AN IMPROVED NON-LOCAL COST AGGREGATION METHOD FOR STEREO MATCHING BASED ON COLOR AND BOUNDARY CUE

Dongming Chen, Moshen Ardabilian, Xiaofang Wang, Liming Chen

Ecole Centrale de Lyon, LIRIS UMR 5205, F-69134, Lyon, France

ABSTRACT

Recently, a novel Non-Local Cost Aggregation (NLCA) algorithm based on a minimum spanning tree (MST) has been proposed [1] for stereo matching providing extremely low computational complexity and outstanding performance. However, since the MST is constructed only based on color cue, this approach suffers from some improper connections at the boundaries of two objects with similar color distribution. To improve this drawback, in this paper, we propose a new weight function which includes not only the color cue but also the boundary cue of the reference image. Because the boundary cue is more discriminative than the color cue to distinguish the object from the background with similar color, the previous improper connections can be avoided and a more faithful MST can be constructed. Experimental results performed on Middlebury benchmark demonstrate the effectiveness of the improvements. The improved algorithm achieves rank 18th out of 143 submissions, while the original algorithm ranks 31st.

Index Terms— stereo matching, minimum spanning tree, non-local cost aggregation, boundary cue

1. INTRODUCTION

Dense two-frame stereo matching, as one of the most active research topics in computer vision, has widely attracted attention during the last decades. Most stereo matching algorithms can be roughly categorized into two streams [2]: global and local method. The former usually formulates an energy function composed of a non-smooth data-fidelity term and a smooth regularization term \( E = E_{\text{data}} + E_{\text{smooth}} \); the matching problem is equivalent to the minimization of this defined energy. Graph cut [3] and belief propagation [4] are most popular tools used in global methods. Although these optimizers can produce more accurate results, they are usually computationally expensive for the slowly convergence speed of the optimization process. On the other hand, in local methods, the matching cost is calculated using the intensity values at a given disparity and aggregated within a finite support window; then the optimal solution can be solved by a fast local optimization, “winner-take-all” strategy.

The performance of local methods mainly depends on the support windows they choose. A variety of cost aggregation approaches have already been proposed, such as the square window approach [2], the shiftable window approach [5] [6], the adaptive window approach [7] [8], the adaptive weight approach (also called the edge-preserving filter based approach) [9] [10] and so on. To our knowledge, the adaptive weight approach represents the state-of-the-art in local methods. Yoon and Kweon [9] presented an adaptive weight approach, in which the cost aggregation is equivalent to apply an edge-preserving filter (bilateral filter [11]) on the cost volume. It can achieve high accuracy results, but it is time-consuming since the naive implementation of bilateral filter is a relatively slow process. As an improvement, Fast Bilateral Stereo [12] is proposed, combining the efficiency of fast incremental calculation schemes (integral images) with the effectiveness of adaptive weight approach; Richardt et al. [13] proposed an approximation but faster implementation of bilateral filter. Recently, guided filter [14] was put forward, which can perform as an edge-preserving filter; but unlike bilateral filter, it has better behavior near the edges and its computational complexity is independent of the filter size. Inspired by the guided filter, De-Maeztu et al. [15] proposed a symmetric and color-based O(1) cost aggregation strategy with complexity independent of the window size. Rhemann et al. [10] applied the guided filter on stereo matching, and their guided filter based algorithm outperformed all the other local methods in terms of both speed and accuracy on the Middlebury benchmark [16]. Different from all of these local methods, which aggregate the matching cost within a local window, Yang [1] first proposed a Non-Local Cost Aggregation (NLCA) stereo matching algorithm based on a minimum spanning tree (MST) computed from the reference image. Every pixel receives supports from all other pixels through the MST. This algorithm is very fast (it is about 10 \times and 70 \times faster than guided filter based algorithm [10] for cost aggregation and disparity refinement respectively). However, in terms of accuracy, the NLCA algorithm ranks 31st on the Middlebury benchmark, while guided filter based algorithm [10] ranks 25th.

The NLCA algorithm consists of five steps: 1) matching cost computation; 2) MST construction; 3) cost aggregation on MST; 4) winner-take-all optimization; 5) disparity refinement on MST. Both the cost aggregation step and the disparity
refinement step are performed on the MST. In the cost aggregation step, the cost of each node (image pixel) is aggregated from its parent node and children nodes on the MST. In other words, once the MST is constructed, the path for cost aggregation is fixed. Thus, the quality of MST directly influences the final result. Firstly, we briefly describe the MST construction step. The reference image is represented as a connected, weighted, undirected graph $G = (V, E)$, in which $V$ is a set of vertices (image pixels) and $E$ is a set of edges. Each edge, connecting two neighboring vertices, is mapped to a real-valued weight (the similarity of these two vertices) by a weight function. The MST is defined as a sub-graph of $G$, which connects all vertices and the sum of its weights is minimum out of all spanning tree. Therefore, the weight function directly decides the graph $G$ and indirectly decides the MST. The MST construction of the original NLCA algorithm is not perfect because only the color cue is considered in weight function, which leads to some mistakes at the boundary of two objects with similar color distribution (we call such boundary as weak boundary for short). In our weight function, we consider not only the color similarity of neighboring pixels (color cue) but also their possible depth discontinuity (boundary cue), because boundary is more powerful than color to measure the similarity of pixels at boundary areas, especially at weak boundary areas. Fig.1 shows an example, (a) is the reference image [16] (named Tsukuba), the area in red box is zoomed in (b). We focus on the area in the white box in (b), called the region of interest (ROI). In the bottom of ROI, the color of the black stick of tripod is similar with the background, thus the MST, constructed by the original algorithm, incorrectly connect the black stick with the background as shown in (c) (the white points represent the edges of MST and the blue lines are used to show the connect relationship between the stick and background), (d) presents the disparity map of the original algorithm, in which the red circle indicates the mistake area. However, in our MST, the pixels of stick are separate from background (e), due to the boundary cue (in (g) and zoomed in (h)). We obtain a better result (f).

The rest of this paper is organized as follows: in section 2, we briefly introduce the original NLCA algorithm. The improved NLCA algorithm based on a new weight function is described in detail in section 3. Then, the experimental results and analysis are given in section 4, and section 5 concludes the paper.

2. THE ORIGINAL NLCA ALGORITHM

The NLCA algorithm [1] consists of five steps: matching cost computation; MST construction; cost aggregation on MST; winner-take-all optimization and disparity refinement on MST. The details are introduced as follows.

1) Matching cost computation: the matching cost is considered to be a truncated absolute difference of the color and the gradient at a pair of matching points (pixel $p$ in left image $I_l$ and pixel $q$ in right image $I_r$).
Then, the MST of graph \( G \) can be calculated by Kruskal’s algorithm [17].

3) Cost aggregation on MST: this process can be divided into two steps:

(i) Aggregating the raw cost from leaf nodes towards the root node of the MST. Let \( v_i \) denote the \( i \)th node on MST, \( P(v_i) \) denote the parent of \( v_i \). The aggregated cost of node \( v \) (\( v \in V \)) at disparity \( d \), \( C^A(d)(v) \), is defined,

\[
C^A(d)(v) = C_d(v) + \sum_{P(v_i)=v} S(v,v_i)C^A(v_i) \tag{3}
\]

where, \( S(v,v_i) = exp\left(\frac{-w(v,v_i)}{\sigma}\right) \) measures the similarity between \( v_i \) and its parent \( v \), \( \sigma \) is a constant used to adjust the similarity between two nodes. If \( v \) is a leaf node, then \( C^A(d)(v) = C_d(v) \).

(ii) The final aggregated cost \( C^A_d \) can be obtained by aggregating \( C^A_d \) from the root node towards leaf nodes using the follow formula,

\[
C^A_d(v) = S(P(v),v) \times C^A(P(v)) + [1 - S^2(v,P(v))] \times C^A_d(v) \tag{4}
\]

Note that \( C^A_d(v) = C^A_d(v) \) for the root node.

4) Winner-take-all optimization: it is commonly used in local methods, which searches for the disparity value that gives the smallest matching cost,

\[
D(v) = \arg \min_d (C^A_d(v)) \tag{5}
\]

5) Disparity refinement: the left and right disparity maps are firstly obtained using the above four steps, taking the left and right image as reference image respectively. Then, the mutual consistency check is employed on these two disparity maps to pick out the unstable pixels; the cost value of unstable pixels are reset to zero for all disparity levels, while the cost value of stable pixels at disparity level \( d \) are reset to \( |d - D(v)| \) (pixel \( v \) is stable and \( D(v) > 0 \)). Finally, the cost aggregation algorithm (step 3) is performed on this new cost again. The new aggregated cost of unstable pixels is propagated from those of stable pixels.

and pixel \( p + d \) in right image \( I_r \). Let \( C_d(p) \) denote the raw matching cost of pixel \( p \) at disparity \( d \).

\[
C_d(p) = (1 - \theta) \times \min(||I_l(p) - I_r(p + d)||, \tau_1) + \theta \times \min(||\nabla_x I_l(p) - \nabla_x I_r(p + d)||, \tau_2) \tag{1}
\]

Where \( \nabla_x \) is the gradient in \( x \) direction, \( \theta \) balances the color and gradient terms and \( \tau_1, \tau_2 \) are truncation values.

2) MST construction: the reference image \( I \) is represented as a connected, weighted, undirected graph \( G = (V,E) \), in which \( V \) is a set of vertices (pixels) and \( E \) is a set of edges connecting two neighboring vertices. The weight value of edge between pixel \( p \) and \( q \) is obtained using weight function:

\[
w(p,q) = |I(p) - I(q)| \tag{2}
\]

Then, the MST of graph \( G \) can be calculated by Kruskal’s algorithm [17].

3. THE IMPROVED NLCA ALGORITHM

In this section, we discuss the proposed weight function in detail, including two parts, color weight function and boundary weight function. The weight function is used to measure the similarity between two neighboring pixels. The color weight function measures the similarity using intensity value and boundary weight function measures it using edge strength of the reference image. As shown in Fig. 1, we expect that the pixels in the same depth of the same object should be strongly connected with high weight value, while those pixels in different depth should be weakly connected with small weight values or disconnected.

3.1. Color Weight

The color cue has been proved to be powerful to measure the similarity between two neighboring pixels [18]. In this paper, we calculate the color weight in RGB space. Formally, the weight between two neighboring pixels \( p \) and \( q \) in each channel is computed as follows,

\[
w_i(p,q) = \exp(-\frac{(I_i(p) - I_i(q))^2}{\sigma_i I_{max}}), \quad (i \in \{r,g,b\}) \tag{6}
\]

where \( I_i(p) \) denotes the intensity values of pixel \( p \) in R, G, B channels respectively. \( I_{max} \) is the maximum value of the whole image \( I \). \( w_i \) denote the color weights of R, G and B channels. Parameter \( \sigma_i \) is used to adjust the similarity. The final color weight \( w_{color} \) between \( p \) and \( q \) is computed by combining the weights of these three channels,

\[
w_{color}(p,q) = w_r x w_g x w_b \tag{7}
\]

3.2. Boundary Weight

The boundary weight function is based on the edge strength of the image. Given image \( I \), the edge strength at pixel \( p \) is
Parameter are simply combined using an empirical formula with a parameter $\alpha$.

$E(p) = \sqrt{(I(p) * F_o)^2 + (I(p) * F_e)^2}$  \hspace{2cm} (8)

where $\ast$ is the convolution operator. As shown in Fig.2, the odd-phase filters $F_o$ and the even-phase filters $F_e$ are a pair of quadrature filters, differing in their spatial phases. They are tuned to detect boundaries of different shapes, parameterized by scale, elongation, and orientation respectively. The odd-phase filters are essentially the first-order derivatives, whereas the even-phase filters are the second-order derivatives, both smoothed with differences of offset Gaussian.

$E(\cdot)$ has a maximum response for contours of shape, whereas the zero-crossings of filter $F_e(\cdot)$ locate the positions of the edges. The boundary weight $w_{boundary}$ is formulated by measuring the magnitude of image boundary between two pixels:

$$w_{boundary}(p, q) = \exp\left(\frac{(E(p) - E(q))^2}{(\sigma_e \times E_{max})^2}\right)$$  \hspace{2cm} (9)

where $E_{max}$ is the maximum value of edge strength $E(\cdot)$. Parameter $\sigma_e$ decides the width of Gaussian kernel.

3.3. Weight Combination

The color weight $w_{color}$ and the boundary weight $w_{boundary}$ are simply combined using an empirical formula with a parameter $\alpha$ to form a new weight $w_{mix}$:

$$w_{mix} = \sqrt{w_{color} \times w_{boundary}} + \alpha \times w_{color}$$  \hspace{2cm} (10)

Then, the weight $w_{mix}$ is used to calculate the edge $E$ of graph $G$ in MST construction step (step 2 in Section 2).

3.4. Computational Complexity

The proposed weight function includes color weight function and boundary weight function. The computational complexity of color weight function (Eqns.(6~7)) is $O(n)$, which is the same as that of original algorithm. In the computation of edge strength (Eqn.(8)), the odd-phase filters $F_o(p)$ and the even-phase filters $F_e(p)$ can be pre-computed and are constants for all test images. Since image $I$ is convoluted with odd-phase filters and even-phase filters, which consists of discrete Fourier transform (DFT) and inverse discrete Fourier transform (iDFT) (the computation complexity of both DFT and iDFT is $O(n\log n)$), the computation complexity of boundary weight function Eqn.(8~9) is $O(n\log n)$.

4. EXPERIMENTAL RESULTS

We evaluated our improved NLCA algorithm on the Middlebury benchmark [16], using four pairs of standard datasets, "Tsukuba", "Venus", "Teddy" and "Cones". Our method is implemented in C++ and all experiments have been conducted on a PC with 3.4GHz processor and 8GB memory.

4.1. Accuracy

We compare our improved algorithm with the original algorithm [1] and the guided filter based algorithm [10]. The quantitative evaluation is presented in Table 1 and visual comparison is shown in Fig.5. We firstly compare our improved algorithm with the original algorithm. For fair comparison, we used the same matching cost computation, cost aggregation and refinement strategy for both algorithms. On the Middlebury benchmark [16], the error percentage are evaluated over three different areas in the reference image, classified as non-occlusion (nonocc.), discontinuous (disc.) and the entire image (all). As shown in Table 1, our improved algorithm is more accurate than the original algorithm in all these three areas of each image, especially in the 'disc' column. The term 'disc' represents the regions near depth discontinuities (white areas in Fig. 3). In this column, the errors are only evaluated in these white areas, which mainly contain boundaries. Thus the good performance in 'disc' column indicates the necessary of considering boundary cue in our weight function. Moreover, our improved NLCA algorithm is more accurate than the guided filter based algorithm (rank 18th vs. 25th, error percentage 5.23% vs. 5.55%), which is the most accurate local algorithm.

In this experiment, we use the same constant parameters for all test images, both $\sigma_c$ and $\sigma_e$ are set to 0.1; $\alpha$ is set to 1.65. $\alpha$ determines the combination of color weight and boundary weight. We test the performance of our improved algorithm with respect to different $\alpha$. In Fig.4, we show the results in different $\alpha \in [1, 2]$, it is observed that our algorithm is robust.
Fig. 3. The term 'disc.' in Table 1 means the regions near depth discontinuities (white areas), occluded and border regions (black), and other regions (gray). In 'disc.' column, errors are only evaluated in the white areas.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Rank</th>
<th>Avg.</th>
<th>tuskuba</th>
<th>venus</th>
<th>teddy</th>
<th>cones</th>
<th>Avg. Error(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>noc all</td>
<td>disc</td>
<td>noc all</td>
<td>disc</td>
<td></td>
</tr>
<tr>
<td>improved NLCA</td>
<td>18</td>
<td>32.1</td>
<td>1.28_47</td>
<td>1.83_40</td>
<td>7.38_40</td>
<td>0.21_28</td>
<td>2.29_39</td>
</tr>
<tr>
<td>guided filter</td>
<td>25</td>
<td>36.8</td>
<td>1.51_51</td>
<td>1.85_43</td>
<td>7.61_44</td>
<td>0.30_24</td>
<td>6.19_38</td>
</tr>
<tr>
<td>original NLCA</td>
<td>31</td>
<td>39.6</td>
<td>1.48_52</td>
<td>1.85_43</td>
<td>7.88_63</td>
<td>0.54_41</td>
<td>6.01_34</td>
</tr>
</tbody>
</table>

Table 1. Quantitative comparison of the improved algorithm, the original algorithm and the guided filter based algorithm with error threshold 1. The subscript numbers indicate the rank in each column.

Fig. 5. Experimental results on the Middlebury datasets [16]. (a) the reference images. (b) the ground truth disparity maps. (c) and (d) are disparity maps obtained using the original algorithm and the proposed algorithm respectively.

4.2. Speed

We test both our improved algorithm and the original algorithm [11] on the same computer and dataset. The difference between our improved algorithm and the original algorithm is the weight function. We need to calculate the edge strength of the image. The theoretic computational complexity of our
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Overall Runtime (second)</th>
</tr>
</thead>
<tbody>
<tr>
<td>improved NLCA</td>
<td>0.588</td>
</tr>
<tr>
<td>orignal NLCA</td>
<td>0.248</td>
</tr>
<tr>
<td>tsukuba</td>
<td>0.861</td>
</tr>
<tr>
<td>venus</td>
<td>1.493</td>
</tr>
<tr>
<td>teddy</td>
<td>1.464</td>
</tr>
<tr>
<td>cones</td>
<td>0.828</td>
</tr>
</tbody>
</table>

Table 2. Quantitative runtime comparison of the improved algorithm and the original algorithm.

weight function is discussed in Section 3.4. The practical runtime is presented in Table 2. The overall runtime of our algorithm is slightly (1.9 times) slower than the original algorithm on average, but much faster than the guide image filter based algorithm \[10\].

5. CONCLUSION

This paper has improved the non-local cost aggregation algorithm \[1\]. We have reformed the weight function by incorporating not only color cue but also boundary cue to improve original MST’s performance. According to the experimental results on Middlebury dataset, our improved algorithm (ranks 18\textsuperscript{th} out of 143 submissions, error percentage 5.23\%) is much more accurate than the original algorithm (ranks 31\textsuperscript{st}, error percentage 5.48\%). Moreover, our improved algorithm is better than guide image filter based algorithm \[10\] (ranks 25\textsuperscript{th}, error percentage 5.55\%), which is the most accurate local stereo matching algorithms.

6. REFERENCES