An Improved Adaptive Window Stereo Matching Algorithm

Wenbo Qiao¹, Yuanping Xu¹*, Chaolong Zhang²,³, Zhijie Xu², Jian Huang¹, Pan Xie¹ and Jun Lu¹

¹ School of Software Engineering, Chengdu University of Information Technology, Chengdu, Sichuan, 610225, China
² School of Computing and Engineering, University of Huddersfield, Huddersfield, West Yorkshire, HD1 3DH, UK
³ Chengdu CHG Robots & Intelligent Equipment Institute, Chengdu, Sichuan, 610045, China

*Corresponding author’s e-mail: ghxpy@hotmail.com.

Abstract. In order to solve the problem that the existing adaptive window stereo matching algorithms have insufficient feature extraction in low-texture regions, resulting in low matching accuracy. An adaptive window stereo matching algorithm based on the gradient is proposed. Firstly, the Sobel operator is used to extract the gradient value of each pixel in the image. Then, each pixel is divided into high, medium and low texture regions according to the gradient value. Next, different arm length thresholds are assigned to different region pixels and matching windows are generated dynamically according to arm length and color threshold. Finally, the pixels closer to the center of the window are given higher weights by generating windows several times. It solves the problem that the stereo matching algorithm cannot select a matching window dynamically. Experimental results on Middlebury dataset show that the proposed method improves the matching accuracy by 5.5% compared with the latest adaptive window stereo matching algorithm.

1. Introduction

Binocular stereo vision is one of the most important branches of computer vision, and binocular stereo vision technology is mainly composed of camera calibration, image correction, stereo matching and 3D reconstruction[1]. The task of the stereo matching algorithm is to calculate the pixels with the same name in two images and to generate the disparity map. Stereo matching, as a prerequisite for obtaining 3D information in binocular stereo vision, is the most critical step in the whole process. In the stereo matching algorithm, how to select a matching window dynamically is very important[2], many researchers have done a lot of work on the adaptive window stereo matching algorithm. Zhang et al.[3] proposed an adaptive window stereo matching algorithm according to the relationship between pixel color information and spatial distance, which provided a good solution for how to select a matching window. However, this method uses a single weight for all pixels in the window, so the matching accuracy is poor. Veksler et al.[4] proposed a point by point adaptive selection method to obtain the appropriate matching window. However, this method needs to do a lot of computation on each pixel, so the efficiency of the algorithm is very low. Qu et al.[5] used the color difference between surrounding pixels and central pixels to construct an adaptive matching window. Fusiello et al.[6] proposed a matching algorithm that sets nine different window types for each pixel and retains the minimum matching cost of window disparity. However, due to the limited window types, it cannot
summarize the size and shape of all types of windows, the robustness of the algorithm is poor. Based on Zhang’s method, Lv et al.[7] proposed a method of multiple generations of adaptive windows, and introduced the method of multiple weights. However, this method does not consider the difference between low texture region and high texture region, and the matching accuracy is poor in the low texture region.

Based on the above problems, this paper proposes a gradient-based adaptive multi-weighted window stereo matching algorithm. Firstly, the gradient information of each pixel is extracted by the Sobel operator. According to the gradient information, the pixel is divided into high texture, medium texture, or low texture region. Then the transverse arm length threshold of the window is determined according to the gradient of high, medium and low regions, and the adaptive window is generated by combining the longitudinal arm length threshold and color threshold. Finally, the weight of pixels is introduced into the algorithm by generating adaptive windows many times. This paper compares the algorithm proposed by Zhang and Lv with the algorithm proposed in this paper on the Middlebury platform. Experiments show that the algorithm proposed in this paper has higher matching accuracy in object edge and low texture region.

2. Proposed Algorithm
The workflow of the proposed algorithm is shown in figure 1. First, we need to calculate the gradient of each pixel to initialize the lateral arm length threshold. Then the matching window is generated by three times according to the threshold. Finally, the matching cost of the image is calculated by the matching algorithm, and the final disparity map is generated.

![Algorithm workflow](image1)

**Figure 1. Algorithm workflow**

2.1. Initialize Parameters
To add the weight of regional texture information to the adaptive window algorithm, firstly, Sobel operator is used to extracting the gradient information of each pixel. As shown in figure 2, the Sobel operator contains two $3 \times 3$ convolution kernels, which can calculate the horizontal and vertical gradients of pixels, respectively.

![Sobel operator](image2)

**Figure 2. Sobel operator**
The horizontal gradient $G_x$ and vertical gradient $G_y$ are calculated according to the Sobel operator in two directions, and the final gradient $G = \sqrt{G_x^2 + G_y^2}$. The analysis shows that the gradient value of the high texture region is larger, and a smaller window should be selected. However, the gradient change of the low texture area is very small, so we need to choose a larger window. Therefore, we can determine the threshold of the lateral arm length of the pixel according to the gradient. Let the current pixel lateral arm length threshold as $L$, three lateral arm length thresholds $L_{h1} > L_{h2} > L_{h3}$, and double threshold $T1 > T2$. The specific judgment formula of the lateral arm length threshold is shown in formula (1).

\[
L = \begin{cases} 
L_{h3}, & G > T1 \\
L_{h2}, & T2 < G < T1 \\
L_{h1}, & G < T2 
\end{cases}
\]  

(1)

2.2. Generate Matching Window

After determining the lateral arm length threshold of each pixel, as shown in figure 3, the matching point $p$ in the image is selected, and the cross window skeleton is constructed with the $p$ point as the center.

In figure 3, the four arm lengths \{HP-, HP+, VP-, VP+\} of the cross skeleton are controlled by lateral arm length threshold, longitudinal arm length threshold and color threshold. The discriminant conditions are shown in formula (2,3,4).

\[
\max_{c \in \{R, G, B\}} |I_c(p_i) - I_c(p)| < \tau 
\]  

(2)

\[
|L_h(p_i) - L_h(p)| < L_h
\]  

(3)

\[
|L_v(p_i) - L_v(p)| < L_v
\]  

(4)

In above formulas, $p_i$ is a pixel on the same row or column of point $p$. $I_c(p_i)$ represents the value of this point in RGB space, and $\tau$ is the selected color threshold. In Formula 3, $L_h(p_i)$ represents the abscissa of the pixel, and $L_h$ is the threshold of the lateral arm length of each pixel calculated in the previous step. In formula 4, $L_v(p_i)$ represents the ordinate of the pixel, and $L_v$ is the selected threshold of longitudinal arm length. According to the above criteria, window expansion is carried out in four
directions with point $p$ as the center. When a certain direction does not meet any of the above conditions, the extension ends. Finally, a cross skeleton region centered on point $p$ is formed. In figure 3, it is $\{HP- \cup HP+ \cup VP- \cup VP+\}$. For all $q$ points in the vertical direction, the expansion process is repeated in the horizontal direction to obtain the region $\{Hq- \cup Hq+\}$. The final adaptive window area is denoted as $A(p)$, as shown in equation 5.

$$A(p) = h_p^- \cup h_p^+ \cup v_p^- \cup v_p^+ \cup h_{qV(q)}^- \cup h_{qV(q)}^+$$ \hspace{1cm} (5)

According to the above steps, an adaptive window can be generated. In order to give a higher weight to the pixels near the center of the window, the window is generated by changing the arm length threshold several times, and the pixels generated in each round are given different weights. Let the generation coefficient be $\lambda$, and taking the cubic generation window as an example, and taking cubic generation coefficient $\lambda_1 = 1$, $\lambda_2 = 1.2$, $\lambda_3 = 1.5$. Taking point $p$ as an example, if point $p$ is the point in the high texture region, the lateral arm length threshold of point $p$ in the first round is $L_h p$ and that of longitudinal arm length threshold is $L_v$. In the second round, the lateral arm length threshold is $1.2L_h p$, and the longitudinal growth is $1.2L_v$. In the third round, the lateral arm length threshold is $1.5L_h p$, and the longitudinal arm length threshold is $1.5 L_v$. In addition, the weight $\alpha$ is assigned to each point in the region based on the rounds. Among them, $\alpha 1 = 3$ in the first round, $\alpha 2 = 2$ in the second round and $\alpha 3 = 1$ in the third round. The final generated matching window is shown in figure 4.

![Figure 4. Final generated matching window](image)

In figure 4, the point $p$ is taken as the center, and the darkest gray area is the adaptive window area generated in the first round, which is the closest to the center point with the weight value of 3; the medium gray area is the adaptive window area generated in the second round with the weight value of 2; the light gray area is the adaptive window area generated in the third round, with the weight value of 1.

### 2.3. Cost Matching

After generating the adaptive window, we need to search the similarity of window pixels in a specific range. In this paper, the Sum of Absolute Differences (SAD) algorithm is used to evaluate the similarity of the two windows.
As shown in figure 5, a point C in A graph is selected as the center point to be matched to generate an adaptive reference window. Then the matching window is generated at the same position in the B graph. In the maximum search range S, matching windows are generated one by one. Then the similarity measure function is used to evaluate the similarity between the matching window and the reference window. After evaluating S+1 matching windows, the center point of the window with the largest similarity is selected as the best matching point of the matching point. SAD[8] algorithm uses similarity measure function as shown in formula 6:

\[ SAD(i, j) = \text{Sum}(|\text{Left}(i, j) - \text{Right}(i, j)|) \]  

(6)

After searching S+1 matching windows in the maximum search range, the two window’s center pixel with the smallest cost matching are selected as the best matching points. Finally, the abscissa difference of the best matching points is recorded, which is the disparity of two points, and the disparity map is generated according to the disparity.

3. Experiment and Analysis
The datasets used in this paper is the Middlebury dataset. And we use three standard images of Baby, Bowling and Flowerpots provided by the Middlebury database to test. The codes are written with C++ on Visio Studio 2019, Windows 10 with Intel Core i7-9570H @ 2.6GHz and 8GB RAM. As shown in figure 6, on the Middlebury dataset, by comparing the results of the improved algorithm proposed by Zhang, and the improved algorithm proposed by Lv, the matching accuracy of the proposed algorithm is verified.
Figure 6. (a) input left image (b) input right image (c) disparity map generated by Zhang’s algorithm (d) disparity map generated by Lv’s algorithm (e) disparity map generated by the proposed algorithm

From the three sets of disparity maps, it can be seen that compared with the other two adaptive stereo matching algorithms, the proposed algorithm has higher matching accuracy in the object edge region and the low texture region. Table 1 shows the evaluation value of the three algorithms on the Middlebury website platform. In table 1, all is the algorithm's mismatch rate on the entire image, nonocc is the algorithm's mismatch rate on the low texture region, and disc is the algorithm's mismatch rate on the object edge region. The mismatch rate of the three algorithms in each region on three images is shown in Table 1.

Table 1. Mismatch rate of algorithms

| Algorithms | Baby          |          |          | Bowling      |          |          | Flowerpots |
|------------|---------------|----------|----------|-------------|----------|----------|------------|
|            | all | nonocc | disc | all | nonocc | disc | all | nonocc | disc |
| Zhang’s    | 19.8 | 7.8    | 9.4    | 25.7 | 10.6   | 9.8    | 17.3 | 8.3     | 7.5 |
| Lv’s       | 13.5 | 6.9    | 5.2    | 16.3 | 7.1    | 5.3    | 16.7 | 6.3     | 4.3 |
| Ours       | 9.3  | 2.7    | 3.1    | 10.5 | 4.3    | 3.2    | 10.2 | 3.5     | 3.6 |

It can be seen from table 1 that, compared with the algorithm proposed by Zhang, in the three images, the algorithm proposed in this paper reduces the average mismatch rate on the entire image by 10.93%. The average mismatch rate in the low-texture region dropped by 5.8%. The average mismatch rate in the edge region of the object dropped by 3.8%. Compared with the algorithm proposed by Lv, the average mismatch rate of the proposed algorithm in the whole image has dropped by 5.5%. The average mismatch rate in the low-texture area decreased by 3.6%. The average mismatch rate in the edge area of the object decreased by 1.6%. In summary, the proposed algorithm has a higher matching accuracy than the current adaptive window stereo matching algorithm.
4. Conclusions
In this paper, an improved adaptive window stereo matching algorithm is proposed. Firstly, we control the size of the generated window by gradient and then introduce multiple weights into the adaptive window stereo matching algorithm through multiple generations. Finally, we use the SAD algorithm to aggregate the cost to generate the final disparity map. We do comparative experiments on three sets of standard images provided by the Middlebury dataset. The experimental results show that the proposed algorithm improves the matching accuracy better than the current adaptive stereo matching algorithm.

Acknowledgments
This work was supported by the NSFC (61203172), the Sichuan Science and Technology Programs (2018JY0146, 2019YFH0187, and 2019104).

References
[1] Sui J., Jin W. (2004). The Realization and Development of Binocular Stereo Vision Technology. Application of Electronic Technology. 030(010):4-6,12.
[2] Gong W., Gu G., Qian W., et al (2014). Stereo matching algorithm based on the inter color correlation and adaptive support weight. Chinese Journal of Lasers. 41(8): 250-255.
[3] Zhang K., Lu J., Lafruit G. (2009). Cross—based local stereo matching using orthogonal integral images. IEEE Transactions on Circuits & Systems for Video Technology. 19 (7):1073—1079.
[4] Vekster O (2003). Fast variable window for stereo correspondense using integral images. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognitions. pp. 1: I-556-I561.
[5] Qu Y., Jiang J., Deng X., et al (2013). Robust local stereo matching under varying radiometric conditions. Iet Computer Vision. 8(4):263-276.
[6] Fusiello, A., Roberto, V., & Trucco, E. (1997). Efficient stereo with multiple windowing. In: The IEEE Computer Society conference on computer vision and pattern recognition. pp. 858-863.
[7] Lv P., LI X. (2018). Stereo Matching Algorithm Based on AD—census and Multi—weight Adaptive Window. Industrial Control Computer. 031(003):49-52.
[8] Yu C., Zhang J. (2019). Research on SAD-based stereo matching algorithm. Journal of Shenyang University of Aeronautics and Astronautics. 36(4):77-83.