Short Papers

Fast Cost-Volume Filtering for Visual Correspondence and Beyond

Asmaa Hosni, *Member*, *IEEE*, Christoph Rhemann, Michael Bleyer, *Member*, *IEEE*, Carsten Rother, *Member*, *IEEE*, and Margrit Gelautz, *Senior Member*, *IEEE*

Abstract—Many computer vision tasks can be formulated as labeling problems. The desired solution is often a spatially smooth labeling where label transitions are aligned with color edges of the input image. We show that such solutions can be efficiently achieved by smoothing the label costs with a very fast edge-preserving filter. In this paper, we propose a generic and simple framework comprising three steps: 1) constructing a cost volume, 2) fast cost volume filtering, and 3) Winner-Takes-All label selection. Our main contribution is to show that with such a simple framework state-of-the-art results can be achieved for several computer vision applications. In particular, we achieve 1) disparity maps in real time whose quality exceeds those of all other fast (local) approaches on the Middlebury stereo benchmark, and 2) optical flow fields which contain very fine structures as well as large displacements. To demonstrate robustness, the few parameters of our framework are set to nearly identical values for both applications. Also, competitive results for interactive image segmentation are presented. With this work, we hope to inspire other researchers to leverage this framework to other application areas.

Index Terms—Stereo matching, optical flow, interactive image segmentation

1 Introduction

Discrete label-based approaches have been successfully applied to many computer vision problems such as stereo, optical flow, interactive image segmentation, or object recognition. In a typical labeling approach, the input data are used to construct a 3D cost volume which store the costs for choosing a label l (e.g., disparities in stereo) at image coordinates x and y. For stereo, these costs are given by pixelwise matching (e.g., absolute differences of the intensities) between corresponding pixels.

Then, the goal is to find a solution which 1) obeys the label costs, 2) is spatially smooth, and 3) where label changes are aligned with edges in the image. To this end, a popular approach is to utilize a Conditional (Markov) Random Field model (CRF). This means that an energy function is formulated where the label costs are encoded in a data term and the spatially smooth edge-aligned solution is enforced by, e.g., a pairwise smoothness term. This cost function can then be minimized using global energy minimization approaches such as graph cut or belief propagation. A drawback is that such global methods are often relatively slow and do not scale well to high-resolution images or large label spaces. Fast approximations (e.g., [1]) usually come at the price of loss in quality due to less-global optimization schema.

Continuous counterparts to discrete labeling methods are based on convex energy functions which can be efficiently optimized on the GPU, e.g., [2], [3], [4]. A drawback is that many of these approaches have a restricted form of the data and smoothness term. For instance, the brightness constancy assumption in optical flow is usually linearized and thus only valid for small displacements. To overcome this problem, a coarse-to-fine framework is commonly used which, however, cannot handle objects whose scale is much smaller than their motion. Another problem is posed by the convexity of the smoothness term, which might oversmooth the solution. This may be the reason why convex models have not reported state-of-the-art stereo results yet.

An interesting alternative to an energy-based approach is to apply local filtering techniques. The filtering operation achieves a form of spatially local smoothing of the label space, in contrast to a potential spatially global smoothing of a CRF. Despite this conceptual drawback, an observation of our work and previous work [6] is that "local smoothing" is able to achieve high-quality results. We believe that one of the reasons may be the dominance of the (pixelwise) data term with respect to the smoothness term.[1] An important observation is that the data term will play an even more dominant role in the future since both video and still-picture cameras are consistently growing in terms of frame resolution and also dynamic range. A detailed comparison between energy and filtering-based methods is beyond the scope of this paper, and we will only briefly discuss them in Section 6.

In general, relatively little work has been done in the domain of filter-based methods for discrete labeling problems [6], [7], [8]. Most importantly, there is no filter-based approach for *general multi-labeling* problems which is both *fast (real time)* and achieves *high-quality* results. The key contribution of this paper is to present such a framework. This is possible due to the recently proposed *guided filter* [5], which has an edge-preserving property and a runtime independent of the filter size. Thus, state-of-the-art results can be achieved without the need to trade off accuracy against efficiency.

Let us now detail our method from a stereo perspective. We first construct a cost volume with axes (x, y, l), which is known as disparity space image (DSI) in stereo [9]. Fig. 1 (left-b) shows an (x, l) slice through this volume for the scanline in Fig. 1 (left-a). We can obtain a solution to the labeling problem by choosing the label of the lowest cost at each pixel (i.e., $\arg\min$ over the columns of Fig. 1 (left-b)). The pixels with the lowest costs are marked red in Fig. 1 (left-b). The resulting disparity map (Fig. 1 (right-b)) is very noisy because the solution is not regularized. However, this method is very fast.

To regularize the solution, we can aggregate (smooth) the costs within a support window before applying the Winner-Takes-All label selection. This is known as window-based matching in the stereo literature. It is known that this aggregation step is equivalent to filtering the (x, y) dimensions of the cost volume with a box filter [9]. The result of filtering the cost volume in Fig. 1 (left-b) using a box filter is shown in Fig. 1 (left-c). The disparity solution of minimum costs (marked red in Fig. 1 (left-c)) is smooth but not aligned with image edges. This is because the box filter overlaps depth discontinuities that are illustrated with green dashed lines in Fig. 1. This leads to the well-known "edge-fattening effect" in stereo (see the disparity map in Fig. 1 (right-c)). While the quality of the disparity map is poor, the advantage is

- *A. Hosni, C. Rhemann, M. Bleyer, and M. Gelautz are with the Institute of Software Technology and Interactive Systems, Vienna University of Technology, Interactive Media Systems Group, Favoritenstrasse 9-11/188/2, A-1040 Vienna, Austria. E-mail: {asmaa, rhemann, bleyer, gelautz}@ims.tuwien.ac.at.*
- *C. Rother is with Microsoft Research Cambridge, Cambridge, United Kingdom. E-mail: carrot@microsoft.com.*

Manuscript received 2 Dec. 2011; revised 21 June 2012; accepted 17 July 2011; published online 23 July 2011.
Recommended for acceptance by N. Paragios.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2011-12-0861.
Digital Object Identifier no. 10.1109/TPAMI.2012.156.

1. It may be possible to show that for some applications the smoothness term of a learned energy propagates information only in a small neighborhood. For some applications, global constraints exist such as the occlusion constraint in stereo matching and optical flow that we model with a fast additional operation.

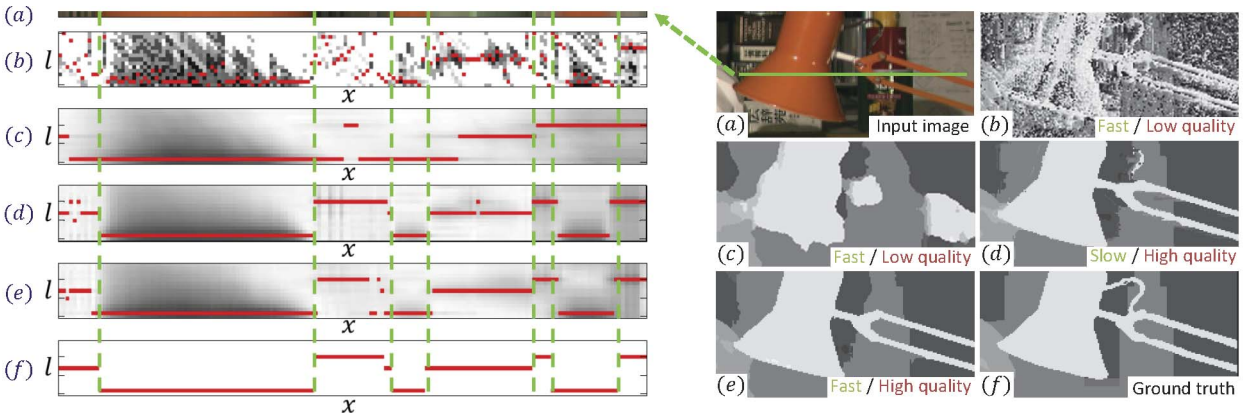

Fig. 1. **Our approach applied on the stereo matching problem.** Left: (a) Zoom of the green line in the input image. (b) Slice of the cost volume (white/black/red: high/low/lowest costs) for the line in (a). (c)-(e) Cost slice smoothed along the x and y-axes (y is not shown here) using (c) the box filter, (d) the joint bilateral filter, and (e) the guided filter [5], respectively. (f) Ground-truth labeling. Right: Crops of the "Tsukuba" disparity maps computed from the filtered costs on the left side. In contrast to other methods, our approach in (e) produces high-quality disparity maps *and* is computationally efficient (real time).

that the filtering process is very fast because it can be speeded up via a sliding window technique.

To overcome the "edge-fattening problem," we can smooth the cost volume with a *weighted* box filter where weights are chosen such that the filter preserves edges of the input color image. For instance, one can apply a joint bilateral filter. Here, weights are computed from the color image and weighted averaging is performed on the cost values. In this case, the approach becomes very similar to the one proposed in the original adaptive support weight paper [6].[2] Fig. 1d shows that applying the joint bilateral filter on the cost volume leads to a spatially smooth disparity solution that is also aligned with image edges. While this method leads to high-quality results, its computational speed represents a problem. The runtime of an exact implementation of the joint bilateral filter depends on the size of the support window (i.e., is slow for applications like stereo matching that require a large support window) and fast approximations degrade the quality considerably (see Section 2).

In our work, we propose an algorithm that produces high-quality results at real-time frame rates and hence combines the advantages of all filtering approaches outlined above. We use the recently proposed guided filter [5] to smooth the cost volume in a way that color edges are preserved (see Fig. 1e). In our experiments, this filtering technique can even outperform the joint bilateral filter in terms of quality of results. Even more importantly, it can be implemented as a sequence of box filters so that runtime is independent of the filter window size. Moving away from the stereo perspective, we show that the concept of filtering the cost volume leads to a generic and fast framework that is widely applicable to other computer vision problems. In particular, we instantiate our framework for three applications:

- a real-time stereo approach that outperforms all other local methods on the Middlebury benchmark both in terms of speed and accuracy,
- a discrete optical flow approach that handles both fine (small-scale) motion structure and large displacements. We tackle the huge label space by fast cost filtering,
- a fast and high-quality interactive image segmentation method.

2 Related Work

There have only been a few attempts to simulate the edge-preserving smoothness properties of a CRF by filtering the label costs. We review those now in the context of three application areas related to this work (i.e., stereo, optical flow, and image segmentation).

Stereo. Yoon and Kweon [6] showed that an edge-preserving bilateral filter on the cost volume can achieve high-accuracy results. Note that the authors of [6] did not use the term "filtering" to describe their method, but called it weighted support window aggregation scheme. This means that they use a naive implementation of the bilateral filter, which is slow and diminishes the runtime advantage of local over global methods. Richardt et al. [7] realized this shortcoming and suggested an approximate but fast (real-time) implementation of the filter. Due to this approximation, it cannot be easily applied to color images due to high memory requirements. As a consequence, this approach is limited to grayscale input images, giving poor results at disparity boundaries. This is reflected in the Middlebury stereo benchmark [10], where the method of Richardt et al. [7] is on the 90th rank out of over 110 methods. In contrast, our real-time implementation ranks 16th. To increase the quality, Richardt et al. [7] proposed an alternative that uses two color channels. However, this method is 13 times slower than their monochromatic approach (no real-time performance) and still inferior to their reimplementation of [6]. It is also important to note that the approach of Richardt et al. [7] is tailored to stereo matching and hence does not convey the important insight that this filtering technique can be leveraged to general labeling tasks, outside stereo matching.

After publication of the conference version of this paper [11], De-Maeztu et al. [12] proposed a stereo approach that is based on a cost aggregation strategy very similar to ours. They presented a symmetric stereo approach that aggregates costs according to both input images simultaneously. It is worth noting that we have also presented a symmetric formulation in [11] that is, in fact, identical to the symmetric stereo method in [12]. However, in general our approach aggregates costs based on a single image. This enables a generalization of our strategy to other vision problems such as interactive image segmentation where only a single input image is available, whereas [12] only works for stereo matching.

Optical flow. Many optical flow methods rely on continuous optimization strategies. In the continuous domain, Tschumperlè and Deriche [13] showed that a diffusion tensor-based smoothness regularizer can be transformed into local convolutions with oriented Gaussians. In the optical flow approach of Xiao et al. [14], the Gaussian kernels were replaced by the bilateral filter. In this line of research, Werlberger et al. [15] and Sun et al. [16] incorporated the adaptive support weights of [6] into a variational

2. The approach in [6] is motivated from a different perspective and does not show its relations to filtering explicitly.

approach. In contrast, the goal of our work is to apply local filtering in a discrete framework.

All of the aforementioned approaches are based on the popular variational coarse-to-fine framework that cannot handle large displacements of small objects, as shown in [17], [18]. To overcome this problem, a discretized data term can be integrated into a variational framework as a soft constraint (see [17], [18], [19]).

Purely discrete label-based approaches do not suffer from this problem, but a major challenge is the huge label space (each flow vector is a label and subpixel accuracy further increases the label space). Due to these difficulties, discrete approaches [20] could not report state-of-the-art performance for a long time and usually have to trade off search space (quality) against speed. In contrast, our filter-based method efficiently deals with the search space and handles both large displacements and fine small-scale structures.

Interactive image segmentation. Interactive image segmentation is a binary labeling problem. It aims to separate the image into foreground and background regions given some hints by the user. Criminisi et al. [8], [21] showed an approach which is applicable to problems with two labels, like binary image segmentation and panoramic stitching of two overlapping images. The idea is to filter a likelihood ratio mask with a fast geodesic morphological operator. It remains unclear whether this approach can be extended to multilabel problems such as stereo. Also, the relationship between their morphological operator and, e.g., the bilateral filter is not discussed in detail.

Related to interactive segmentation, He et al. [5] adopted the guided filter to compute a soft-segmentation, i.e., a so-called alpha matte [22]. This is done by filtering a binary segmentation mask, as opposed to the cost volume, as in our approach. We close the loop by estimating the binary segmentation via filtering the cost volume constructed from coarse user input. This results in a purely filter-based segmentation and matting pipeline.

3 Cost-Volume Filtering

We now describe our labeling framework. Let us consider a general labeling problem where the goal is to assign each pixel i with coordinates (x, y) in the image I a label l from the set $\mathcal{L} = \{1, \dots, L\}$. The label assigned to pixel i is denoted by f_i and f is the collection of all label assignments. Our approach consists of three steps: 1) constructing the cost volume, 2) filtering the cost volume, and 3) label selection. The cost volume C is a 3D array which stores the costs for choosing label l at pixel $i = (x, y)$.

The L slices of the cost volume are now filtered. To be more precise, the output of the filtering at pixel index i with label l is a weighted average of neighboring pixels in the same slice:

$$C'_{i,l} = \sum_j W_{i,j}(I) C_{j,l}. \tag{1}$$

Here, C' is the filtered cost volume and i and j are pixel indices. The filter weights $W_{i,j}$ depend on the guidance image I, which is, in the case of, e.g., stereo, the reference image.

Once the cost volume is filtered, the label at pixel i is simply chosen in a Winner-Takes-All manner as

$$f_i = \arg\min_{l \in \mathcal{L}} C'_{i,l}. \tag{2}$$

The filter weights $W_{i,j}$ in (1) should be chosen such that intensity changes in the guidance image are maintained in the filter output. In this work, we use the weights of the guided filter [5], but other weights are also possible.

Given a color guidance image I, the weights $W_{i,j}$ of the guided filter are given by

$$W_{i,j} = \frac{1}{|\omega|^2} \sum_{k:(i,j) \in \omega_k} \left(1 + (I_i - \mu_k)^T (\Sigma_k + \epsilon U)^{-1} (I_j - \mu_k)\right), \tag{3}$$

where μ_k and Σ_k are the mean vector and covariance of I in a squared window ω_k with dimensions $2r + 1 \times 2r + 1$, centered at pixel k. U denotes the identity matrix. We denote the number of pixels in this window with $|\omega|$ and ϵ is a smoothness parameter. We refer the reader to He et al. [5] for further details.

It has been shown [5] that a weighted average with weights as defined in (3) can be implemented efficiently on the CPU as a sequence of box filters using the integral imaging technique [23]. We apply the same technique to obtain an even more efficient GPU implementation.

4 Applications

We now show three applications for our cost filtering framework. The method for stereo and optical flow is almost identical and only one parameter is set differently (see the explanation in Section 5).

4.1 Stereo Matching and Optical Flow

For stereo matching and optical flow, the labels l correspond to vectors (u, v) which define the displacement in x and y directions. (For stereo, the displacement in the x direction corresponds to the disparity d ($u = d$) and there is no shift in the y direction ($v = 0$).)

Cost computation. The cost volume expresses how well a pixel i in image I matches the same pixel in the second image I' shifted by vector l. We choose our pixel-based matching costs to be a truncated absolute difference of the color and the gradient at the matching points. Such a model has been shown to be robust to illumination changes and is commonly used in optical flow estimation [17], [24]:

$$\begin{aligned} C_{i,l} = (1 - \alpha) \cdot \min\left[\|I_i - I'_{i-l}\|, \tau_1\right] \\ + \alpha \cdot \min\left[\|\nabla_x I_i - \nabla_x I'_{i-l}\| + \|\nabla_y I_i - \nabla_y I'_{i-l}\|, \tau_2\right]. \end{aligned} \tag{4}$$

Here, ∇_x and ∇_y are the grayscale gradients in x and y direction, respectively.[3] α balances the color and gradient terms and τ_1, τ_2 are truncation values.

We then filter the cost volume according to (1) with weights in (3) using I as guidance image. Finally, we compute the disparity/flow map f for image I as per (2).

Occlusion detection. To detect pixels with unreliable disparities/flow vectors which are mainly caused by occlusions, we apply a left-right cross checking procedure. To this end, we additionally compute the disparity/flow map f' for the right image I' in the same way as described above. We mark a pixel in the left disparity/flow map f as invalid if the value of its matching pixel differs, i.e., if $f_i \neq f'_{i-l}$. This process detects occluded as well as mismatched pixels.

Postprocessing for stereo. First, invalid pixels are assigned to a new disparity. To this end, we simply assign an invalid pixel to the lowest disparity value of the spatially closest nonoccluded pixel located on the same scanline. This simple strategy for filling invalidated pixels generates streak-like artifacts in the disparity map. To remove these artifacts while preserving the object boundaries, we apply a weighted median filter for the filled-in pixels. Note that the weighted median is applied *only* to invalid pixels, i.e., pixels which fail the left-right cross checking. As weights, we use those of the bilateral filter:

$$W^{bf}_{i,j} = \frac{1}{K_i} \exp\left(-\frac{|i - j|^2}{\sigma_s^2}\right) \exp\left(-\frac{|I_i - I_j|^2}{\sigma_c^2}\right), \tag{5}$$

where σ_s and σ_c adjust the spatial and color similarity and K_i is a normalization factor. We truncate the weights at a radius of 7 pixels. We discuss the influence of the postprocessing in Section 5.1.

3. For stereo matching, we use the gradient in the x-direction only.

Fig. 2. **Results on the Middlebury evaluation images.** Disparity maps from our algorithm using constant parameter settings are shown. In the left upper corners, we show comparisons with the respective ground truth (errors larger than one pixel in nonoccluded regions are black, while gray denotes errors in occluded areas).

Postprocessing for optical flow. As for stereo, we start by assigning invalid pixels to a new flow vector. For optical flow, we cannot assign the flow vector with the lowest magnitude of the spatially closest pixels because objects with smaller flow magnitude can occlude objects with higher flow magnitude. Therefore, we use a weighted median filter to fill the occluded regions based on their color similarity to the visible flow regions. In detail, we apply a weighted median filtering with weights as in (5) to the occluded pixels. The windows of the median filter overlap the valid regions. Hence, we can propagate the flow vectors into the invalid image parts. If the invalidated region is larger than the size of the filter window, some pixels will not be assigned to any flow vector. In such situations, we iterate the median filtering procedure to incrementally fill invalid pixels.

To find subpixel accurate flow vectors, we upscale the cost volume in label dimension and derive the image colors at subpixel positions via bicubic interpolation. In practice, we found that smoothing the final flow vectors with the guided filter can compensate for a lower upscaling factor We empirically found that an upscaling factor of 4 gives visually pleasing results. However, in this paper we apply an upscaling factor of 8 to demonstrate the best possible performance.

Alternative—symmetric cost aggregation. The cost aggregation in our stereo/flow approach can also be formulated symmetrically for both input images. This means that we filter the cost volume while preserving the edges in *both* input images simultaneously. In particular, we compute the filter weights for pixel i in (3) based on the color of the pixel in the left image I_i and its matching pixel in the right image I'_{i-l} that can be computed using disparity/flow hypothesis l. To this end, we replace the 3×1 vector I_i in (3) with a 6×1 vector whose entries are given by the RGB color channels of both I_i and I'_{i-l}. The dimensions of I_j, Σ_k, U, and μ_k change accordingly.

We found that for stereo the symmetric approach gives visible improvements near occlusion boundaries. For the symmetric optical flow approach, we found a slight overall performance improvement over the asymmetric flow approach. We compare asymmetric with symmetric approaches in Sections 5.1 and 5.2.

4.2 Interactive Image Segmentation

In interactive image segmentation, the labels encode whether a pixel belongs to the foreground F or background B; thus, $\mathcal{L} = \{F, B\}$. For initialization, the user assigns parts of the image to fore- and background.

Cost computation. From the user assignments, we build fore and background color histograms denoted as θ^F and θ^B, which sum up to 1. Each histogram has K bins and we denote the bin into which pixel i falls with $b(i)$. We can also use a bounding box as input, where the pixels outside the box build θ^B and all pixels inside the box build θ^F as in [25]. Then, the cost volume is given by $C_{i,l} = 1 - \theta^l_{b(i)}$; $l \in \mathcal{L}$. For binary labeling problems, we can reduce the cost volume $C_{i,l}$ to a 2D cost surface C_i which denotes the costs of a pixel to belong to foreground $C_i = 1 - \theta^F_{b(i)}/(\theta^F_{b(i)} + \theta^B_{b(i)})$. If a pixel i has been assigned to the fore or background by the user, C_i is set to 0 or 1, respectively. After filtering the cost surface, a pixel i is assigned to the foreground if $C_i < 0.5$ and assigned to the background otherwise. When using bounding boxes, we iteratively update the color models as in [26]. In practice, we achieved good results using five iterations.

To account for semitransparent pixels along the object boundary in an efficient manner, we filter the computed binary mask with the guided filter. This has been shown [5] to approximate a matting method.

5 Experimental Results

We now demonstrate that our method is capable to generate state-of-the-art results for stereo matching, optical flow and interactive image segmentation. We use the following, the *same* constant parameter settings for stereo and optical flow to generate all of our results: $\{\mathrm{r}, \epsilon, \alpha, \sigma_s, \sigma_c, \tau_1\} = \{9, 0.01^2, 0.9, 9, 0.1, 0.028\}$. This demonstrates the robustness of our method. The only exception is the truncation value τ_2 of the matching costs in (4). This value depends on the signal-to-noise ratio of an image [9] as well as on the size of the occluded regions. Thus, we use $\tau_2 = 0.008$ for stereo and $\tau_2 = 0.016$ for optical flow.

Interactive image segmentation is a very different problem; hence, we found different, constant parameter settings (i.e., more smoothing) work well: $\{r, \epsilon\} = \{11, 0.2^2\}$. Note that for interactive segmentation the only further parameter we use is $K = 32$, which defines the number of bins of the color histogram.

We implemented our method on the GPU using CUDA. All experiments were conducted on an Intel Core 2 Quad, 2.4 GHZ processor and a GeForce GTX480 graphics card. Our approach takes about 2.85 ms to filter a 1 Mpix image. Thus, we can process about 351 labels per second in a 1 Mpix image. Problem-specific timings are reported below. A (slower) Matlab implementation of our stereo method is available at www.ims.tuwien.ac.at/research/costFilter.

5.1 Stereo Matching

We now discuss results on the Middlebury stereo benchmark [10] which provides 35 image pairs with known ground truth. A subset of these images (four images) is used to compare over 110 stereo matching methods in the Middlebury online evaluation table. The disparity maps generated by our approach on these four test

TABLE 1
Rankings for Selected Local Stereo Methods
(We Are the Best Performing Method)

Algorithm	Rank	Avg. Error (%)
CostFilter (sym.)	**12**	**5.35**
CostFilter	**16**	**5.55**
GeoSup [27]	21	5.80
CostFilter (w/o post-processing)	23	5.77
CostFilter (Y & K Weights [6])	24	5.86
AdaptWeight [6]	48	6.67
DCBGrid [7]	90	10.9

Fig. 3. **Effect of postprocessing.** Postprocessing was applied only in invalidated regions. From left to right: Disparity map with invalidated pixels in red; disparity after scanline-based filling; disparity after applying weighted median filtering for invalid pixels.

TABLE 2
Evaluation for Selected Stereo Methods on All 35 Middlebury Stereo Pairs

Algorithm	Rank	Avg. Error (%)
CostFilter (sym.)	**1**	**8.16**
CostFilter	**2**	**8.36**
CostFilter (w/o post-processing)	3	8.51
GeoSup [27] (our GPU impl.)	4	9.21
CostFilter (Y & K Weights [6])	5	9.28
AdaptWeight [6] (our GPU impl.)	6	15.69
DCBGrid [7]	7	16.73

Second column: Rank of each method according to its average error.

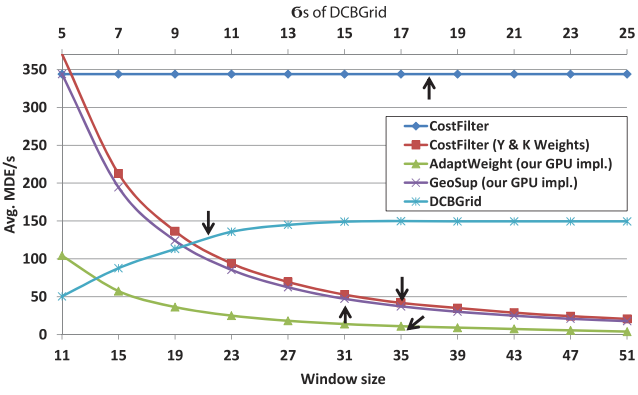

Fig. 4. **Efficiency comparison.** A larger MDE/s score is better. Arrows indicate optimal window sizes. Our algorithm is independent of the window size.

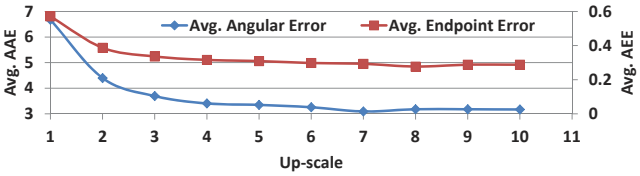

Fig. 6. **Effect of subsampling factor on the error rates.** We show the error on the Middlebury training images.

images are shown in Fig. 2. We can see that our method generates high-quality disparity maps and preserves the boundaries of thin objects. We report quantitative results that we have taken from the Middlebury online evaluation system for the four test images in Table 1. The table plots Middlebury ranks and average percentages of bad pixels (rightmost column of the Middlebury table) using the Middlebury default error threshold of 1. Our approach (CostFilter) ranks 16th out of more than 110 methods.

We also tested our alternative symmetric stereo approach described in Section 4.1 (see entry "CostFilter (sym.)" in Table 1). We found that the average error for the four test images is reduced from 5.55 to 5.35 percent leading to an improved rank at position 12. The entry "CostFilter(w/o postprocessing)" in Table 1 shows the error rate of our method without postprocessing (i.e., without filtering the invalid regions by a weighted median filter). We see that the average error slightly increases, which results in a slightly worse overall rank 23. Fig. 3 shows the effect of our postprocessing on the teddy image pair.

An important observation from the Middlebury online table is that, to our knowledge, our stereo algorithm is the top performer among all *local* stereo methods which are listed in the ranking. In particular, as can be seen in Table 1, it can outperform the geodesic approach (GeoSup), the original adaptive support weight algorithm of [6] (AdaptWeight), and a fast approximation of the latter technique (DCBGrid). To understand why our method performs better than [6], we plugged their support weights into our algorithm. Hence, we use the same matching costs, occlusion handling, and postprocessing methods as in our algorithm. We tuned the parameters of the resulting algorithm to optimize the Middlebury ranking. This approach (CostFilter (Y & K Weights [6])) ranks behind our guided filter-based algorithm, i.e., on rank

Fig. 5. **Outdoor results.** We show the left view of two stereo pairs captured with a consumer stereo camera and our disparity maps without occlusion filling and postprocessing (invalidated pixels are black).

TABLE 3
Optical Flow Evaluation on Middlebury

Method	Angular Error				Endpoint Error				Time
	Rank	Schefflera	Grove	Teddy	Rank	Schefflera	Grove	Teddy	(sec)
Layers++	2	(1,1,12)	(1,1,1)	(2,1,6)	2	(1,1,8)	(1,1,2)	(1,1,7)	18206
Classic+NL [16]	5	(9,7,16)	(5,3,4)	(4,4,11)	5	(10,10,13)	(3,3,5)	(3,2,12)	972
MDP-Flow [19]	8	(5,5,22)	(9,8,15)	(27,29,30)	7	(4,5,17)	(9,9,15)	(29,32,28)	188
CostFilter	**10**	**(2,2,4)**	**(2,2,3)**	**(1,2,1)**	**13**	**(2,2,8)**	**(2,2,1)**	**(1,4,3)**	**55.31**
CostFilter (sym.)	**11**	**(4,3,2)**	**(2,3,1)**	**(1,4,1)**	**12**	**(4,4,4)**	**(3,3,1)**	**(1,4,1)**	**110.65**
OFH	12	(23,25,5)	(11,11,21)	(12,17,8)	8	(20,25,4)	(18,17,21)	(12,17,12)	620
NL-TV-NCC [15]	13	(16,16,2)	(26,31,11)	(11,13,10)	10	(16,16,2)	(14,15,9)	(10,11,2)	20
DPOF [29]	17	(4,4,13)	(13,15,20)	(9,6,5)	16	(4,4,17)	(5,5,3)	(7,4,1)	287
ACK-Prior [30]	18	(6,9,3)	(19,13,27)	(19,12,14)	20	(6,6,2)	(15,14,18)	(27,18,23)	5872

Overall our method ranks 10th and 13th with respect to the angular and endpoint error, respectively. It works particularly well for the challenging "Schefflera," "Grove," and "Teddy" sequences that are difficult for many competitors. We report ranks for these sequences in brackets (all, disc, untext). Runtime is given for the "Urban" sequence, which has the largest label space.

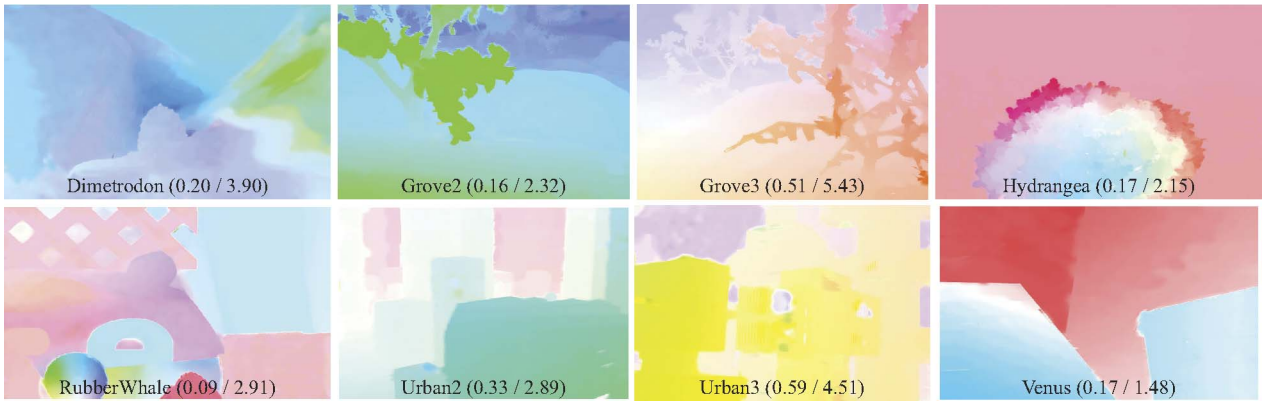

Fig. 7. **Optical flow results.** Results for the training sequences of the Middlebury benchmark data set. The respective AEE and AAE are given in parentheses (AEE/AAE). Color coding as in [10].

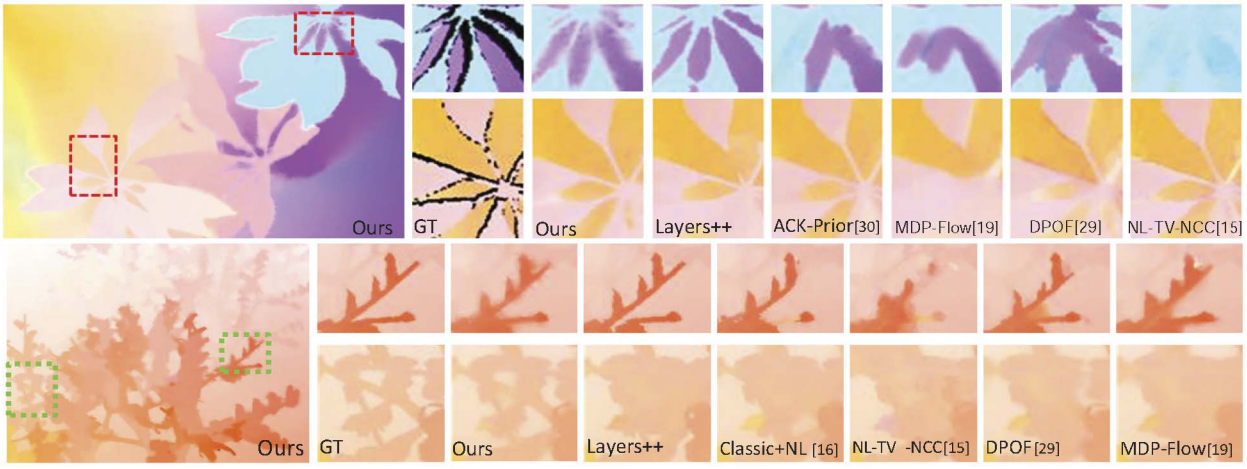

Fig. 8. **Detailed flow results.** Comparison of two sequences with thin structure, where many competitors fail to preserve flow discontinuities. (We boosted the colors in the second row from the top for better visualization.)

24 in contrast to rank 16. This suggests that the guided filter and the joint bilateral filter are both well suited for stereo matching. However, our guided filter-based approach seems to perform slightly better in terms of quality and considerably better in terms of computational efficiency, as we discuss later.

In Table 2, we compare the performance of local stereo matching methods on all 35 Middlebury ground-truth images. This test is important not only because it is a richer test set, but also because the four Middlebury evaluation images are close to being "solved," which means that a minor error in the disparity map leads to a large difference in the ranking. Since only four of the 35 test scenes are used for the online evaluation, no results are available for competitors to our algorithm. To obtain results of competing techniques, we reimplemented [6] (AdaptWeight [6](our GPU impl.)) as well as [27] (GeoSup [27] (our GPU impl.)) on the GPU and used the authors' GPU implementation of DCBGrid [7]. As a performance measure, we use the percentage of pixels that have a disparity error larger than one pixel in nonoccluded regions and build the average of this measure over all 35 test images. We plot the corresponding values in Table 2. It can be seen that the ranking of the selected methods in Table 1 is almost the same as in Table 2, i.e., our symmetrical method is the winner and slightly outperforms our asymmetrical technique. In this test, the parameters for each method were tuned to give the best results for the four images which are used in the online Middlebury evaluation. These settings were then used for all 35 test pairs, shown in Table 2, where the four evaluation images are a subset.

Let us now focus on computational efficiency of the methods in Table 2 that we have implemented on the GPU. For a fair comparison, we have run them on the same computer, with specifications given at the beginning of this section. We measure computational performance in terms of Million Disparity Estimations per second (MDE/s), as done previously in, e.g., [28]. The MDE/s measure of an algorithm for a stereo test pair is computed as: $\mathrm{MDE/s} = (W \times H \times \mathcal{D} \times \mathrm{FPS})/10^6$. Here, W and H are image width and height, $\mathcal{D}$ is the number of allowed disparities, and FPS is the number of frames per second. Hence, a larger MDE number means a better performing system.

In Fig. 4, we plot the MDE/s of various methods with respect to different sizes of the support window. As can be seen, our method is the only one with a runtime independent of the support window size.[4] The runtime of all other algorithms, except for DCBGrid (see discussion below), increases with larger window sizes, which is a disadvantage since large windows are necessary for good results. We have marked optimal window sizes reported in the corresponding papers for each method using arrows in Fig. 4. For a window size of 35×35, our algorithm "CostFilter" is more than eight times faster than our GPU implementation of the asymmetric joint bilateral filter "CostFilter (Y & K Weights)."

For DCBGrid [7], there is no explicit window size. We plot the average MDE/s for varying σ_s.[5] DCBGrid is the algorithm that is

4. Our algorithm runs at 33.3 fps (excluding the computational overhead for rendering and rectification) for 640×480 pixel images with allowed disparities ranging from 0 to 40.

5. σ_s in DCBGrid [7] is the spatial sampling rate. It can be considered as equivalent to the window size in naive aggregation methods because it controls the amount of smoothing. The number of grid cells is inversely proportional to the sampling rate: A larger σ_s yields a smaller number of grid cells and requires less memory. Thus, the average MDE/s for this method increases with larger σ_s.

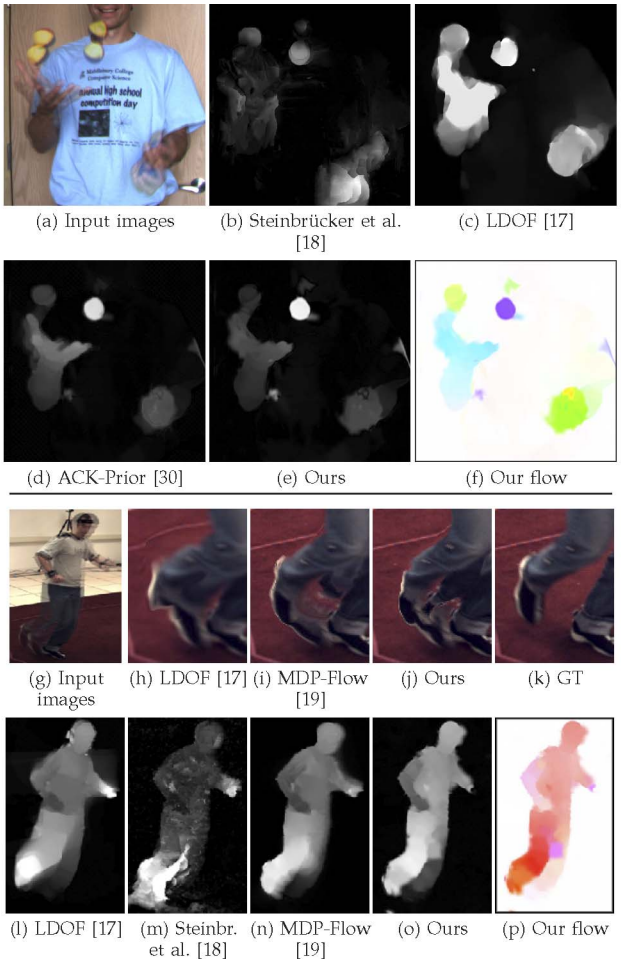

Fig. 9. **Large displacement flow.** (b)-(e) and (l)-(o) Motion magnitude for different methods. (f), (p) Our flow vectors with the color coding as in [10]. (h)-(k) Backward warping results using flow of different methods—the tip of the foot is correctly recovered by our method. Occluded regions cannot be handled by any method.

closest to ours in terms of runtime. However, it is considerably inferior in terms of matching quality (see Tables 1 and 2) and results get worse with larger σ_s, in which case the algorithm would be attractive in terms of runtime.

We also show results of our algorithm for two complex outdoor scenes captured with a consumer stereo camera (Fujifilm FinePix Real3D W1). Input images and corresponding disparity maps generated by our method are shown in Fig. 5. Our method seems to work very well, although the input images contain many challenges, including untextured regions (e.g., walls), thin structures (e.g., bicycles), or different illumination conditions.

5.2 Optical Flow

We evaluate our approach on the Middlebury flow benchmark [10]. The benchmark comprises an evaluation data set of eight images with hidden ground-truth flow as well as eight training scenes with publically available ground-truth flow. Table 3 compares our method on the evaluation data set with selected approaches from the Middlebury table that compete with our algorithm in terms of method and quality. To be more precise, we show the rank of our method computed over all eight evaluation data sets, as well as detailed results for three challenging data sets "Schefflera," "Grove," and "Teddy." Overall, our approach is on the 10th and 13th rank, with respect to the Average Angular Error (AAE) and Average Endpoint Error (AEE), out of almost 60 methods. This performance is comparable to the method of Werlberger et al. (NL-TV-NCC) [15] that uses adaptive support weights in a variational framework. Our method has several advantages over its competitors, including [15]. First, our approach outperforms most other methods on scenes with thin structures and strong motion discontinuities, such as in "Schefflera," "Grove," and "Teddy," where we achieve an average rank of 2.2 (not shown in the table) and rank 1 on the "Teddy" scene (see details in Table 3 and Fig. 8). Second, our approach, which uses fixed parameter settings, can handle scenes with large displacements, which is difficult for approaches such as [15] that are restricted by their coarse-to-fine framework. Finally, our method is simpler than many other approaches that often require a large number of parameters to be tuned, e.g., pyramid levels and interpolation strategy.

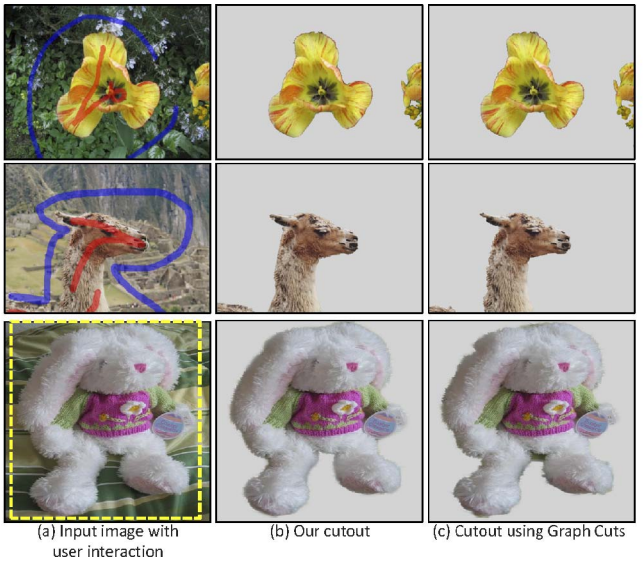

Fig. 10. **Segmentation results.** (b) Binary segmentation from (a). (c) Result of GrabCut [26]. The "Bunny" segmentations (last row) were postprocessed with the guided filter to obtain an alpha matte.

Our approach performs less well on the "Wooden" and "Yosemite" sequence. This is because in the "Wooden" sequence our algorithm assigns wrong flow values to a shadowed region. The artificial "Yosemite" sequence contains many untextured regions where the data term is unreliable. Variational methods smoothly interpolate over these regions while our method misinterprets them as motion discontinuities. We observed that this is less of a problem in natural high-resolution images where the data term gives useful information even in regions that appear homogeneous at a first glance.

The total runtime of our method for the "Urban" scene, which has the largest label space (640×480 pixels with 30,000 labels at a subsampling factor of 8) is approximately 55.31 seconds. In Fig. 6, we study the effect of using different subsampling factors. As can be seen from this plot, smaller subsampling factors give results of comparable quality at considerably lower runtimes. For example, an upscale factor of four has a runtime of 13.52 seconds.

For completeness, we show the performance of our optical flow algorithm for the Middlebury training images in Fig. 7. The reader is referred to the supplementary material, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2012.156, for the complete result.

Large displacement flow results. An important advantage of our method is that it can also handle large displacements without the need of changing any parameters (see Fig. 9 for a comparison to other methods). Our approach generates results that are visually comparable to, or better than, methods which are specifically tailored on large displacement flow and are not top performing for small displacements.

Symmetric approach. We also tested a symmetric optical flow formulation of our approach, see Table 3. The symmetric optical flow approach does not seem to have a big effect on the results. For the AAE measure, it degrades performance by one rank, while on the AEE error it improved by one rank. We also tested the symmetric optical flow approach on the training sequences from Middlebury. We found that the overall improvement is negligible. The AEE is reduced from 0.28 to 0.27 and the AAE is reduced from 3.2 to 3.1.

5.3 Interactive Image Segmentation

To show that our approach also performs well for image segmentation, we visually compare our results to those of GrabCut [26], Fig. 10. As user input, we either use coarse scribbles/trimaps or a single bounding box. The results are visually comparable at lower runtimes (2.85 ms versus approximately 300 ms using the graph cut implementation of [21] and 425 ms using the graph cut implementation of [31] on a 1 Mpixel image). Furthermore, our method gives comparable results to GrabCut [26] on a ground-truth database of 50 images [26]. Here, error is measured as the percentage of misclassified pixels in the area not marked by the user. Given the trimap input of [26], the error is 5.3 percent for GrabCut and 6.2 percent for our method. This shows the potential of our approach to be successfully applied to other vision applications. A video of our real-time segmentation tool is shown in the supplementary material, available online.

6 Discussion and Future Work

This paper presented a simple, yet powerful filtering approach for solving discrete labeling problems. As mentioned in the introduction, the relationships between filtering-based operations and energy-based optimization schema, for both continuous and discrete label spaces, are to the best of our knowledge not fully understood. One relationship, given in [5], is that the guided filter is one step of a conjugate gradient solver of a particular linear system. We believe that a better understanding of this relationship can lead to fast and even better (iterative) filtering approaches.

Finally, we note that all the aforementioned stereo algorithms, as well as ours, assume that pixels within a support window have a constant disparity value. This assumption is violated for slanted surfaces, where the support window is comprised of pixels that lie on different disparities. Recently, Bleyer et al. [32] proposed to overcome this problem by estimating a 3D plane at each pixel onto which the support region is projected. An interesting future research direction is to accelerate this technique by levering the insights of our proposed approach.

Acknowledgments

Asmaa Hosni was supported by the Vienna PhD School of Informatics. Christoph Rhemann and Michael Bleyer have been funded by the Vienna Science and Technology Fund (WWTF) through project ICT08-019.

References

[1] O. Veksler, "Stereo Correspondence by Dynamic Programming on a Tree," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2005.
[2] T. Pock, T. Schoenemann, G. Graber, H. Bischof, and D. Cremers, "A Convex Formulation of Continuous Multi-Label Problems," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.
[3] M. Klodt, T. Schoenemann, K. Kolev, M. Schikora, and D. Cremers, "An Experimental Comparison of Discrete and Continuous Shape Optimization Methods," *Proc. 10th European Conf. Computer Vision,* 2008.
[4] P. Gwosdek, H. Zimmer, S. Grewenig, A. Bruhn, and J. Weickert, "A Highly Efficient GPU Implementation for Variational Optic Flow Based on the Euler-Lagrange Framework," *Proc. ECCV Workshop Computer Vision with GPU,* 2010.
[5] K. He, J. Sun, and X. Tang, "Guided Image Filtering," *Proc. European Conf. Computer Vision,* 2010.
[6] K. Yoon and S. Kweon, "Adaptive Support-Weight Approach for Correspondence Search," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 4, pp. 650-656, Apr. 2006.
[7] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. Dodgson, "Real-Time Spatiotemporal Stereo Matching Using the Dual-Cross-Bilateral Grid," *Proc. European Conf. Computer Vision,* 2010.
[8] A. Criminisi, T. Sharp, and C. Rother, "Geodesic Image and Video Editing," *ACM Trans. Graphics,* vol. 29, pp. 1-15, 2010.
[9] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *Int'l J. Computer Vision,* vol. 47, pp. 7-42, 2002.
[10] http://vision.middlebury.edu, 2012.
[11] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, "Fast Cost-Volume Filtering for Visual Correspondence and beyond," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2011.
[12] L. De-Maeztu, S. Mattoccia, A. Villanueva, and R. Cabeza, "Linear Stereo Matching," *Proc. IEEE Int'l Conf. Computer Vision,* 2011.
[13] D. Tschumperlè and R. Deriche, "Vector-Valued Image Regularization with PDE's: A Common Framework for Different Applications," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.
[14] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi, "Bilateral Filtering-Based Optical Flow Estimation with Occlusion Detection," *Proc. European Conf. Computer Vision,* 2006.
[15] M. Werlberger, T. Pock, and H. Bischof, "Motion Estimation with Non-Local Total Variation Regularization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2010.
[16] D. Sun, S. Roth, and M. Black, "Secrets of Optical Flow Estimation and Their Principles," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2010.
[17] T. Brox and J. Malik, "Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 33, no. 3, pp. 500-513, Mar. 2011.
[18] F. Steinbrücker, T. Pock, and D. Cremers, "Large Displacement Optical Flow Computation without Warping," *Proc. IEEE 12th Int'l Conf. Computer Vision,* 2009.
[19] L. Xu, J. Jia, and Y. Matsushita, "Motion Detail Preserving Optical Flow Estimation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2010.
[20] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 11, pp. 1222-1239, Nov. 2001.
[21] A. Criminisi, T. Sharp, and A. Blake, "GeoS: Geodesic Image Segmentation," *Proc. European Conf. Computer Vision,* 2008.
[22] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott, "A Perceptually Motivated Online Benchmark for Image Matting," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2009.
[23] F. Crow, "Summed-Area Tables for Texture Mapping," *Proc. ACM Siggraph,* 1984.
[24] A. Bruhn and J. Weickert, "Optical Flow Estimation on Coarse-to-Fine Region-Trees Using Discrete Optimization," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.
[25] S. Vicente, V. Kolmogorov, and C. Rother, "Joint Optimization of Segmentation and Appearance Models," *Proc. IEEE Int'l Conf. Computer Vision,* 2009.
[26] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut-Interactive Foreground Extraction Using Iterated Graph Cuts," *Proc. ACM Siggraph,* 2004.
[27] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann, "Local Stereo Matching Using Geodesic Support Weights," *Proc. IEEE Int'l Conf. Image Processing,* 2009.
[28] R. Yang and M. Pollefeys, "Multi-Resolution Real-Time Stereo on Commodity Graphics Hardware," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.
[29] C. Lei and Y. Yang, "Optical Flow Estimation on Coarse-to-Fine Region-Trees Using Discrete Optimization," *Proc. IEEE Int'l Conf. Computer Vision,* 2009.
[30] K. Lee, D. Kwon, I. Yun, and S. Lee, "Optical Flow Estimation with Adaptive Convolution Kernel Prior on Discrete Framework," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.
[31] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 9, pp. 1124-1137, Sept. 2004.
[32] M. Bleyer, C. Rhemann, and C. Rother, "Patchmatch Stereo—Stereo Matching with Slanted Support Windows," *Proc. British Machine Vision Conf.,* 2011.