LOCAL STEREO MATCHING USING GEODESIC SUPPORT WEIGHTS

Asmaa Hosni, Michael Bleyer, Margrit Gelautz and Christoph Rhemann

Institute for Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstr. 9-11/188/2, A-1040 Vienna, Austria, [asmaa, bleyer, gelautz, rhemann]@ims.tuwien.ac.at

ABSTRACT

Local stereo matching has recently experienced large progress by the introduction of new support aggregation schemes. These approaches estimate a pixel's support region via color segmentation. Our contribution lies in an improved method for accomplishing this segmentation. Inside a square support window, we compute the geodesic distance from all pixels to the window's center pixel. Pixels of low geodesic distance are given high support weights and therefore large influence in the matching process. In contrast to previous work, we enforce connectivity by using the geodesic distance transform. For obtaining a high support weight, a pixel must have a path to the center point along which the color does not change significantly. This connectivity property leads to improved segmentation results and consequently to improved disparity maps. The success of our geodesic approach is demonstrated on the Middlebury images. According to the Middlebury benchmark, the proposed algorithm is the top performer among local stereo methods at the current state-of-the-art.

Index Terms— Local stereo, segmentation-based stereo, adaptive support weights, geodesic distance transform

1. INTRODUCTION

Local stereo matching algorithms center a support window on a pixel of the reference image. This window is then displaced in the second view along the corresponding epipolar line in order to find a matching point of maximum correspondence.

The major challenge in local stereo is to find a well-suited size for the typically square support window. A window should thereby be large enough to capture sufficient intensity variation for handling regions of poor texture. At the same time, the window should be small enough to not include pixels of different disparities in order to avoid the well-known edge fattening effect at disparity discontinuities. In practice, there is no golden middle between these conflicting requirements. This has led to a long track of research that varies window sizes and shapes depending on the image position (e.g., [1]). However, the moderate results of these early approaches are in general not able to compete with state-of-the-art algorithms that typically rely on global optimization [2].

Local stereo matching has recently experienced a renaissance with the upcome of novel segmentation-based support aggregation schemes. These methods can deliver results close to the quality of global approaches. They assume that pixels of homogeneous color share the same disparity value. Hence, it is reasonable to find a pixel's support region via the use of color segmentation. In the original adaptive weight approach [3], weights are computed for all pixels within a square window. These weights regulate a pixel's influence in the matching process. A pixel's weight is thereby inversely proportional (i) to its color dissimilarity to the window's center pixel and (ii) to its spatial distance from the center.

Alternative methods for computing these weights have been proposed recently. The work of [4, 5] suggests using a precomputed (mean shift-based) color segmentation in the weight calculation. While this approach seems to work slightly better, it takes away some elegance by "outsourcing" the segmentation into a preprocessing step that also consumes additional processing time. In the context of prior work, also the approach of [6] is mentioned. Here, weight computation is accomplished via energy minimization. For more details and evaluation of related aggregation methods, the reader is referred to two recent studies [7, 8].

2. ALGORITHM

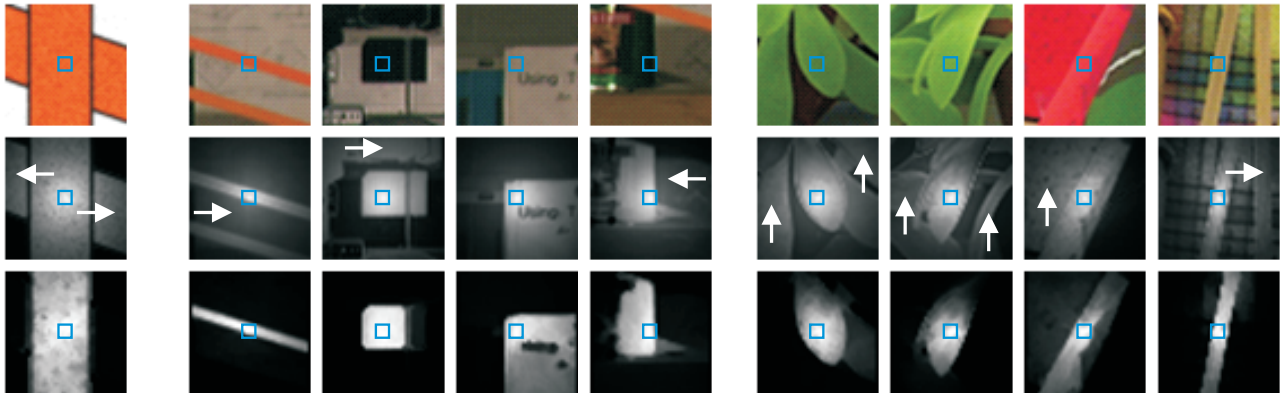
2.1. Local Matching with Geodesic Support

The basic idea of this paper is to compute the support weights within a square window via the use of geodesic distances. This is explained as follows. The geodesic distance $D(p, c)$ between a pixel p of the support window and the window's center c is defined as the shortest path that connects p with c in the color volume:

$$D(p, c) = \min_{P \in \mathcal{P}_{p,c}} d(P). \quad (1)$$

Here, $\mathcal{P}_{p,c}$ denotes the set of all paths between p and c. A path P is defined as a sequence of spatially neighbouring points in

Asmaa Hosni is supported by a One-World Scholarship of the Austrian Orient Society Hammer-Purgstall. Michael Bleyer is financed by the Austrian Science Fund (FWF) under project P19797. Christoph Rhemann is supported by Microsoft Research Cambridge through its PhD Scholarship Program.



(a) (b) (c)

Fig. 1. Support regions for selected windows of the Middlebury images. (First row) Image crops. (Second row) Support weights computed by the adaptive weight method [3]. The segmentation method gives relatively high support weights to pixels whose disparity is different from that of the center pixel (see arrows). (Third row) Our geodesic support weights. Due to enforcing connectivity, such wrong high weights are avoided.

8-connectivity $\{p_1, p_2, \cdots, p_n\}$. The costs $d()$ of a path are computed by

$$d(P) = \sum_{i=2}^{i=n} d_C(p_i, p_{i-1}). \tag{2}$$

with $d_C()$ being a function that determines the color difference. This function is implemented by

$$d_C(p,q) = \sqrt{(r_p - r_q)^2 + (g_p - g_q)^2 + (b_p - b_q)^2} \tag{3}$$

where r_p, g_p and b_p are the values of p's RGB channels. Intuitively spoken, the geodesic distance from pixel p to the window center c is low, if there exists a path between these points along which the color varies only slightly. We refer to this property as *connectivity*.

We assume that pixels of low geodesic distance lie on the same disparity as c itself. Low geodesic distances should therefore convert into high support weights. The function $w()$ implements this by

$$w(p,c) = \exp\left(-\frac{D(p,c)}{\gamma}\right) \tag{4}$$

with γ being a user-defined parameter that defines the strength of the resulting segmentation.

Once the support weights are know, they are exploited to aggregate pixel dissimilarities within the support window. The aggregated matching costs for a pixel c at disparity d are derived by

$$m(c,d) = \sum_{p \in \mathcal{W}_c} w(p,c) \cdot f(p, p-d) \tag{5}$$

with $\mathcal{W}_c$ representing all pixels of the square support window centered on pixel c. The window size is thereby given by the user. The function $f(p,q)$ computes the color dissimilarity between a pixel p of the reference image and a pixel q of the match view. In principle, one can use equation (3) to implement $f()$. However, we have chosen Mutual Information [9] in order to handle illumination differences that may exist between the two input images. In addition, we put an upper bound f_{max} on the pixel dissimilarity so that values exceeding f_{max} are truncated. This serves to reduce the influence of occluded pixels [10].

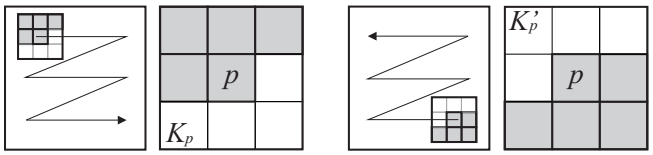
Finally, we obtain a disparity map by determining the disparity d_p of each pixel p of the reference view using local optimization:

$$d_p = \underset{d \in \mathcal{D}}{\operatorname{argmin}}\, m(p,d) \tag{6}$$

where $\mathcal{D}$ represents the set of all allowed disparities. In fact, there is nothing that speaks against replacing this Winner-Takes-All strategy with a global optimization technique such as graph-cuts or belief propagation. This would most likely further improve the quality of results.

2.2. Geodesic versus Original Adaptive Weights

Let us use the support weight examples of figure 1 to illustrate the advantage of our geodesic approach over the method of [3]. Figure 1a shows an artificial image in which a foreground object is placed in front of a background object of similar color. For computing the support weights, the original adaptive weight method just considers the color difference and the distance to the center pixel. In our example, it therefore erroneously gives high weights to background pixels. In contrast to this, our approach can successfully handle this example by considering the whole structure of the image patch. Recall that our geodesic computation requires that there is a path of approximately constant color to the center pixel. For background pixels, such a constant color path does



(a) (b)

Fig. 2. Efficient approximation of the geodesic distance inside the support window. (a) Forward pass. (b) Backward pass.

not exist due to the edge that separates the foreground from the background object. Figures 1b and 1c show this effect on real-world images. Figures 1b thereby uses the same image patches as in the paper of [3].

It is important to notice that the computational efficiency of our geodesic support weight strategy is practically the same as that of [3]. For both methods, the computation of a single weight mask has a complexity of $O(|\mathcal{W}|)$ with $\mathcal{W}$ being the set of all pixels inside the square window. (We describe an efficient algorithm for approximating the geodesic weight mask in section 2.3). However, both methods share the performance bottleneck that originates from evaluating the window at each disparity. This pixel comparison operation has complexity $O(|\mathcal{W}| \cdot |\mathcal{D}|)$ with $\mathcal{D}$ denoting the set of allowed disparities. Due to using adaptive weights, this operation cannot easily be speeded up using sliding window techniques from which local methods commonly take their high efficiency. Nevertheless, it has been shown [7] that by GPU programming a real-time implementation of [3] can be accomplished. This would most likely also work for our method. In our current implementation, it takes approximately a minute to compute the disparity map for standard test image pairs.

2.3. Approximation of Geodesic Distances

We apply the method of [11] for efficiently approximating the geodesic distance of each pixel within the window to the center pixel (equation (1)). This method is reviewed as follows.

Each pixel p inside the window is assigned to costs $C(p)$. Initially, the costs of the center pixel are set to 0, while the costs of all other pixels are set to a large constant value. In the forward pass of the algorithm, we traverse the support window in row major order (see figure 2a). The costs of a pixel p are thereby updated by

$$C(p) := \min_{q \in K_p} C(q) + d_C(p, q) \tag{7}$$

with the kernel K_p being a set of pixels consisting of p itself as well as its left, left upper, upper and right upper neighbours (figure 2a). The cost update is thereby performed immediately so that the new costs already affect the cost computation of the next pixel. Once the forward pass is completed, we invoke the backward pass. This pass traverses the window in reverse direction (figure 2b). It thereby updates the costs using equation (7) in conjunction with the kernel K_p' of figure 2b. Forward and backward passes are iterated. (In our experiments, we found 3 iterations to be sufficient for giving reasonable results.) The final costs $C(p)$ represent our estimate of the geodesic distance of p to the center pixel.

2.4. Occlusion Detection and Filling

Up to this point we have ignored the occlusion problem. In order to detect occlusions, we follow common practice in local stereo by applying left-right consistency checking. We therefore use the algorithm of section 2.1 to compute a first disparity map with the left image chosen as reference frame. A second disparity map is derived by choosing the right image as reference. For each pixel of the left disparity map, we then check whether it carries the same disparity assignment as its matching point in the right disparity map. If this is not the case, the pixel is invalidated. This left-right check is effective in filtering out occluded pixels as well as mismatches.

In the occlusion filling step, we assign invalidated pixels to new disparity values. For each invalidated pixel p, we estimate p's first valid neighbour to the left l and to the right r. The disparity d_p is then computed by $\min(d_l, d_r)$. This simple filling strategy typically generates horizontal streaks. To eliminate these artifacts, we apply a smoothening filter on the invalidated pixels. We thereby use a weighted median filter with the filter weights obtained from the precomputed geodesic weight masks of section 2.1. In comparison to standard median filtering, our weighted median filter does not suffer from the problem of distorting object boundaries.

3. RESULTS

We have used the Middlebury stereo benchmark [2] to evaluate the performance of our approach. In our test run, the algorithm's parameters are set to the constant values of $\gamma := 10$ and $f_{max} := 120$. The window size is chosen to be 31. These parameters have been found empirically. We plot our results on the Middlebury images along with corresponding error maps in figure 3. One can see that our algorithm performs well in the reconstruction of disparity borders, while it also finds correct disparities for regions of low texture. It is traditionally difficult for a local method to fulfill these two requirements at the same time.

Table 1 shows quantitative results that are taken from the Middlebury online table. Our algorithm currently takes the 10th rank of 63 submissions. This is specifically promising when considering that we do not use global optimization. This is in contrast to all better performing algorithms. According to the Middlebury table, our method is the currently best performing local method. It can outperform the original adaptive weight approach [3] relatively clearly.

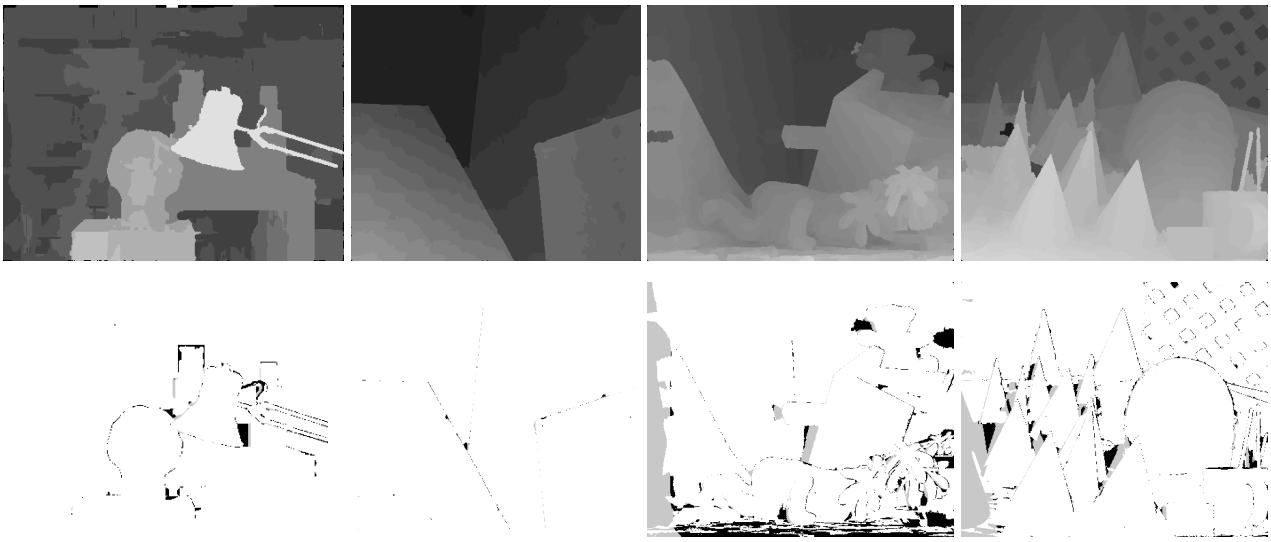


Fig. 3. Results on Middlebury images generated using constant parameter settings. The first row shows the results computed by our algorithm. The second row shows a comparison against the ground truth by plotting disparity errors larger than one pixel.

Algorithm	Rank	Avg. Error [%]	Error non-occluded pixels [%]			
			Tsukuba	Venus	Teddy	Cones
GeoSup	**10**	**5.80**	**1.45**	**0.14**	**6.88**	**2.94**
AdaptDispCalib	13	6.10	1.19	0.23	7.80	3.62
DistinctSM	18	6.14	1.21	0.35	7.45	3.91
CostAggrOcc [6]	20	6.20	1.38	0.44	6.80	3.60
SegmentSup [5]	21	6.44	1.25	0.25	8.43	3.77
AdaptWeight [3]	26	6.67	1.38	0.71	7.88	3.97
SSD+MF [2]	59	15.7	5.23	3.74	16.5	10.6

Table 1. Rankings of selected local methods in the Middlebury online database. Our algorithm (denoted by *GeoSup*) is currently the overall best-performing local method.

4. CONCLUSIONS

This paper has proposed a novel support aggregation approach for stereo matching. To derive support weights, we have computed geodesic distances for all pixels of the support window to the window's center point. The advantage over previous work is that we implement the concept of connectivity that proves to be effective for obtaining improved segmentation results. We have tested our results using the Middlebury benchmark. According to the results, the proposed geodesic support weight approach is the top performer among stereo methods that rely on local optimization.

5. REFERENCES

[1] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," *PAMI*, vol. 16, no. 9, pp. 920–932, 1994.

[2] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *IJCV*, vol. 47, no. 1/2/3, pp. 7–42, 2002. http://www.middlebury.edu/stereo/.

[3] K.J. Yoon and I.S. Kweon, "Locally adaptive support-weight approach for visual correspondence search," in *CVPR*, 2005.

[4] M. Gerrits and P. Bekaert, "Local stereo matching with segmentation-based outlier rejection," in *CRV*, 2006.

[5] F. Tombari, S. Mattoccia, and L. Di Stefano, "Segmentation-based adaptive support for accurate stereo correspondence," in *PSIVT*, 2007.

[6] D. Min and K. Sohn, "Cost aggregation and occlusion handling with wls in stereo matching," *TIP*, vol. 17, no. 8, pp. 1431–1442, 2008.

[7] M. Gong, R. Yang, L. Wang, and M. Gong, "A performance study on different cost aggregation approaches used in real-time stereo matching," *IJCV*, vol. 75, no. 2, pp. 283–296, 2007.

[8] F. Tombari, S. Mattoccia, L. Di Stefano, and E. Addimanda, "Classification and evaluation of cost aggregation methods for stereo correspondence," in *CVPR*, 2008.

[9] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *PAMI*, vol. 30, no. 2, pp. 328–341, 2008.

[10] T. Noguchi and Y. Ohta, "A simple but high-quality stereo algorithm," in *ICPR*, 2002.

[11] G. Borgefors, "Distance transformations in digital images," *Computer Vision, Graphics and Image Processing*, vol. 34, no. 3, pp. 344–371, 1986.