: Completeness, mean consistency rating, and mean variance comparisons for some views of Dino and Temple. Views are sorted in preferred orders of LP+HF+CVF's results.

as well as LP+HF+CVF in Table 3, the cross-view filtering improves reconstruction density and depth coherence in terms of both consistency and variance at the same time. This is underlined in Figs. 6 and 7 where our algorithm shows better results for all tested views in Dino and Temple scenes.

We also evaluate the effectiveness of including cross-view filtering in the inner loop rather than using the consistency optimization as a post-processing filter at the finest scale (LP+HF+CVF^1 in Fig. 6 and Table 3). Only used as a post-process, coverage, consistency and variance are reduced. Similar results would appear if one included similar weighting in the mesh fusion step. The benefit of the propagation step after the cross-view filtering becomes obvious in the last column of Table 3 where the second propagation step is missing, resulting in higher coherence and higher density but lower accuracy.

As shown in the first row of Table 3, we obtain larger values of the overall relative error using the methods with the cross-view filtering than those of the methods without. However, as presented in the last row of the table, significantly higher accuracies are achieved if only the pixels reconstructed by other methods are considered. This demonstrates that, our combined approach can find a desirable balance between depth accuracy and multi-view coherence by iterating between correlation and consistency optimization, because high performance of either cue alone can not guarantee correct reconstruction of the 3D model.

Depth Map Merging and 3D Meshing. Figure 8 presents the normalized depth estimates and reconstructed models for sample regions of Fountain-P11 and Sofa. Our combined approach yields higher quality of mesh models with smooth surfaces due to the denser and more coherent depth maps, and the depth map refinement step further improves the results. We can get the same conclusion from the full pipeline as shown in Fig. 1. To compare the full models of all the datasets, please refer to the supplementary material.

To evaluate the benefit of our superior depth consistency in different depth map merging and meshing schemes, Fig. 9 shows the reconstructed models from our refined depth maps using: the minimum least square (MLS) algorithm [13] combined with Delaunay triangulation [26], MLS with subsequent Poisson surface reconstruction (PSR) [21], and the depth

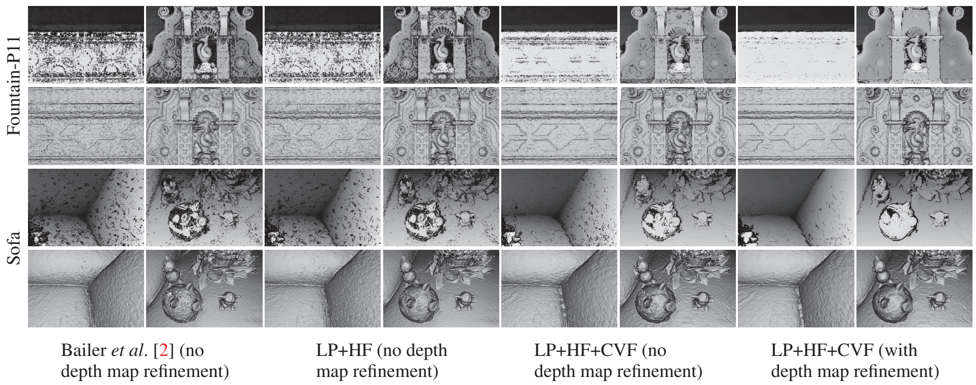
Fountain-P11
Sofa
Bailer *et al*. [2] (no depth map refinement)
LP+HF (no depth map refinement)
LP+HF+CVF (no depth map refinement)
LP+HF+CVF (with depth map refinement)

Figure 8: Depth maps (the first and third rows) and 3D models of the green regions in Fig. 5 using different methods after outlier removal and our final results with depth map refinement.

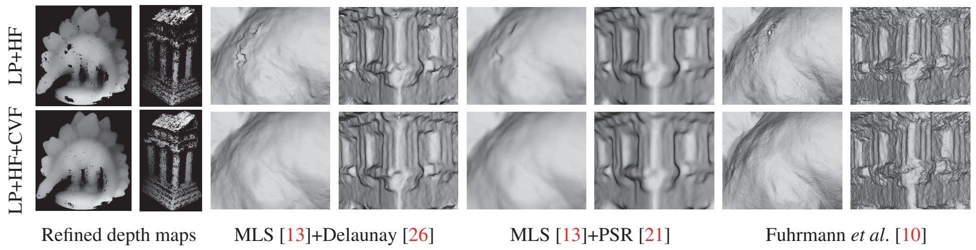

Figure 9: Refined depth maps of Dino and Temple without and with the cross-view filtering, and reconstructed details by using different depth map merging and meshing methods.

map fusion work of Fuhrmann *et al*. [10]. Our method with cross-view filtering produces smooth surfaces without loss of details, even if the homogeneous Dino has little texture data, which is problematic for most patch-based methods.

Comparisons to Other MVS Methods. We benchmark the reconstructions of our full pipeline (LP+HF+CVF) using the Middlebury evaluation website [1] which considers accuracy, completeness, and processing time. For Temple, we achieve an accuracy of 0.34mm at 99.4% completeness. Our high accuracy is ranked first among all evaluated MVS works. For Dino, we achieve an accuracy of 0.42mm at 98.1% completeness, demonstrating that our work is competitive with other state-of-art methods, in particular, better than some region-growing-based [17, 20] and depth-map-fusion [25, 40] techniques. We reconstruct Dino in 63 mins and Temple in 27 mins, placing our work among the most efficient approaches.

5 Conclusions

We have proposed a hierarchical depth estimation algorithm using local propagation and cross-view filtering for MVS. The method is accelerated by parallel propagation along short scanlines. The coarse-to-fine estimation produces denser reconstructions at reduced cost. The main focus is on optimizing the cross-view coherence of depth maps at all scales. Inconsistent depth estimates are removed and reliable, averaged candidates are propagated to neighboring views potentially helping in recovering the correct depth where otherwise would be a hole. The results show that all of our improvements lead to faster estimation, significantly denser and more consistent depth maps as well as more convincing 3D models.

References

[1] Multi-view stereo evaluation. http://vision.middlebury.edu/mview/.

[2] C. Bailer, M. Finckh, and H.P.A. Lensch. Scale robust multi view stereo. In *Proc. ECCV*, 2012.

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

[4] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. Pmbp: Patchmatch belief propagation for correspondence field estimation. In *Proc. BMVC*, 2012.

[5] M. Bleyer, C. Rhemann, and C. Rother. Patchmatch stereo–stereo matching with slanted support windows. In *Proc. BMVC*, 2011.

[6] D. Bradley, T. Boubekeur, and W. Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *Proc. CVPR*, 2008.

[7] N. Campbell, G. Vogiatzis, C. Hernandez, and R. Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *Proc. ECCV*, 2008.

[8] C.Hernandez and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *Comput. Vis. Image Underst.*, 96(3):367–392, 2004.

[9] B. Curless and M. Levoy. A volumetric method for building complex models from range images. *ACM Transactions on Graphics*, 30:303–312, 1996.

[10] S. Fuhrmann and M. Goesele. Fusion of depth maps with multiple scales. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH Asia)*, 30, 2011.

[11] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *PAMI*, 32(8):1362–1376, 2010.

[12] Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Manhattan-world stereo. In *Proc. CVPR*, 2009.

[13] E. Gobbetti G. Cuccuru, F. Marton, R. Pajarola, and R. Pintus. Fast low-memory streaming mls reconstruction of point-sampled surfaces. In *Proc. GI, Canadian Information Processing Society*, 2009.

[14] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *Proc. CVPR*, 2007.

[15] E.S.L. Gastal and M.M. Oliveira. Domain transform for edge-aware image and video processing. *ACM Transactions on Graphics*, 30(4):69:1–69:12, 2011.

[16] M. Goesele, B. Curless, and S.M. Seitz. Multi-view stereo revisited. In *Proc. CVPR*, 2006.

[17] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S.M. Seitz. Multi-view stereo for community photo collections. In *Proc. ICCV*, 2007.

[18] M. Habbecke and L. Kobbelt. A surface-growing approach to multi-view stereo reconstruction. In *Proc. CVPR*, 2007.

[19] X. Hu and P. Mordohai. Least commitment, viewpointbased, multi-view stereo. In *3DIMPVT*, 2012.

[20] M. Jancosek, A. Shekhovtsov, and T. Pajdla. Scalable multi-view stereo. In *Proc. ICCV*, 2009.

[21] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proc. Symp. Geometry Processing*, 2006.

[22] K. Kolev, T. Pock, and D. Cremers. Anisotropic minimal surfaces integrating photoconsistency and normal information for multiview stereo. In *Proc. ECCV*, 2010.

[23] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proc. ECCV*, 2002.

[24] J. Kopf, M. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 26(3), 2007.

[25] A. Kuhn, H. Hirschmueller, and Helmut Mayer. Multi-resolution range data fusion for multi-view stereo reconstruction. In *Proc. GCPR*, 2013.

[26] P. Labatut, J. Pons, and Keriven R. Robust and efficient surface reconstruction from range data. *Computer Graphics Forum*, 28:2275–2290, 2009.

[27] M. Lang, O. Wang, T. Aydin, A. Smolic, and M. Gross. Practical temporal consistency for image-based graphics applications. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 31(4), 2012.

[28] J. Li, E. Li, Y. Chen, and L. Xu. Bundled depth-map merging for multi-view stereo. In *Proc. CVPR*, 2010.

[29] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nister, and M. Pollefeys. Real-time visibility-based fusion of depth maps. In *Proc. ICCV*, 2007.

[30] P.J. Narayanan, P. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *Proc. ICCV*, 1998.

[31] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool. Metric 3d surface reconstruction from uncalibrated image sequences. In *Proc. SMILE Workshop*, pages 138–153, 1998.

[32] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *IJCV*, 59(3):207–232, 2004.

[33] C. Schroers, H. Zimmer, L. Valgaerts, A. Bruhn, O. Demetz, and J. Weickert. Anisotropic range image integration. In *DAGM*, 2012.

[34] S.M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR*, 2006.

[35] S.N. Sinha, P. Mordohai, and M. Pollefeys. Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. In *Proc. ICCV*, 2007.

[36] N. Snavely, S.M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):835–846, 2006.

[37] P. Song, X. Wu, and M. Wang. Volumetric stereo and silhouette fusion for image-based modeling. *The Visual Computer*, 26:1435–1450, 2010.

[38] C. Strecha, W. von Hansen, L. Van Gool, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proc. CVPR*, 2008.

[39] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *PAMI*, 32(5):815–830, 2010. http://cvlab.epfl.ch/cms/site/cvlab2/lang/en/software/daisy.

[40] C. Zach. Fast and high quality fusion of depth maps. In *3DPVT*, 2008.

[41] A. Zaharescu, E. Boyer, and R. Horaud. Transformesh: A topology-adaptive mesh-based approach to surface evolution. In *Proc. ACCV*, 2007.

[42] E. Zheng, E. Dunn, R. Raguram, and J.-M. Frahm. Efficient and scalable depthmap fusion. In *Proc. BMVC*, 2012.