Front. Comput. Sci. China 2011, 5(2): 227–235
DOI 10.1007/s11704-011-9190-2

RESEARCH ARTICLE

Efficient normalized cross correlation calculation method for stereo vision based robot navigation

Yehu SHEN (✉)

Department of System Integration and IC Design, Suzhou Institute of Nano-tech and Nano-bionics, Chinese Academy of Sciences, Suzhou 215125, China

Abstract Stereo vision systems are widely used for autonomous robot navigation. Most of them apply local window based methods for real-time purposes. Normalized cross correlation (NCC) is notorious for its high computational cost, though it is robust to different illumination conditions between two cameras. It is rarely used in real-time stereo vision systems. This paper proposes an efficient normalized cross correlation calculation method based on the integral image technique. Its computational complexity has no relationship to the size of the matching window. Experimental results show that our algorithm can generate the same results as traditional normalized cross correlation with a much lower computational cost. Our algorithm is suitable for planet rover navigation.

Keywords normalized cross correlation (NCC), stereo matching, integral image

1 Introduction

Autonomous robot navigation is a fundamental ability for mobile robots. It is still one of the most important research fields in the robotics community. The key component for autonomous robot navigation is the range sensor. Different kinds of sensors can be used for this purpose, to name a few, radio direction and ranging (RADAR), laser range finder, sonar, and video cameras, etc. Laser range finder and RADAR are commonly used in land based rovers for their high accuracy and large effective range [1,2]. Sonar is the first choice for underwater robots. It is also widely used in ground robots for short range obstacle detection. As computational power and image quality improve, computer vision based range sensors gradually emerge. Stereo vision [3,4] is the most widely used computer vision technique for acquiring range information. Images contain more information than other range sensors. Using modern techniques stereo vision can provide dense 3D point clouds in real-time. More and more autonomous robots are equipped with such stereo vision systems [5]. As the payload and power supply carrying capabilities are very restricted for planet rovers, such as lunar and mars rover, heavy and power consuming laser range finders and RADAR are rarely used in these applications. Furthermore, a camera is an all-solid-state approach that might be more easily space qualifiable. As a natural consequence, many planet rovers use stereo vision for autonomous navigation. Of all the stereo vision based planet rovers, NASA's Mars exploration rover (MER) project [6] may be the most famous and successful one.

Stereo vision has been an intense area of research for decades and its theoretical background is now quite mature. Stereo vision algorithms can be divided approximately into two groups [4]: local correspondence methods, and global correspondence methods. Generally speaking, the results of local correspondence methods perform more poorly than global correspondence methods but local correspondence methods are much faster. Block matching is the most commonly used local correspondence method. Block matching seeks to estimate disparity at a point in one image by comparing a small surrounding region, the template, with a series of small regions

Received January 12, 2010; accepted July 30, 2010

E-mail: yhshen2009@sinano.ac.cn

extracted from the other image, the search region [4]. For land based autonomous robot navigation, real-time application is most important. Furthermore, the computational power of the onboard computers is very restricted, so most robot navigation applications choose block matching methods. There are several commonly used metrics for block matching: sum of absolute differences (SAD), normalized cross correlation (NCC), and Census transform [7].

SAD is often used for computational efficiency [4]. Many commercially available real-time stereo vision systems use SAD [8,9]. Unfortunately, SAD is not robust to different illumination conditions between cameras, so image pairs are usually fed to high pass filters before matching [10] which increases computational complexity. Census transform is immune to different illumination conditions but it needs special hardware [11] to reach real-time performance which raises the total cost of the system. NCC is also robust to different illumination conditions. Unfortunately, its computational cost is much higher than SAD and can hardly be implemented in real time so it is not widely used in autonomous robot navigation.

There exists some prior work on effective implementation of NCC. Faugeras et al. [12] proposed a method based on a sliding window technique. But it can only be applied to specific modified versions of cross correlation instead of the standard version. Our method, however, can be applied to NCC without any difficulty. Converting cross correlation into efficient implementation of fast Fourier transform (FFT) is also a well known method. Unfortunately, it cannot be applied directly for the normalized version of cross correlation. Lewis [13] presented an algorithm by combining sum tables and FFT. It is well known that FFT is designed for a size which is equal to powers of two. So it is inefficient to apply to image of an arbitrary size. Hii et al. [14] proposed to combine FFT and sum tables for calculating the denominator of the NCC definition equation. They used sum-table and basis functions to estimate the numerator of NCC. Despite similar drawbacks to FFT based methods, the basis functions approach can only achieve an approximation of the numerator. Briechle and Hanebeck [15] proposed a similar efficient algorithm for the application of object recognition.

To overcome the drawbacks of the previous methods, we propose an efficient method to calculate NCC in the framework of integral image theory [16]. The computational complexity does not grow with the size of template window in block matching so offers potential for autonomous robot navigation applications. Furthermore, our algorithm does not require any special hardware so the cost of deployment is very low.

In Section 2, a brief introduction to the integral image technique is given. We then provide a detailed explanation of how to extend the original integral image technique to NCC calculation. Experimental results are given in Section 3. Section 4 provides our conclusions.

2 Efficient NCC calculation based on integral image techniques

2.1 Integral image

Summed-area tables were introduced by Crow [17] for texture mapping. Viola and Jones [16] introduced them into computer vision for efficient feature calculation, renaming the technique *integral image*. Veksler [18] used the integral image technique to achieve efficient SAD like window cost calculations. The application of integral image is simple and direct [18].

The size of an integral image is the same as the original image. The definition of every element in the integral image is as follows

$$ii(x,y) = \sum_{x' \leqslant x, y' \leqslant y} i(x',y'), \tag{1}$$

where $i(x, y)$ is the original image and $ii(x, y)$ is its corresponding integral image. Fig. 1 shows the idea of integral image.

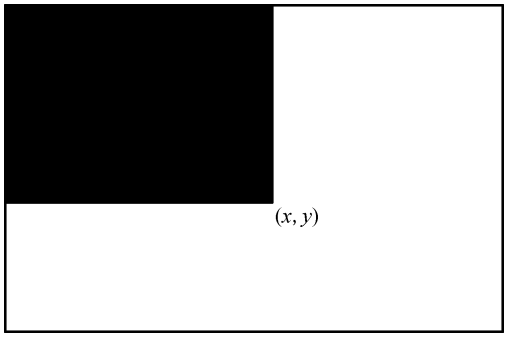

Fig. 1 Element at (x, y) of integral image is the sum of all the elements in the black area of the original image

An integral image can be calculated in an iterative manner efficiently from the upper left corner to the lower right corner as follows

$$ii(x,y) = i(x,y) + ii(x-1,y) + ii(x,y-1) - ii(x-1,y-1). \quad (2)$$

So the computational cost for constructing an integral image is $O(WH)$, where W and H are the width and height of the image, respectively. Using an integral image, the sum of the values in block D, S_D, from the original image, shown in Fig. 2, can be calculated with the following equation

$$S_D = S_{A+B+C+D} - S_{A+B} - S_{A+C} + S_A = ii(x,y) - ii(x,y') - ii(x',y) + ii(x',y'). \quad (3)$$

It is easy to see that the computational cost of the above equation is $O(1)$. That is to say it has no relationship to the size of block D.

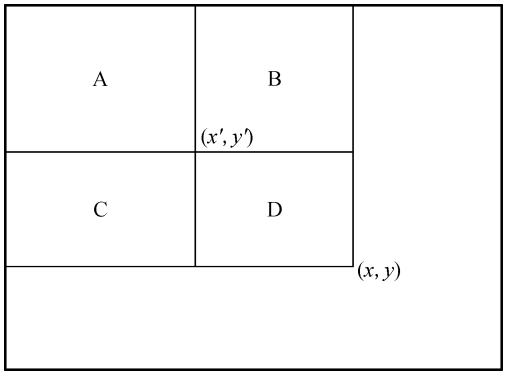

Fig. 2 Demonstration of calculating the sum of values in block D from the original image with its corresponding integral image

2.2 Efficient NCC calculation algorithm

We assume the stereo pair is horizontally aligned. The definition of NCC is

$$C(p,d) = \frac{\sum_{(x,y)\in W_p} \left(I_1(x,y) - \overline{I_1}(p_x,p_y)\right)\left(I_2(x+d,y) - \overline{I_2}(p_x+d,p_y)\right)}{\sqrt{\sum_{(x,y)\in W_p} \left(I_1(x,y) - \overline{I_1}(p_x,p_y)\right)^2 \sum_{(x,y)\in W_p} \left(I_2(x+d,y) - \overline{I_2}(p_x+d,p_y)\right)^2}}, \quad (4)$$

where W_p is the matching window with center pixel p whose coordinate is (p_x, p_y). $\overline{I_1}(x,y)$ is the mean intensity of the pixels in the matching window with center pixel (x, y) in image I_1. $\overline{I_2}(x+d,y)$ is the mean intensity of the pixels in the matching window with center pixel $(x+d, y)$ in image I_2. If we directly apply Eq. (4) to calculate NCC, the computational cost is approximately $O(WHD_r|W_p|)$, where D_r is the disparity search range and $|W_p|$ is the number of pixels in the matching window. From the above analysis we find that the computational cost is greatly affected by the size of the matching window. If the size of the matching window is large, which is the case with a large image, it is very slow to calculate NCC by Eq. (4).

In [18], the author claims that if there exists a function, $f(x, y)$, which converts pixels to real numbers. And if we wish to compute the sum of this function in some rectangular area of the image, which is in the form

$$\sum_{x' \leqslant i \leqslant x, y' \leqslant j \leqslant y} f(i,j), \quad (5)$$

it can be calculated efficiently by the integral image technique. It is obvious that Eq. (4) is not of the same form as Eq. (5). Neither can Eq. (4) be directly divided into subparts so that each part is in the form of Eq. (5), since every term contains the mean value computed in a rectangular neighborhood which is the same size as the correlation window and centered on each pixel. We have to find other ways to calculate NCC efficiently.

Obviously, the following two equations hold

$$\overline{I_1}(p_x,p_y) = \frac{1}{|W_p|} \sum_{(x,y)\in W_p} I_1(x,y), \quad (6)$$

$$\overline{I_2}(p_x+d,p_y) = \frac{1}{|W_p|} \sum_{(x,y)\in W_p} I_2(x+d,y). \quad (7)$$

Substituting Eqs. (6) and (7) into Eq. (4) and rearranging, we get

$$C(p,d) = \frac{\sum_{(x,y)\in W_p} I_1(x,y)I_2(x+d,y) - \frac{1}{|W_p|}\sum_{(x,y)\in W_p} I_1(x,y) \sum_{(x,y)\in W_p} I_2(x+d,y)}{\sqrt{\left\{\sum_{(x,y)\in W_p} I_1(x,y)^2 - \frac{1}{|W_p|}\left[\sum_{(x,y)\in W_p} I_1(x,y)\right]^2\right\}\left\{\sum_{(x,y)\in W_p} I_2(x+d,y)^2 - \frac{1}{|W_p|}\left[\sum_{(x,y)\in W_p} I_2(x+d,y)\right]^2\right\}}}. \quad (8)$$

We define

$$F_1(p,d) = \sum_{(x,y)\in W_p} I_1(x,y)I_2(x+d,y), \tag{9}$$

$$F_2(p,d) = \sum_{(x,y)\in W_p} I_2(x+d,y)^2, \tag{10}$$

$$F_3(p,d) = \sum_{(x,y)\in W_p} I_2(x+d,y), \tag{11}$$

$$F_4(p) = \sum_{(x,y)\in W_p} I_1(x,y)^2, \tag{12}$$

$$F_5(p) = \sum_{(x,y)\in W_p} I_1(x,y). \tag{13}$$

Eq. (8) can be rewritten as

$$C(p,d) = \frac{F_1(p,d) - \frac{1}{|W_p|}F_5(p)F_3(p,d)}{\sqrt{\left(F_4(p) - \frac{1}{|W_p|}F_5(p)^2\right)\left(F_2(p,d) - \frac{1}{|W_p|}F_3(p,d)^2\right)}}. \tag{14}$$

Eqs. (9)–(13) are all of the form in Eq. (5). If we fix d, all five equations can be efficiently calculated using the integral image technique. From Eq. (14) we can draw the conclusion that NCC is the static combination of Eqs. (9)–(13) which can be calculated using integral images, so NCC can also be calculated efficiently.

We assume that ii_k is the integral image which corresponds to image I_k. It can be calculated efficiently according to Eq. (2). We define

$$ii_k(W_p) = \sum_{(x,y)\in W_p} I_k(x,y). \tag{15}$$

Fast calculation of Eq. (15) can be achieved by applying Eq. (3). For any horizontally aligned stereo pair, the NCC calculation method based on integral image contains the following three steps:

Step 1: Choose the matching window, W_p, with appropriate size according to the size of image pair.

Step 2: Calculate ii_1 and ii_2 according to Eq. (2). We define images I_{11} and I_{22} as follows:

$$I_{11}(x,y) = I_1(x,y) \times I_1(x,y) \text{ and}$$
$$I_{22}(x,y) = I_2(x,y) \times I_2(x,y).$$

From the above two images, we calculate their corresponding integral images, ii_{11} and ii_{22}.

Step 3: Let $D_{\min}$ and $D_{\max}$ be the minimum and maximum disparity search levels respectively. Without losing generality, we temporarily only consider the integer disparity level so that we have

$$D_r = D_{\max} - D_{\min} + 1. \tag{16}$$

For every disparity search level, d, which satisfies $D_{\min} \leqslant d \leqslant D_{\max}$, we define image I_{12} as follows:

$$I_{12}(x,y) = I_1(x,y) \times I_2(x+d,y).$$

Then we calculate the corresponding integral image, ii_{12}. Finally, we calculate the NCC cost for every pixel, p, in the image at the disparity search level d by the following equation:

$$C(p,d) = \frac{ii_{12}(W_p) - \frac{1}{|W_p|}ii_1(W_p)ii_2(W_{p+d})}{\sqrt{\left[ii_{11}(W_p) - \frac{1}{|W_p|}ii_1(W_p)^2\right]\left[ii_{22}(W_{p+d}) - \frac{1}{|W_p|}ii_2(W_{p+d})^2\right]}}, \tag{17}$$

where W_{p+d} is the matching window around pixel $p + d$ whose coordinate is $(p_x + d, p_y)$.

Let us analyze the computational complexity of our algorithm. In step 2, calculation of ii_1 and ii_2 is direct and from Section 2.1 we can conclude that the time complexity is $O(WH)$ for each of them. From the definitions of I_{11} and I_{22}, it is obvious that the computational complexity for obtaining any of them is still $O(WH)$. To obtain their corresponding integral images, the extra computational cost for each is $O(WH)$. So the total computational cost of

step 2 is

$$O(WH)+O(WH)+(O(WH)+O(WH))+(O(WH)+O(WH))=O(WH). \quad (18)$$

In step 3, when d is fixed, I_{12} can be implemented in $O(WH)$. By applying the technique described in Section 2.1, for every pixel in the image, Eq. (17) can be calculated in constant time irrespective of the size of the matching window $|W_p|$. So with I_{12}, NCC cost for the entire image with any fixed disparity, d, is $O(WH)$. The total time complexity for any fixed disparity, d, in step 3 is

$$O(WH)+O(WH)=O(WH).$$

Since we have to visit all possible disparity levels in the disparity search range, the total computational cost of step 3 is $O(WHD_r)$. Therefore the computational cost of our proposed algorithm is a combination of step 2 and 3

$$O(WH)+O(WHD_r)=O(WHD_r). \quad (19)$$

Recall that the computational cost of the simple NCC calculation method is $O(WHD_r|W_p|)$ which is much higher than our method.

3 Experimental results

All the experiments are performed on a PC with a Pentuim4 2.8 GHz CPU and 1 GB RAM. All the source code is compiled in Visual C++ 6.0 and only optimized at the algorithmic level. We do not use any CPU optimization techniques.

3.1 Computational Cost Analysis

In this section, we compare the computational time for performing NCC, between our proposed algorithm and simple NCC calculation methods using Eq. (4) directly. The experiments are performed on the same image pair for both algorithms. Fig. 3 shows the results.

In the first experiment, we choose an image pair with resolution 640×480. The disparity search range is 11 to 100; so there are 90 disparity levels. We vary the matching window sizes from 3×3 to 25×25 and record only the time consumed by the NCC calculation. From Fig. 3(a) we find that the computational time of our method is an almost constant 3 seconds for all window sizes. The simple NCC calculation method, which directly applies Eq. (4), uses much more time especially when the matching window size is large. Its computational cost grows almost quadratically with the edge size of the matching window. Fig. 3(a) agrees with our theoretical

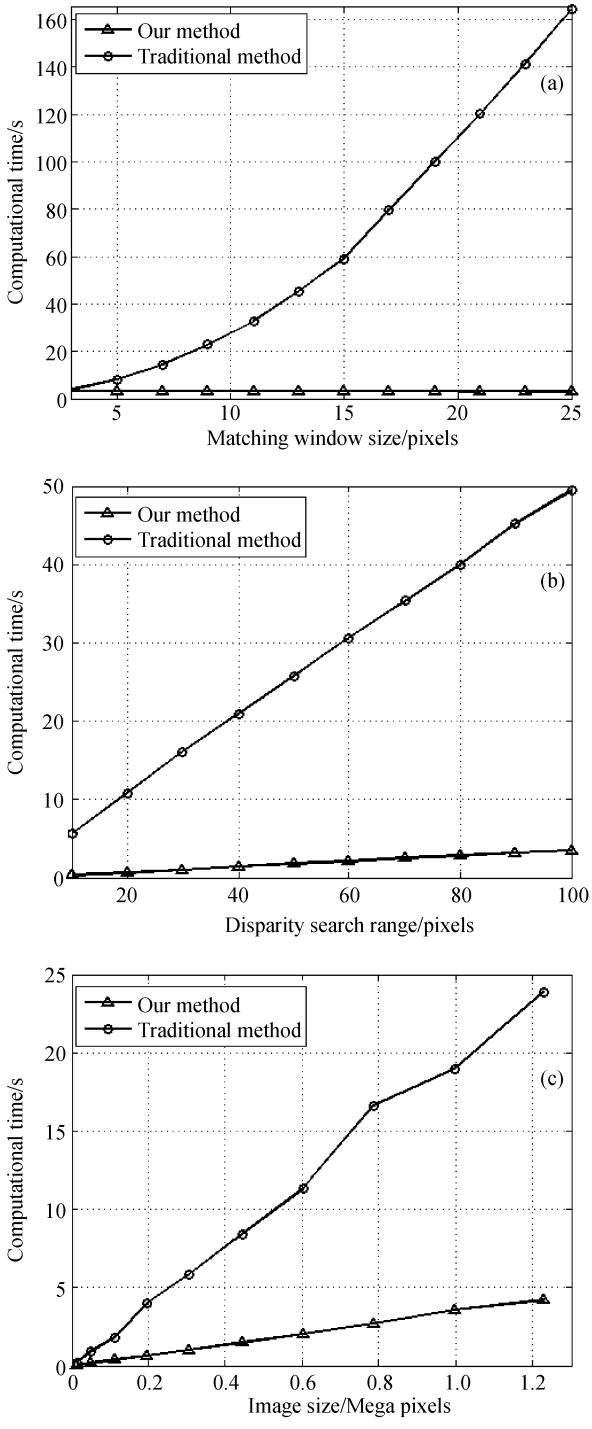

Fig. 3 Comparison of computational costs between our proposed algorithm and the simple NCC calculation method for (a) different matching window sizes; (b) different disparity search ranges; (c) different image sizes

analysis in Section 2.2. Although in our experiments we use square matching windows. Any rectangular shape can be used for real applications, and the computational cost of our method will still not change with the size of matching window.

In the second experiment, we again choose an image pair with the resolution 640×480. We fix the size of the matching window to be 17×17. We vary the size of disparity search ranges from 10 to 100 levels and record only the time consumed by NCC calculation. From Fig. 3(b) we know that the computational cost of both methods grows linearly with the size of disparity search range. But our method grows much slower than the simple NCC calculation method.

In the third experiment, we choose the size of the matching window to be 7×7. We also fix the disparity search range to be 30 levels. We vary the image pair resolution between 128 × 96 and 1280×960, and record only the time consumed by NCC calculation. From Fig. 3(c) we can see that the computational cost of both methods grows approximately linearly with the size of the image pair. But our method again grows much slower than simple NCC calculation method.

The results from the above 3 experiments confirm our theoretical analysis in Section 2; the computational complexity of our algorithm is $O(WHD_r)$. This is much lower than simple NCC calculation method.

3.2 Comparison between NCC and SAD

SAD is widely used in real-time stereo matching algorithms. So in this section we compare the computational complexity of SAD and NCC. The stereo matching algorithm first performs SAD or NCC, recording the time cost, then a simple winner-take-all method [3] is used to generate the final disparity images.

We compare stereo image pairs from a lunar terrain simulation laboratory. The resolution of the image is 640×480. The size of the matching window is 17×17. The disparity search range is from 11 to 100 so that there are 90 disparity levels. The second stereo pair is downloaded from [19] which was taken by navigation cameras of the NASA's MER Spirit on Mars. The resolution of the image is 512×512. The size of the matching window is 13×13. The disparity search range is from 17 to 80, so that there are 64 disparity levels.

Table 1 shows the computation time for SAD and NCC (calculated by our algorithm) in the above 2 stereo pairs. SAD is calculated directly by its definition equation and is not optimized. From the table we can see that SAD takes less time but this is comparable to NCC. Considering that by applying NCC we can obtain a much better disparity image under complex illumination conditions, there is great potential for using the NCC calculation method proposed in this paper for planet rover navigation.

Table 1 Comparison of computational times between SAD and NCC

stereo pair index	calculation times for SAD/s	calculation times for NCC/s
1	1.406	3.156
2	0.906	1.828

3.3 Combination with sophisticated stereo matching algorithms

In the previous section, a winner-take-all method is used for generating disparity images for comparison purposes. It is a very simple method. Usually in practical applications, more sophisticated stereo matching algorithms are needed. Matching cost is still an important part for all the stereo matching algorithms. So our efficient NCC calculation method can be incorporated into nearly all stereo matching algorithms. Our method greatly accelerates the computational speed of the whole stereo matching algorithm without sacrificing performance since we can directly generate the same results using Eq. (4).

In [20], the authors propose a stereo matching algorithm based on Ground Control Points (GCPs). They choose NCC as the matching metric. We implemented their algorithm using our efficient NCC calculation method, which we call combination GCP + ENCC. As mentioned in the previous section, global correspondence methods usually provide a better disparity image. We compare the disparity images generated by GCP + ENCC with the classical dynamic programming algorithm (DP) and Graph Cuts algorithm (GC) [21] which are currently among the top performing stereo matching algorithms [3]. We also compare the computational cost. The DP algorithm is implemented by the author of [3] and GC algorithm is provided by the author of [21].

The stereo pair in Fig. 4 is captured in a lunar terrain simulation laboratory. The resolutions of the images are 640×480. The disparity search range is from 11 to 100 so that there are 90 disparity levels for all 3 algorithms. The size of the matching window for GCP + ENCC is 17×17. The stereo pair in Fig. 5 was taken by navigation cameras of Spirit on mars. The resolution of the images is 512×512. The disparity search range is from 1 to 60 so

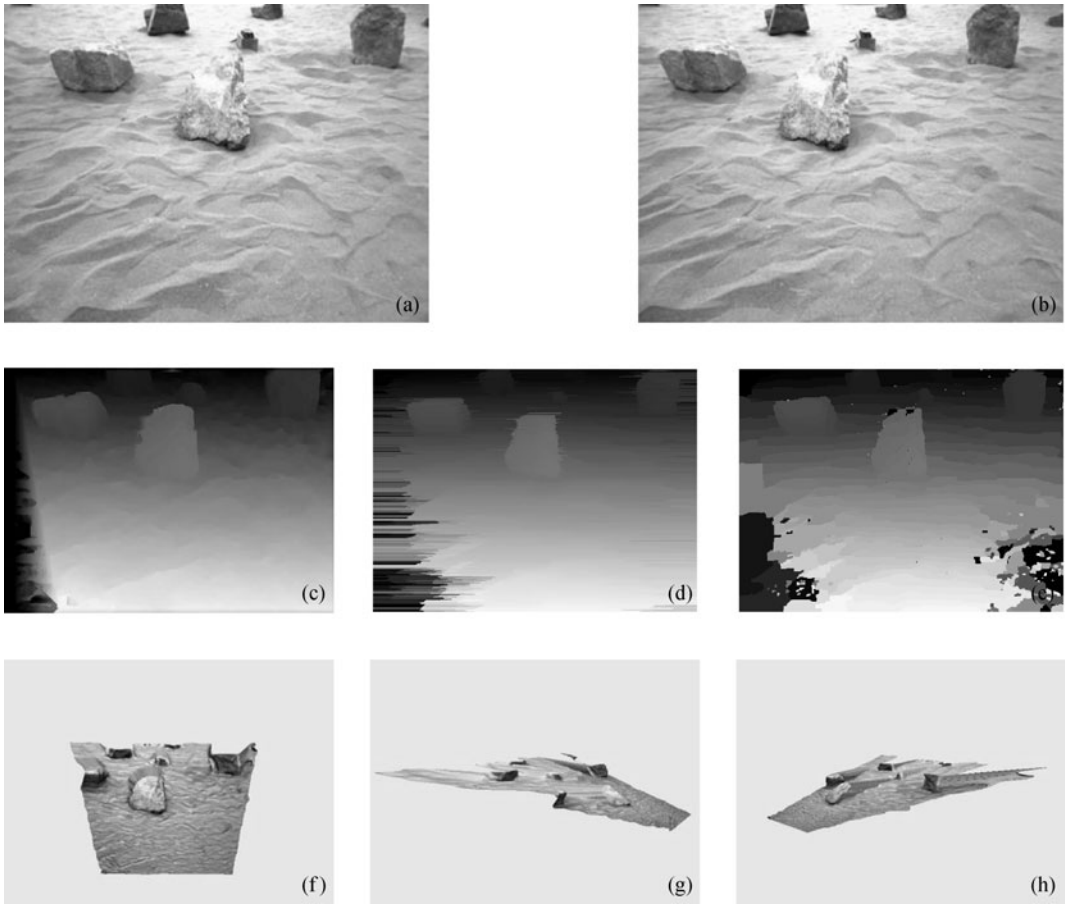

Fig. 4 3rd stereo pair. (a) Left image; (b) right image; (c) disparity image of GCP + ENCC; (d) disparity image of DP; (e) disparity image of GC; (f)–(h) 3D reconstruction views from the disparity image (c)

that there are 60 disparity levels for all 3 algorithms. The size of the matching window for GCP + ENCC is 13×13. From the above two figures we can see that GCP + ENCC produces comparable disparity images to the DP method and is even slightly better than the GC algorithm. The 3D reconstruction results based on disparity images produced by GCP + ENCC are vivid and can be used for navigation and scientific research purposes. From Table 2 we know that GCP + ENCC is the fastest of the three techniques, thanks to our efficient NCC calculation method. Our proposed method provides potential for future use in stereo vision based autonomous robot navigation.

4 Conclusion and future works

In this paper, we propose an efficient NCC cost calculation method based on integral images for stereo matching. The computational complexity does not grow with the size of the template window so that it is much faster than traditional NCC calculation method. Experimental results show that our method greatly accelerates the computational speed of the whole stereo matching algorithm without sacrificing performances. It offers great potential for future autonomous robot navigation applications.

As future work, we plan to incorporate the proposed efficient NCC method into a real-time application. From Fig. 3(c) we can see that the computational time is about 40 ms for an image resolution 128×96 with disparity search range of 30 pixels. So it is possible to implement a real-time stereo matching algorithm with the image pyramid technique. Furthermore, a disparity search range selection method based on application distance range and stereo camera pair calibration results must be incorporated in order to achieve a compact disparity search range and reduce the computational complexity. Last but not least, code level optimization techniques such as multimedia extensions (MMX), streaming

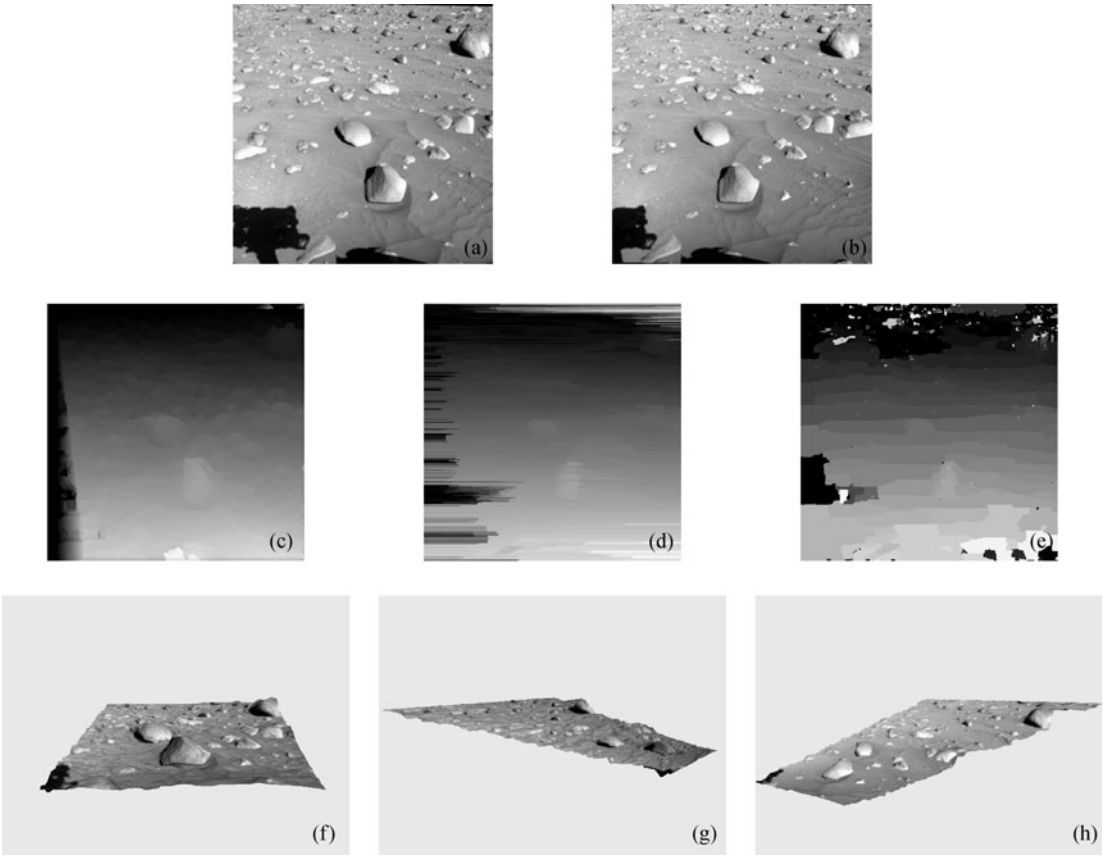

Fig. 5 4th stereo pair. (a) Left image; (b) right image; (c) disparity images of GCP + ENCC; (d) disparity image of DP; (e) disparity image of GC; (f–h) 3D reconstruction views from the disparity image in (c)

Table 2 Comparison of computational costs for GCP + ENCC, DP and GC

Stereo pair index	Calculation times/s		
	GCP + ENCC	DP	GC
3	5.266	8.406	234.190
4	4.094	5.125	180.000

single instruction multiple data (SIMD) extensions (SSE) etc. also need to be studied for real-time purposes.

References

1. Urmson C, Anhalt J, Bagnell D, et al. Autonomous driving in urban environments: boss and the urban challenge. Journal of Field Robotics, 2008, 25(8): 425–466
2. Thrun S, Montemerlo M, Dahlkamp H, et al. Stanley, the robot that won the DARPA grand challenge. Journal of Field Robotics, 2006, 23(9): 661–692
3. Scharstein D, Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision, 2002, 47(1/3): 7–42
4. Brown M, Burschka D, Hager G. Advances in computational stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003, 25(8): 993–1008
5. Bellutta P, Manduchi R, Matthies L, et al. Terrain perception for DEMO III. In: Proceedings of 2000 IEEE Intelligent Vehicles Symposium. 2000, 326–331
6. Goldberg S, Maimone M, Matthies L. Stereo vision and rover navigation software for planetary exploration. In: Proceedings of 2002 IEEE Aerospace Conference. 2002, 2025–2036
7. Zabih R, Woodfill J. Non-parametric local transforms for computing visual correspondence. In: Proceedings of 3rd European Conference on Computer Vision. 1994, 151–158
8. Point Grey Triclops. http://www.ptgrey.com/
9. Small Vision System. http://www.videredesign.com/
10. Matthies L, Maimone M, Johnson A, et al. Computer vision on Mars. International Journal of Computer Vision, 2007, 75(1): 67–92
11. Woodfill J, Gordon G, Jurasek D, et al. The Tyzx DeepSea G2 vision system, a taskable, embedded stereo camera. In: Proceedings

of the IEEE Computer Society Workshop on Embedded Computer Vision, Conference on Computer Vision and Pattern Recognition. 2006, 126–132

12. Faugeras O, Hotz B, Mahieu H, et al. Real time correlation-based stereo: algorithm, implementations and applications. Technical report, INRIA, Sophia-Antipolis, 1993
13. Lewis J. Fast normalized cross-correlation. Online, 1995, Available: http://www.idiom.com/~zilla/Work/nvisionInterface/nip.pdf
14. Hii A J, Hann C E, Chase J G, et al. Fast normalized cross correlation for motion tracking using basis functions. Computer Methods and Programs in Biomedicine, 2006, 82(2): 144–156
15. Briechle K, Hanebeck U. Template matching using fast normalized cross correlation. In: Proceedings of the Society for Photo-Instrumentation Engineers, 2001, 4387: 95–102
16. Viola P, Jones M. Robust real-time face detection. International Journal of Computer Vision, 2004, 57(2): 137–154
17. Crow F. Summed-area tables for texture mapping. In: Proceedings of 11th Annual Conference on Computer Graphics and Interactive Techniques. 1984, 207–212
18. Veksler O. Fast variable window for stereo correspondence using integral images. In: Proceedings of 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2003, 556–561
19. MER Analyst's Notebook. http://anserver1.eprsl.wustl.edu/
20. Chen Q, Medioni G. Building 3-D human face models from two photographs. Journal of VLSI Signal Processing, 2001, 27(1/2): 127–140
21. Kolmogorov V, Zabih R. Computing visual correspondence with occlusions using Graph Cuts. In: Proceedings of 8th IEEE International Conference on Computer Vision. 2001, 508–515