Fast Guided Global Interpolation for Depth and Motion

Yu Li[1], Dongbo Min[2], Minh N. Do[3], and Jiangbo Lu[1] ⋆

[1] Advanced Digital Sciences Center, Singapore
[2] Chungnam National University, Korea
[3] University of Illinois at Urbana-Champaign, US

Abstract. We study the problems of upsampling a low-resolution depth map and interpolating an initial set of sparse motion matches, with the guidance from a corresponding high-resolution color image. The common objective for both tasks is to densify a set of sparse data points, either regularly distributed or scattered, to a full image grid through a 2D guided interpolation process. We propose a unified approach that casts the fundamental guided interpolation problem into a hierarchical, global optimization framework. Built on a weighted least squares (WLS) formulation with its recent fast solver – fast global smoothing (FGS) technique, our method progressively densifies the input data set by efficiently performing the cascaded, global interpolation (or smoothing) with alternating guidances. Our cascaded scheme effectively addresses the potential structure inconsistency between the sparse input data and the guidance image, while preserving depth or motion boundaries. To prevent new data points of low confidence from contaminating the next interpolation process, we also prudently evaluate the consensus of the interpolated intermediate data. Experiments show that our general interpolation approach successfully tackles several notorious challenges. Our method achieves quantitatively competitive results on various benchmark evaluations, while running much faster than other competing methods designed specifically for either depth upsampling or motion interpolation.

Keywords: Image-guided interpolation, depth upsampling, optical flow

1 Introduction

Dense depth or optical flow maps often serve as a fundamental building block for many computer vision and computational photography applications, *e.g.*, 3D scene reconstruction, object tracking, video editing, to name a few. An active range sensing technology such as time-of-flight (ToF) cameras has been recently advanced, emerging as an alternative to obtaining a depth map. It provides 2D depth maps at a video rate, but the quality of ToF depth maps is not as good as that of a high-quality color camera. The depth map is of low-resolution and noisy,

⋆ This study is supported by the HCCS grant at the ADSC from Singapore's Agency for Science, Technology and Research (A*STAR). Corresponding author: J. Lu

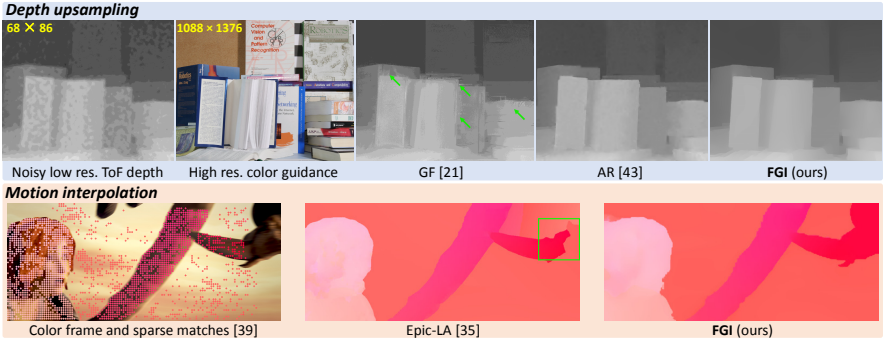

Fig. 1. Using the same guided interpolation pipeline, our technique gives strong results for two tasks: (top) depth upsampling and (bottom) optical flow field interpolation. Local or non-local methods (*e.g.* GF [21] and Epic-LA [35]) are usually efficient, but suffer from limitations like copying texture from the color guidance (green arrows) and inability to interpolate pixels in a distance (green rectangle). Methods using complicated models and global optimization (*e.g.* AR [43]) can obtain high quality results, but are often rather slow in computation. Our method, with a unified framework for both problems, is 1000× faster than AR [43] and even faster than local methods, while yielding competitive results when compared with state-of-the-art task-specific methods.

and thus a post-processing for depth upsampling is usually required to enhance the quality of depth maps. Over the years, numerous approaches for optical flow estimation have also been developed, but several challenging issues still remain, including large displacements, non-rigid fine motion, large occlusion, and flow boundaries. To address these issues, modern optical flow approaches often use a discriminative descriptor (or patch) matching to estimate sparse or quasi-dense motion matches. These important anchor points are used to interpolate a dense flow map, which is then embedded into subsequent optimization procedures.

One prevailing strategy for both depth upsampling and motion field interpolation is via a guided interpolation that uses an associated high-quality color image, exploiting the correlation between the color guidance and the depth/motion data. For depth upsampling, many methods based on the guided interpolation have been proposed with either local or global formulations [21, 43, 10, 12, 17, 22, 28–30, 34, 32]. Similarly, the guided motion field interpolation has been actively adopted as a key element in state-of-the-art optical flow approaches [35, 6, 39, 14, 26, 25, 8]. Though both tasks share the same goal of densifying a set of sparse input data points to a full image grid, *most existing interpolation approaches have been developed in isolation, tailored to either the depth upsampling or motion densification tasks due to different characteristics of two data sets.* For instance, ToF depth observations are noisy but regularly distributed in the high-resolution image grid, while sparse motion matches after outlier removal are typically reliable but highly scattered with a varying density of valid motion data across an image. In addition, existing interpolation methods are usually complicated and computationally inefficient.

We propose a *unified* approach to cast the fundamental guided interpolation (or densification) problem for both depth and motion data into a hierarchical, global optimization framework. Leveraging a recent fast global smoothing (FGS) technique [31] based on a weighted least squares (WLS) formulation [16], our method progressively densifies the input data set by efficiently performing a cascaded, guided global interpolation (or smoothing). While most existing approaches for depth upsampling and motion interpolation primarily rely on the color guidance image, the proposed method alternates the color image and an interpolated intermediate depth or flow map as the guidance. As a result, our cascaded scheme effectively addresses the potential structure inconsistency between the sparse input data and the guidance image, while preserving depth or motion discontinuities. To prudently select reliable new data points to augment the input sparse data, we evaluate the consensus between the interpolated data points using guidances and the data points from a spatial interpolation. Fig. 1 shows example results from our method. The contributions of this paper include:

- We propose a general fast guided interpolation (FGI) approach for both 1) noisy but regularly distributed depth maps and 2) typically reliable but highly scattered motion data.
- The proposed method successfully tackles several challenges such as texture-copy artifacts and loss of depth discontinuities in depth upsampling, and also large occlusions and motion boundaries in optical flow through a cascaded, guided global interpolation framework with alternating guidances.
- It achieves quantitatively competitive results on both tasks, while running much faster than state-of-the-art methods designed *specifically* for depth upsampling (over 600× faster) or motion interpolation (over 2× faster).
- Our technique is also generally applicable to other edge-aware filters such as the guided filter [21], and is shown to improve their interpolation quality.

1.1 Related Work

We review related work on depth upsampling and motion interpolation based on the guided interpolation. Other interpolation tasks (*e.g.*, spline fitting or single image super-resolution) without a guidance signal are beyond this paper's scope.

Depth upsampling: In an early work, Diebel and Thrun [12] cast a depth upsampling problem into a MRF formulation and solved it using a conjugate gradient method, but it tends to generate oversmooth results and is also sensitive to noise. Lu *et al.* [29] proposed an improved MRF-based depth upsampling method, but it is computationally expensive due to a complex global optimization. In [34], a non-local means (NLM) regularization term was additionally used in the MRF optimization. Ferstl *et al.* [17] defined the depth upsampling as a convex optimization problem using a high-order regularization term, called total generalized variation (TGV), which enforces piecewise affine results. An adaptive color-guided auto-regressive (AR) model [43] was proposed by formulating the depth upsampling task into a minimization of AR prediction errors, producing satisfactory results on real depth data. As filtering-based approaches, the joint bilateral upsampling (JBU) [22] was proposed to upsample a low-resolution depth

map by applying the bilateral filtering [38] with a guidance of a high-resolution color image. Afterwards, numerous filtering-based methods have been proposed using edge-aware filtering techniques, *e.g.* guided filter (GF) [21], cross-based local multipoint filter (CLMF) [30], and joint geodesic filter (JGF) [28]. The weighted mode filter (WMF) upsamples a low-resolution depth map by estimating a dominant mode from a joint histogram computed with an input depth data and a color guidance image [32]. One interesting work on devising a noise-aware filter [10] for depth upsampling took into account an inherent noisy nature of a depth map, preventing undesired texture-copy artifacts in the output depth map. Recently, Shen *et al.* [36] dealt with inconsistent structures existing in a pair of input signals *e.g.* NIR and RGB images with the concept of mutual-structure. But, this method focuses on the tasks such as joint restoration rather than tackling specific challenges in depth upsampling or motion interpolation, where the input data points are highly sparse.

Motion field interpolation: Modern optical flow algorithms have often used an interpolated motion data at an intermediate step for dealing with large displacement optical flow estimation. A set of sparse, reliable correspondences, first computed using a discriminative descriptor (or patch) matching, is interpolated and used as dense inputs for subsequent optimization steps. Using a non-local, approximated geodesic averaging or affine estimation, Revaud *et al.* [35] proposed an effective sparse-to-dense interpolation scheme termed 'EpicFlow', where sparse motion matches are computed from the deep matching method [39]. The interpolated output flow field was further refined through a variational energy minimization. The same interpolation and optimization pipeline was recently adopted in [6] to densify reliable flow fields after outlier filtering. However, EpicFlow solely counts on color edges detected with a structured edge detector (SED) [13] as the guidance for its interpolation, which are prone to the known issues of weak color boundaries or erroneous texture transferring in sparse data interpolation. Drayer and Brox proposed a combinatorial refinement of the initial matching [14], and then applied it to modern optical flow algorithms [35, 8, 39]. They utilized sparse motion matches as an input for motion interpolation and refinement. Though improving the flow accuracy, the whole process of [14] is still complex and slow. A sparse-to-dense interpolation approach was also used in [26], but a costly optimization process is involved in finding a set of initial matches and computing an affine model independently for each pixel.

2 Image Smoothing with WLS

Our pipeline is built on edge-aware image smoothing techniques, *e.g.* [38, 15, 16, 21, 18, 37, 7, 31]. In this paper, we choose the weighted least squares (WLS) formulation [16] as our fundamental engine based on the two considerations: (1) it uses a global optimization formulation that overcomes the limitation (*e.g.* halo artifacts) of edge-aware local filters [21, 18, 30] in the smoothing process; (2) the recent proposed fast WLS solver [31] shows comparable runtime to fast local filters. Such favorable properties allow us to exploit it with full extent in a

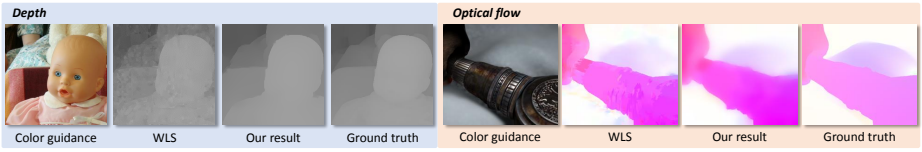

Fig. 2. Limitations of the WLS formulation [31] in depth upsampling (left) and motion match interpolation (right). Spurious structures mistakenly transferred from the color guidance are clearly visible in both cases, while our interpolator generates interpolation results that resemble the ground truth. (Best viewed in electronic version.)

hierarchical, multi-pass framework, meeting our requirements in terms of accuracy and efficiency, which would not be practical previously. Besides WLS, our framework is generally applicable to other edge-aware smoothing filters (Sec. 4).

In a WLS based smoothing approach, given an input image f and a guidance image g, an output image u is computed by minimizing the objective $\mathcal{E}$ as:

$$\mathcal{E}(u) = \sum_p (u_p - f_p)^2 + \lambda \sum_p \sum_{q \in \mathcal{N}_4(p)} w_{p,q}(g)(u_p - u_q)^2, \tag{1}$$

where $\mathcal{N}_4(p)$ consists of four neighbors for p. $w_{p,q}$ is a spatially varying weight function measuring how similar two pixels p and q are in the guidance image g. The weight λ balance the data term and the regularization term. This objective can also be written in a matrix/vector form:

$$\mathcal{E}(\mathbf{u}) = (\mathbf{u} - \mathbf{f})^\top (\mathbf{u} - \mathbf{f}) + \lambda\, \mathbf{u}^\top \mathbf{A} \mathbf{u}\,. \tag{2}$$

The matrix $\mathbf{A} = \mathbf{D} - \mathbf{W}$ is usually referred to as a Laplacian matrix. $\mathbf{D}$ is a degree matrix, where $\mathbf{D}(i,i) = \sum_{j \in N(i)} w_{i,j}(g)$, and $\mathbf{D}(i,j) = 0$ for $i \neq j$. $\mathbf{W}$ is an adjacency matrix defined with $\{w_{i,j|j \in N(i)}\}$. Eq. (2) is strictly convex, and thus $\mathbf{u}$ is obtained by solving a linear system with a large sparse matrix as

$$(\mathbf{E} + \lambda \mathbf{A})\mathbf{u} = \mathbf{f}, \tag{3}$$

where $\mathbf{E}$ is an identity matrix. Though several methods [23, 24] have been proposed for efficiently solving the linear system Eq. (3), they are an order of magnitude slower than the local filters [18, 21]. Recently, a fast global smoothing (FGS) technique [31] was proposed as an efficient alternative to compute the solution of Eq. (2). The key idea is to approximate the large linear system by solving a series of 1D sub-systems in a separable way. The 1D sub-systems are defined with horizontal or vertical passes for an input 2D image. It has an efficient solution obtained by the Gaussian elimination algorithm. Specifically, its runtime for filtering a 1M pixel RGB image on a single CPU core is only $\mathbf{0.1}s$ [31].

Limitations of WLS in interpolation. The WLS based formulation was originally proposed for computational photography and image processing applications. We found that directly applying it to depth or motion interpolation, where an input data is highly sparse and/or noisy, does not yield an excellent

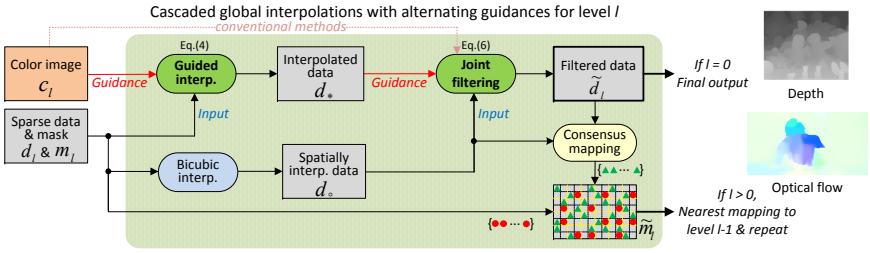

Fig. 3. Our fast guided interpolation (FGI) framework, taking sparse input data d_l, its mask m_l and a color guidance c_l as inputs for the level l. The guided interpolation/filtering with alternating guidances and consensus checking iterate a few times in a hierarchical way to get to the result at final resolution. In contrast, existing methods *e.g.* [43] directly take the bicubic estimation and the color guidance to infer the final results in a single pass joint filtering/optimization manner (denoted in dashed lines).

result. Fig. 2 shows the interpolation result based on the single scale WLS. It is because the one-pass optimization at a single scale is insufficient for interpolating highly sparse and noisy data with very low-density regions (*e.g.* 16× upsampling for depth data, namely only *one* raw depth value is available in a $16 \times 16 = 256$ region in the targeted full scale). This signal recovery instability caused by the highly deficient data observation matrix can also be understood from a theoretical perspective of the conditioning of linear systems [19, 43]. In such a case, large gaps between observed sparse data points are forced to be filled by primarily counting on the structures in the color guidance. However, the color guidance is not always faithfully correlated with the data to be interpolated, thus often producing texture-copy artifacts or over-smoothing around weak edges.

3 Fast Guided Global Interpolation

As shown in Fig. 2, a single-pass WLS-based optimization often fails to generate high-quality interpolation results, when the input data is too sparse or scattered on a full image grid. To address these issues especially in depth upsampling and motion interpolation tasks in a unified manner, we propose a hierarchical, multi-pass guided interpolation framework. Specifically, we address the challenges in an iterative coarse-to-fine manner that divides the problems into a sequence of interpolation tasks with smaller scale factors, and gradually fills the large gap between the sparse measurement and the dense data.

Suppose the number of levels used in the hierarchical structure is L. We start the guided interpolation from the coarsest level ($l = L - 1$), and progressively generates reliable data points to densify the sparse input data. This process is repeated until the finest level ($l = 0$) is reached. Fig. 3 illustrates the procedure of the proposed framework at the l^{th} level. At each level, we first interpolate

the sparse input d_l[4] by performing the WLS optimization using a corresponding color image c_l as the guidance and also a simple bicubic interpolation technique, respectively. Then, another WLS is applied with the interpolated dense data d_* from the first WLS interpolation output as the guidance and the bicubic interpolated map as the input signal. Finally, we select reliable points via consensus checking, and pass the augmented data points to the next level $l-1$.

For a sparse data input, we use a mask m_l ($l = 0, ..., L-1$) to denote the data observation or constraint map whose elements are 1 for pixels with valid data and 0 otherwise. At each level, we upsample the signal by a factor of 2. We also pre-compute a downsampled color image c_l for each level from a high resolution color image c such that $c_0 = c, c_l = c_{l-1} \downarrow (l = 1, 2, \ldots, L-1)$, where $\downarrow$ denotes a downsampling operation by a factor of 2. A sparse input at the starting level ($l = L-1$) can be depth data from a low resolution depth map or irregular sparse motion matches mapped from descriptor matching methods (*e.g.* [39]).

3.1 Cascaded Filtering with Alternating Guidances

For a progressively densified input data d_l at a certain level l, our technique performs two cascaded WLS by alternating the color image c_l and an intermediate interpolated depth or flow map d_* as the guidance (see Fig. 3). For the first WLS-based interpolation using the color guidance, the sparse input data d_l is quickly densified at the current scale. In this pass, the sparse data is interpolated in accordance with the color structures. This process may introduce spurious structures to the interpolated data d_* (*e.g.* texture-copying effects) due to inconsistent structures between the color and sparse input depth/motion data, but d_* interpolated from the sparse input data d_l contains much weaker texture patterns than the original guidance signal c_l (see d_* in Fig. 4). Therefore, we propose to append the second modified WLS smoothing step using the newly interpolated data d_* as the guidance. During this second pass, the WLS optimization is solved with a more faithful guidance of the same modality (*i.e.* d_* rather than c_l), while being subject to dense data constraints from the bicubic-upsampled data $d_\circ$. We find this cascaded scheme effectively addresses the potential structure inconsistency between the sparse input data and the guidance image, while preserving true depth or motion discontinuities (see Fig. 2 & 4).

1^{st} **WLS using c_l as the guidance.** When the sparse input data $\mathbf{d}_l$ and the guidance color image $\mathbf{c}_l$ are given at the l^{th} level, the first WLS step, minimizing the following objective, is invoked to obtain an intermediate dense output $\mathbf{d}_*$:

$$\mathcal{E}(\mathbf{d}_*) = (\mathbf{d}_* - \mathbf{d}_l)^\top \mathbf{M}_l (\mathbf{d}_* - \mathbf{d}_l) + \lambda_1 \, \mathbf{d}_*^\top \mathbf{A}_{c_l} \mathbf{d}_* \, , \tag{4}$$

where $\mathbf{M}_l$ is a diagonal matrix with its elements given by the mask map m_l. $\mathbf{A}_{c_l}$ denotes the spatially varying Laplacian matrix defined by the guidance image c_l at the l^{th} level. Unlike the image smoothing task using a dense input in (2), the input data $\mathbf{d}_l$ is sparse, and thus directly minimizing it in a separable manner

[4] Hereinafter we denote the corresponding vectorized form of d as $\mathbf{d}$.

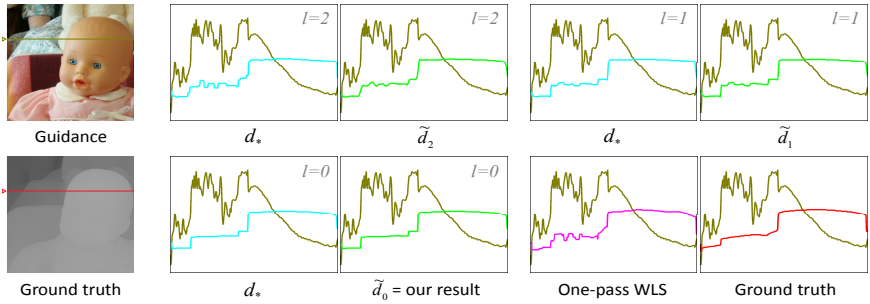

Fig. 4. Comparison of the 1D scanline results obtained by our cascaded WLS steps over three levels $l = 2, 1, 0$, and that obtained by the one-pass WLS. In all the subplots (right), the corresponding color signal is in *olive green*, while other kinds of signals are in different colors. The same is observed for optical flow interpolation.

leads to unstable results. Instead, as in [31, 18], we compute the solution $\mathbf{d}_*$ with

$$\mathbf{d}_*(p) = \frac{((\mathbf{E} + \lambda \mathbf{A}_{c_l})^{-1}\mathbf{d}_l)(p)}{((\mathbf{E} + \lambda \mathbf{A}_{c_l})^{-1}\mathbf{m}_l)(p)} \ , \tag{5}$$

where $\mathbf{m}_l$ denotes the corresponding vectorized form of m_l. The WLS is applied twice to $\mathbf{d}_l$ and $\mathbf{m}_l$, respectively.

2^{nd} WLS using d_* as the guidance. Here, the input data $\mathbf{d}_\circ$ is obtained by a bicubic interpolation of $\mathbf{d}_l$ at the l^{th} level, and the guidance signal is the intermediate interpolated data $\mathbf{d}_*$. A similar objective is minimized as:

$$\mathcal{E}(\tilde{\mathbf{d}}_l) = (\tilde{\mathbf{d}}_l - \mathbf{d}_\circ)^\top (\tilde{\mathbf{d}}_l - \mathbf{d}_\circ) + \lambda_2 \tilde{\mathbf{d}}_l^\top \mathbf{A}_{d_*} \tilde{\mathbf{d}}_l \ , \tag{6}$$

where $\mathbf{A}_{d_*}$ denotes the Laplacian matrix defined by d_*. Note that the input data $\mathbf{d}_\circ$ is dense in this pass, while $\mathbf{d}_l$ is sparse in the 1^{st} WLS.

To give more intuitions of the proposed cascaded filtering process with alternating guidances, we show in Fig. 4 the processing results for one scanline (extracted from real images in Fig. 2): d_* and $\tilde{d}_l$ from the 1^{st} and 2^{nd} WLS steps, iterating from $l = 2$ down to $l = 0$. Over the iterations, both of our intermediate guidance signal d_* and the 2^{nd} WLS output $\tilde{d}_l$ are progressively improved, with the final output $\tilde{d}_0$ close to the ground truth. In contrast, the result of applying the one-pass WLS contains spurious color structures (though attenuated), which are mistakenly transferred from highly-varying texture regions.

It is worth noting the difference from the rolling guidance filter (RGF) [44], though using progressively improved guidance signals appears somewhat related. First, they are developed for different objectives: RGF focuses on removing small structural details for image smoothing, but our FGI tackles notorious interpolation issues such as inconsistent structures between a color guidance image and a sparse depth or flow map. Second, RGF needs to carefully set the target scale parameter for its Gaussian prefiltering, but inconsistent structures across different signal modalities are often not small. Third, RGF has the limitation of blunting image corners, while FGI preserves important depth or motion structures.

3.2 Consensus-Based Data Point Augmentation

Thanks to the hierarchical interpolation framework, the proposed algorithm is not required to generate a fully dense data set for any intermediate level, which may propagate some unreliable data points to the next level otherwise. Therefore, we can be prudent in selecting reliable new data points to augment the input sparse data set d_l. As the last consistency checking in this current iteration l, we evaluate the consensus between the interpolated data points $\tilde{d}_l$ obtained from alternating guidance filtering and the data points $d_\circ$ from a direct bicubic interpolation. In fact, without using the color guidance in the spatial interpolation process, $d_\circ$ is free from color texture copying artifacts, though it has difficulties in restoring sharp edges/structures. Therefore, if we impose a consensus checking in the interpolated data between $d_\circ$ and $\tilde{d}_l$, those unwanted color texture patterns in $\tilde{d}_l$ will not be chosen. This cautious design helps preventing those new data points of low confidence (*e.g.* undesired texture-copy patterns) from contaminating the next interpolation process.

Our consensus-based data point augmentation proceeds in a non-overlapping patch fashion. For each pixel q in the patch we check the consistency between the interpolated data points $\tilde{d}_l$ and the bicubic upsampled data points $d_\circ$ as $\delta(q) = \|\tilde{d}_l(q) - d_\circ(q)\|$. After the consensus checking we pick the most consistent data location in the current patch (*i.e.* with the smallest $\delta(q)$ and also smaller than a preset threshold τ) and add this location to the data mask map $\tilde{m}_l$. Fig. 3 illustrates this data augmentation process by denoting new data points in m_l^a as green triangles and initial sparse data points m_l as red dots. By selecting at most one new data point in each patch, we intend to avoid propagating the interpolation error to the next level. We use 2×2 patches in this paper.

3.3 Computational Complexity

The computational cost is mainly from solving the WLS objective in (3), as other parts have marginal computational overhead. To compare the complexity of our method with a single scale WLS based interpolation, we first count how many times the linear system is solved in both methods. In each level, the WLS based guided interpolation of (4) requires solving the linear system twice as in (5)– one for the input signal and one for the binary index signal, after which the final solution is obtained by an element-wise division . The WLS of (6) needs to solve the linear system once as its input $\mathbf{d}_\circ$ is dense. Thus, the linear system solver is applied 3 times at each level of our method. Since the interpolation grid is progressively rescaled with a factor of 2, our hierarchical framework increases the total computational complexity at most by $1/(1-1/4) = 4/3$. One can expect our hierarchical approach to have $(2+1) \times 4/3 = \mathbf{4}$ passes of executing the linear system solver, while the single scale WLS based interpolation needs **2** passes.

4 Experiments

We perform our experiments on a PC with Intel Xeon CPU (3.50 GHz) and 16 GB RAM. The implementation was in C++. For minimizing the WLS objective

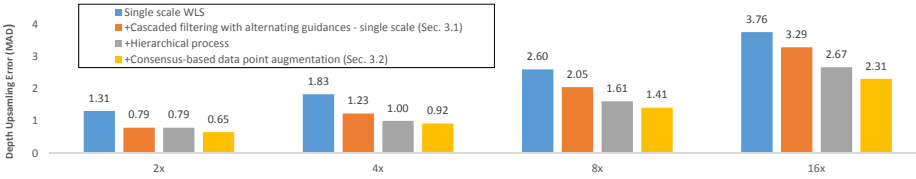

Fig. 5. The effect of each component of our pipeline evaluated on depth upsampling.

Table 1. Quantitative comparison (MAD) on ToF-like synthetic datasets [43]. Best results are in **bold**, and the second best are underlined.

Method	*Art*				*Book*				*Moebius*				*Reindeer*				*Laundry*				*Dolls*				**Average**			
	2x	4x	8x	16x	2x	4x	8x	16x	2x	4x	8x	16x	2x	4x	8x	16x	2x	4x	8x	16x	2x	4x	8x	16x	2x	4x	8x	16x
Bicubic	3.52	3.84	4.47	5.72	3.30	3.37	3.51	3.82	3.28	3.36	3.50	3.80	3.39	3.52	3.82	4.45	3.35	3.49	3.77	4.35	3.28	3.34	3.47	3.72	3.35	3.49	3.76	4.31
JGF [28]	2.36	2.74	3.64	5.46	2.12	2.25	2.49	3.25	2.09	2.24	2.56	3.28	2.18	2.40	2.89	3.94	2.16	2.37	2.85	3.90	2.09	2.22	2.49	3.25	2.17	2.37	2.82	3.85
GF [21]	1.49	1.97	3.00	4.91	0.8	1.22	1.95	3.04	1.18	1.90	2.77	3.55	1.29	1.99	2.99	4.14	1.28	2.05	3.04	4.10	1.19	1.94	2.80	3.50	1.21	1.85	2.76	3.87
CLMF0[30]	1.19	1.77	2.95	4.91	0.90	1.48	2.38	3.36	0.87	1.44	2.32	3.3	0.96	1.56	2.54	3.85	0.94	1.55	2.50	3.81	0.96	1.54	2.37	3.25	0.97	1.56	2.51	3.75
MRF+nlm[34]	1.69	2.40	3.60	5.75	1.12	1.44	1.81	2.59	1.13	1.45	1.95	2.91	1.20	1.60	2.40	3.97	1.28	1.63	2.20	3.34	1.14	1.54	2.07	3.02	1.26	1.68	2.34	3.60
TGV[17]	0.82	1.26	2.76	6.87	0.50	0.74	1.49	2.74	0.56	0.89	1.72	3.99	0.59	0.84	1.75	4.40	0.61	1.59	1.89	4.16	0.66	1.63	1.75	3.71	0.62	1.16	1.89	4.31
AR [43]	**0.76**	**1.01**	**1.70**	**3.05**	**0.47**	**0.70**	1.15	1.81	**0.46**	**0.72**	**1.15**	1.92	**0.48**	**0.80**	**1.29**	**2.02**	**0.51**	**0.85**	**1.30**	**2.24**	**0.59**	**0.91**	1.32	2.08	**0.55**	**0.83**	**1.32**	**2.19**
WLS [31]	1.34	1.90	2.95	4.63	1.25	1.70	2.39	3.29	1.34	1.92	2.66	3.56	1.47	2.05	2.82	4.09	1.11	1.55	2.24	3.49	1.34	1.85	2.55	3.50	1.31	1.83	2.60	3.76
FGI (ours)	0.79	1.17	2.01	3.65	0.58	0.80	**1.13**	**1.75**	0.58	0.80	**1.15**	**1.71**	0.65	0.89	1.36	2.37	0.65	0.97	1.49	2.43	0.67	**0.91**	**1.31**	**1.95**	0.65	0.92	1.41	2.31

function, we use the FGS [31] solver provided on its project site [1] (also possible to use its OpenCV 3.1 function [2]). We will make our code publicly available. In all the experiments, we fix the smooth weights $\lambda_1 = 30.0^2, \lambda_2 = 10.0^2$, and the consensus checking threshold $\tau = 15(\text{depth})/1(\text{motion})$. For the affinity weight $w_{p,q}(g)$ in (1), we follow [31] to set $w_{p,q}(g) = \exp(-\|g_p - g_q\|/\sigma)$ with $\sigma = 0.005$.

4.1 Depth Upsampling Results

Pipeline Validation. First, we present a quick study on our pipeline design given in Sec. 3 on the depth upsampling task with the dataset provided by [43]. Starting from the single pass WLS based interpolation, we gradually add in new features until getting to our full pipeline. The comparison of the average error in the upsampled depth maps is plotted in Fig. 5. As can be seen, adding the cascaded filtering with one more WLS using alternating guidance in the single scale leads to lower errors in depth upsampling. Note, however, the gain from this step is almost fixed for all upsampling factors. To handle more challenging cases with high upsampling rates (*e.g.* 8 or 16), employing the hierarchical process yields better results, which meets our expectation. The last module tested is the consensus-based data point augmentation. This strategy further reduces the upsampling errors. Overall, our whole pipeline obtains much better depth upsampling results than the direct single pass WLS interpolation (see also Fig. 2).

We now evaluate the performance of depth upsampling with different edge-aware smoothing filters. We take the popular GF [21] for this test. The average error of a single pass interpolation with GF are 1.31/1.54/2.04/3.12 for upscaling rate 2/4/8/16. When using our pipeline with GF (*i.e.* replacing all the WLS steps with GF), the results are 1.06/1.21/1.63/2.59. The improvements confirm that our framework is generic to other edge-aware filtering techniques. We choose FGS [31] as our fundamental block for its best efficiency and accuracy.

Table 2. Average runtime (in sec) to upsample by 4× an input depth map 272×344.

Method	MRF+nlm[34]	TGV [17]	AR [43]	GF [21]	CLMF0 [30]	WLS [31]	FGI (ours)
Runtime(s)	170	420	900	1.26	2.4	0.32	0.65

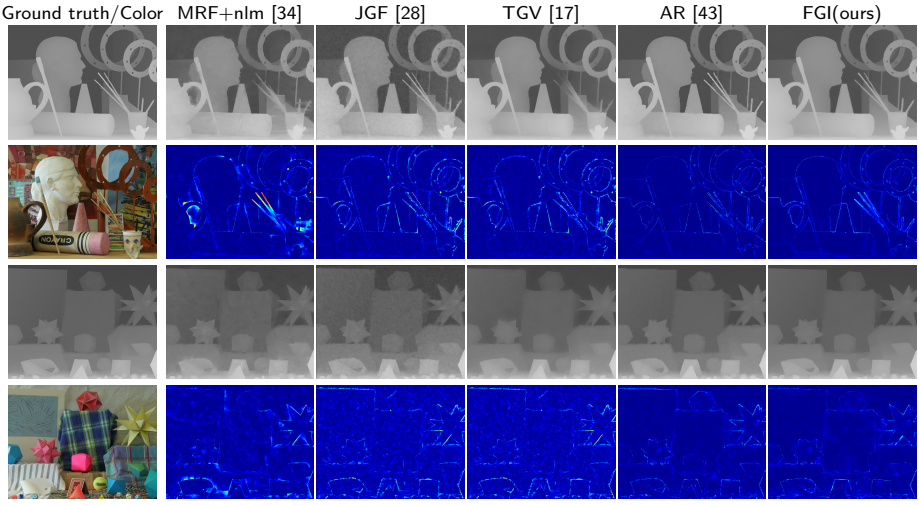

Fig. 6. Visual comparison on 8× upsampling results and error maps of *Art* and *Moebius* from the ToF-like synthetic dataset [43]. (Best viewed in electronic version.)

Results on ToF-like synthetic datasets [43]. We evaluate the proposed FGI method on a ToF depth upsampling task using the synthetic datasets provided by [43]. They used six datasets from Middlebury benchmarks [3] to simulate ToF-like depth degradation by adding noise and performing downsampling with four different scales, *i.e.* $2, 4, 8, 16$. Our FGI uses $L = 1, 2, 3, 4$ levels architecture for four different upsampling scales. Table 1 reports the Mean Absolute Difference (MAD) between ground truth depth maps and the results by various depth upsampling methods including ours. The proposed method clearly outperforms several existing methods like CLMF0 [30], JGF [28], MRF+nlm [34] and TGV [17] that used different color-guided upsampling or optimization techniques. Our method also yields much smaller error rates than the single-pass WLS interpolation, validating the effectiveness of our hierarchical structure. Finally, when compared with the state-of-the-art AR method [43] over all test image sets for challenging higher upsampling rates $(8, 16)$, our FGI actually yields more accurate depth maps on half of them, *i.e. Book*, *Moebius*, and *Dolls*. Though slightly worse than AR [43] in terms of the MAD, our FGI is the second best among all leading methods, and runs over **1000**× faster than AR [43].

Table 2 summarizes the runtime of various methods whose source codes are available and timed on our PC. Generally, the methods using global optimizations *e.g.* MRF+nlm [34], TGV [17] and AR [43] come with much higher computational costs. GF [21] and CLMF0 [30], as local filtering methods, take less time

Table 3. Quantitative results (MAD in millimeter) on the real ToFMark datasets [4].

	Bicubic	JBU [22]	GF [21]	JGF [28]	TGV [17]	**FGI (ours)**
Books	16.23	16.03	15.74	17.39	12.36	13.03
Devil	16.66	27.57	27.04	19.02	14.68	15.09
Shark	17.78	18.79	18.21	18.17	15.29	15.82

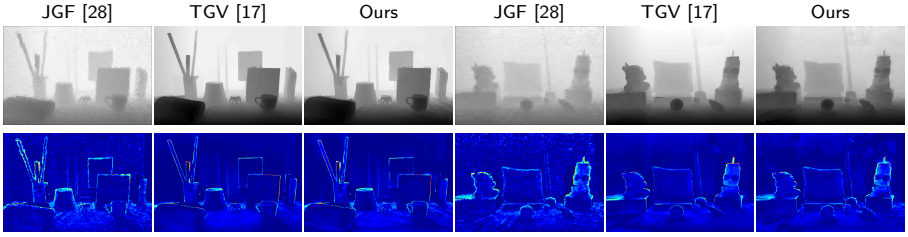

Fig. 7. Depth upsampling results and error maps of *Books* and *Devil* in ToFMark [4].

to upsample a depth map, but they are still slower than our fast optimization-based method. The single-pass WLS with the FGS solver [31] takes only $0.32s$. Our FGI also takes advantage of the FGS solver [31] and is fast. Since we use different numbers of levels for different upsample rates, its runtimes vary slightly for them, *i.e.* $0.51/0.65/0.69/0.73s$ for upscaling rate 2/4/8/16. The runtime results of the single-scale WLS and our FGI are also consistent with the complexity analysis in Sec. 3.3. Another efficient method JGF [28] reports $0.33s$ in upsampling $8\times$ to $0.4M$ depth images, but FGI takes $0.19s$ on the same size.

Fig. 6 shows two visual comparisons of depth maps upsampled by different methods on this synthetic dataset. The results of MRF+nlm [34] fail to recover the depth for the thin structures in the *Art* case and show texture-copy artifacts in the *Moebius* case. The depth maps by JGF [28] contain noticeable noise as it is designed without any consideration of the noise issue from depth sensors. A separate noise removal process may be applied before JGF to solve the noise problem while our method (like most leading depth upsampling methods) does not require such a separate pre-processing. It is clearly observed that among all methods compared, AR [43] and our FGI recover accurate depths in homogeneous regions and along depth boundaries, and preserve thin structures better than other methods. More visual results are given in the supplemental materials.

Results on the ToFMark datasets [4]. We further test on the ToFMark datasets [4] provided in the TGV paper [17] that contain three real ToF depth and intensity image pairs *i.e. Books, Devil, Shark*. The ToF depth maps of spatial resolution 120×160 are real depth values in millimeter (mm), while the intensity images are of size 610×810. Table 3 presents the quantitative results measured by MAD in mm. Our method outperforms prevailing methods like JBU [22], GF [21], JGF [28], and obtains performance quite close to TGV [17]. Fig. 7 shows the visual comparison of different methods. The depth recovered by JGF [28] again exhibits noticeable noise, while the results of TGV [17] and ours are much sharper and cleaner, but our FGI runs about **650**$\times$ faster than TGV [17].

Table 4. Performance comparison (EPE) on the Sintel training set [9].

Method	Clean	Final	Runtime	Method	Clean	Final	Runtime
Epic-NW [35]	3.17	4.55	0.80s	WLS [31]	3.23	4.68	0.21s
Epic-LA [35]	2.65	4.10	0.94s	**FGI (ours)**	2.75	4.14	0.39s

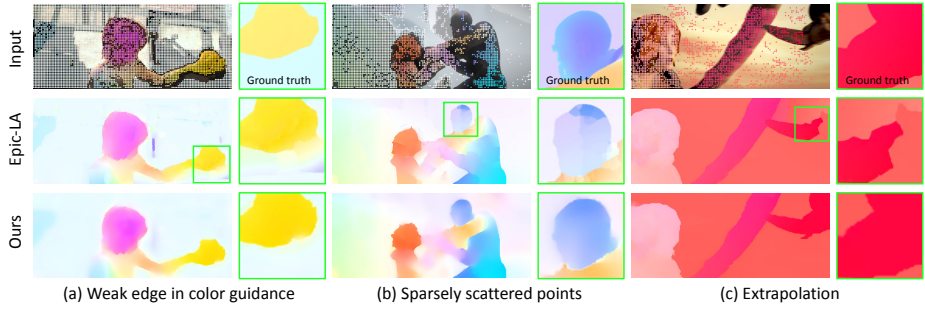

Fig. 8. Optical flow fields interpolated by our method and Epic-LA [35] on the Sintel datasets on three challenging cases. Please refer to Sec. 4.2 for more analysis.

4.2 Motion Field Interpolation for Optical Flow

We evaluate our motion interpolator using the MPI Sintel dataset [9], a modern optical flow evaluation benchmark with large displacement flow and complex non-rigid motions. The evaluation is conducted on two types of rendered frames, *i.e. clean* pass and *final* pass, where the final pass includes more complex effects such as specular reflections, motion blur, defocus blur, and atmospheric effects. We evenly sampled 331 frames from the whole training set and used them for evaluation and the rest of the frames were used to choose the best parameters. The number of levels L in the hierarchical structure is fixed to 3.

To generate a set of sparse matches, we adopt a leading descriptor-based matching algorithm – DeepMatching [39], one of the top methods on the Sintel benchmark. We perform the same match pruning step as EpicFlow [35] to remove unreliable matches, so the set of sparse matches used in our method and EpicFlow are exactly the same. After the pruning step, we usually can get 5000∼6000 reliable matches on 436×1024 color frames. This motion interpolation task from sparse data with about 1% density is challenging, especially considering the data points are not uniformly distributed. Note that besides [39], our framework is flexible to take in other choices of reliable motion matches *e.g.* [6].

For performance comparison on the 331 frames , we test with the single pass WLS [31], and also with both the locally-weighted affine (LA) and Nadaraya-Watson (NW) interpolators of EpicFlow [35], denoted as Epic-LA and Epic-NW. Table 4 reports the performance of these methods. Compared with the single pass WLS, our FGI shows improvements in the sparse-to-dense interpolation results, again demonstrating the effectiveness of our pipeline. Our method achieves a quantitative performance better than EpicFlow-NW and very close to EpicFlow-LA, while reducing the runtime of the interpolation process by over 50%.

Table 5. Performance comparison (EPE) on the Sintel testing benchmark [9].

	FlowFields[6]	EpicFlow[35]	PH-Flow[42]	**FGI (ours)**	Deep+R[14]	SPM-BP[27]	DeepFlow[39]	PCA-Layers[40]	MDP-Flow2[41]
Clean	3.748	4.115	4.388	4.664	5.041	5.202	5.377	5.730	5.837
Final	5.810	6.285	7.423	6.607	6.769	7.325	7.212	7.886	8.445

Fig. 8 compares optical flow fields interpolated by our method and EpicFlow-LA. The EpicFlow method uses the color edges detected by a state-of-the-art technique [13] as a guidance signal, but it is still unavoidable to produce undesired results with missing motion boundaries around weak color boundaries (Fig. 8a). Furthermore, the motion interpolation used in the EpicFlow involves finding a set of nearest matches (*e.g.* 100 matches) and generating a Voronoi diagram, and such hard decisions or assignments often tend to produce patch-wise discretization artifacts (Fig. 8b). To address these issues, a variational energy minimization is applied as a post-processing, but such problems are not fully resolved due to the local minima of the variational approach. More seriously, in extremely low density regions, it often fails to densify the flow map properly due to its essentially localized, approximated geodesic propagation strategy (Fig. 8c). In contrast, our approach does not detect color edges to directly guide the interpolation process, and also does not need an extra variational post-processing[5] thanks to the global optimization formulation using a hierarchical strategy.

Table 5 reports the quantitative evaluation on the Sintel test set [5]. On the benchmark, our FGI ranks the 8th for both clean and final passes among all 63 methods listed at the time of submission (Mar. 2016). Recent optical flow algorithms [35, 6, 14, 40] use sparse matching or dense approximate nearest neighbor fields to handle large displacement and usually perform better than conventional methods [41]. The full version of EpicFlow in this table takes the advantage of local-weighted affine (LA) fitting and a variational energy minimization, performing slightly better than our FGI. However, unlike these optical flow specific methods [35, 6, 14, 40], our proposed FGI is a generic, fast interpolator and does not need an extra post processing like variational optimization [35, 6].

5 Conclusion

This paper presented a hierarchical, cascaded WLS-optimization based technique that handles low-resolution and noisy depth upsampling and sparse motion match densification in a unified manner. Compared with the existing methods tailored specifically for one of these different tasks, our FGI achieves leading functional quality on benchmark evaluations as well as a highly efficient runtime over those top performers on each task. We used a basic WLS formulation as our key engine here to demonstrate its strength, but more robust sparse norms or an adaptively aggregated data term [31] can also be employed. Moreover, our technique has a potential advantage for further acceleration on GPUs and FPGA [33, 20, 11], offering a common engine for guided interpolation.

[5] We find adding it in FGI gives only marginal accuracy gain, unlike in EpicFlow [35].

References

1. Https://sites.google.com/site/globalsmoothing/
2. Http://docs.opencv.org/master/da/d17/group__ximgproc__filters.html
3. http://vision.middlebury.edu/stereo/
4. http://rvlab.icg.tugraz.at/tofmark/
5. http://sintel.is.tue.mpg.de/results
6. Bailer, C., Taetz, B., Stricker, D.: Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In: ICCV (2015)
7. Bao, L., Song, Y., Yang, Q., Yuan, H., Wang, G.: Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree. IEEE Trans. on Image Processing 23(2), 555–569 (2014)
8. Brox, T., Bregler, C., Malik, J.: Large displacement optical flow. In: CVPR (2009)
9. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: ECCV (2012)
10. Chan, D., Buisman, H., Theobalt, C., Thrun, S.: A noise-aware filter for real-time depth upsampling. In: ECCV Workshop (2008)
11. Chaurasia, G., Ragan-Kelley, J., Paris, S., Drettakis, G., Durand, F.: Compiling high performance recursive filters. In: High Performance Graphics (2015)
12. Diebel, J., Thrun, S.: An application of markov random fields to range sensing. In: NIPS (2005)
13. Dollár, P., Zitnick, C.L.: Structured forests for fast edge detection. In: ICCV. IEEE (2013)
14. Drayer, B., Brox., T.: Combinatorial regularization of descriptor matching for optical flow estimation. In: BMVC (2015)
15. Elad, M.: On the origin of the bilateral filter and ways to improve it. IEEE Trans. on Image Processing 11(10), 1141–1151 (2002)
16. Farbman, Z., Fattal, R., Lischinski, D., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. ACM Trans. Graph. 27(3) (2008)
17. Ferstl, D., Reinbacher, C., Ranftl, R., Rüther, M., Bischof, H.: Image guided depth upsampling using anisotropic total generalized variation. In: ICCV (2013)
18. Gastal, E.S.L., Oliveira, M.M.: Domain transform for edge-aware image and video processing. ACM Trans. Graph. 30(4) (2011)
19. Golub, G.H., Loan, C.F.V.: Matrix Computations. Johns Hopkins University Press (1996)
20. Greisen, P., Runo, M., Guillet, P., Heinzle, S., Smolic, A., Kaeslin, H., Gross, M.: Evaluation and FPGA implementation of sparse linear solvers for video processing applications. IEEE Trans. on Circuits and Systems for Video Technology 23(8), 1402–1407 (Aug 2013)
21. He, K., Sun, J., Tang, X.: Guided image filtering. In: ECCV (2010)
22. Kopf, J., Cohen, M.F., Lischinski, D., Uyttendaele, M.: Joint bilateral upsampling. ACM Trans. Graph. 26(3) (2007)
23. Koutis, I., Miller, G.L., Tolliver, D.: Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. CVIU 115(12), 1638–1646 (2011)
24. Krishnan, D., Fattal, R., Szeliski, R.: Efficient preconditioning of laplacian matrices for computer graphics. ACM Trans. Graph. (2013)
25. Lang, M., Wang, O., Aydin, T.O., Smolic, A., Gross, M.H.: Practical temporal consistency for image-based graphics applications. ACM Trans. Graph. 31(4), 34–1 (2012)

26. Leordeanu, M., Zanfir, A., Sminchisescu, C.: Locally affine sparse-to-dense matching for motion and occlusion estimation. In: ICCV (2013)
27. Li, Y., Min, D., Brown, M.S., Do, M.N., Lu, J.: SPM-BP: Sped-up patchmatch belief propagation for continuous MRFs. In: ICCV (2015)
28. Liu, M.Y., Tuzel, O., Taguchi, Y.: Joint geodesic upsampling of depth images. In: CVPR (2013)
29. Lu, J., Min, D., Pahwa, R.S., Do, M.N.: A revisit to MRF-based depth map super-resolution and enhancement. In: ICASSP (2011)
30. Lu, J., Shi, K., Min, D., Lin, L., Do, M.N.: Cross-based local multipoint filtering. In: CVPR (2012)
31. Min, D., Choi, S., Lu, J., Ham, B., Sohn, K., Do, M.N.: Fast global image smoothing based on weighted least squares. IEEE Trans. on Image Processing 23(12), 5638–5653 (2014)
32. Min, D., Lu, J., Do, M.N.: Depth video enhancement based on weighted mode filtering. IEEE Trans. on Image Processing 21(3), 1176–1190 (2012)
33. Nehab, D., Maximo, A., Lima, R.S., Hoppe, H.: GPU-efficient recursive filtering and summed-area tables. ACM Trans. Graph. 30(6), 176:1–176:12 (Dec 2011)
34. Park, J., Kim, H., Tai, Y.W., Brown, M.S., Kweon, I.: High quality depth map upsampling for 3D-ToF cameras. In: ICCV (2011)
35. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: EpicFlow: Edge-preserving interpolation of correspondences for optical flow. In: CVPR (2015)
36. Shen, X., Zhou, C., Xu, L., Jia, J.: Mutual-structure for joint filtering. In: ICCV (2015)
37. Talebi, H., Milanfar, P.: Nonlocal image editing. IEEE Trans. on Image Processing 23(10), 4460–4473 (2014)
38. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: IEEE Int. Conf. on Computer Vision. pp. 839–846 (1998)
39. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: ICCV (2013)
40. Wulff, J., Black, M.J.: Efficient sparse-to-dense optical flow estimation using a learned basis and layers. In: CVPR (2015)
41. Xu, L., Jia, J., Matsushita, Y.: Motion detail preserving optical flow estimation. TPAMI 34(9), 1744–1757 (2012)
42. Yang, J., Li, H.: Dense, accurate optical flow estimation with piecewise parametric model. In: CVPR (2015)
43. Yang, J., Ye, X., Li, K., Hou, C., Wang, Y.: Color-guided depth recovery from RGB-D data using an adaptive autoregressive model. IEEE Trans. on Image Processing 23(8), 3443–3458 (2014)
44. Zhang, Q., Shen, X., Xu, L., Jia, J.: Rolling guidance filter. In: ECCV (2014)