A Comparative Evaluation of Leading Dense Stereo Vision Algorithms using OpenCV

Dr. Rachna Verma
Department of Computer Science and Engineering
J.N.V. University, Jodhpur, Rajasthan, India
rachna_mbm@yahoo.co.in

Dr. A. K. Verma
Department of Production and Industrial Engineering
J.N.V. University, Jodhpur, Rajasthan, India
arvindrachna@yahoo.com

***Abstract*- Stereo vision is an active research area of machine vision aiming to develop human like vision abilities in robots and other machines. During the last decade several stereo vision algorithms for computing dense disparity map have been developed aiming at improving the accuracy and computation speed of the algorithms. Currently, Graph Cut, Semi Global Block Matching and Block Matching are the most researched algorithms. This paper presents a comparative evaluation of these algorithms for their accuracy and speed on standard test bed images and images captured by a prototype stereo vision setup developed by the first author.**

Keywords- Machine Vision, Stereo Vision, Disparity map, OpenCV

I. Introduction

Stereo vision is a key step to estimate depth for automatic navigation of mobile robots, 3D reconstruction of the real world from images, terrain mapping and many other applications in engineering and medicine where human vision like capabilities are required. It has been established that the depth can be recovered from two (or more) images of the scene taken from slightly different known viewpoints using principle of triangulation. The depth of various scene points may be recovered if the disparity of the corresponding points can be calculated from rectified stereo images. In rectified stereo images the corresponding points will always lay on the same horizontal scan line. The disparities of all image points of a pair of stereo image is called disparity map. A disparity map is a 2D image where the intensity of each pixel represents the distance of that point to the camera. Light pixels are near to the camera and dark pixels are farther from the camera. The depth *z* of a point in the scene is calculated using equation (1) that uses principal of triangulation.

$$z = \frac{bf}{(x_l - x_r)} \qquad (1)$$

Where, *b* is the baseline distance (difference between the position of two cameras), *f* is the focal length of the camera, *z* is the depth to be computed and $d = x_l - x_r$ is the calculated disparity.

Stereo correspondence algorithms developed during the last three decades can be broadly divided into local methods and global methods. Local methods use the pixels in a small window surrounding the pixel of interest, whereas global methods use complete scan lines or the entire image. The global methods are found to be more accurate as compared to the local methods, though they are computationally more expensive and fail to meet real time requirements. On the other hand, the local methods are simple and fast but fall short of producing accurate results at object boundaries and in texture less regions. OpenCV is a widely used library written in C/C++ containing functions for computer vision. Presently, Graph Cut (GC), Block Matching (BM) and Semi Global Block Matching (SGBM) are the most researched disparity map generation algorithms and are implemented in OpenCV. GC is a global method, BM is a local method and SGBM is a semi global method that uses both local and global approach.

This paper investigates these three approaches and evaluates their performances in terms of both the computational speed and the quality of the disparity map computed. The results are reported for testbed images as well as for the stereo image captured by the authors by the prototype stereo vision setup developed by the first author.

II. Dense Stereo Methods

The past two decades have produced vast literature on stereo vision which has been critically reviewed by Scharstein and Szelinski [1], Brown *et. al* [2] and Nalpantidis[3]. Wang *et. al.*, [4] evaluated various cost aggregation functions used in local methods for evaluating their suitability for real time implementation. Gong *et. al.*, [5] reported results which are an extension of Wang *et. al.*, [4]. They evaluated only those cost aggregation methods that are suited to real-time implementation on GPU. In this paper, an effort is being made to evaluate the currently most researched and leading stereo methods namely GC, SGBM and BM highlighting the important features of the algorithm which have been implemented in OpenCV library. The standard test bed images and actual stereo images have been used for the same. The brief introduction of these methods is described next.

III. Graph Cut Method

The Graph Cut method is based on energy minimization. Energy minimization algorithms define the best disparity configuration *f* to be the one that minimizes the energy.

Kolmogorov and Zabih [6] addressed the correspondence problem by constructing an energy function *E*. The energy function is of the form

$$E(f) = E_{data}(f) + E_{occ}(f) + E_{smooth}(f).$$

The three terms are

E_{data}, a data term, which represents the differences in intensity between corresponding pixels.

E_{occ}, an occlusion term, which imposes a penalty for making a pixel occluded and

E_{smooth}, a smoothness term, which makes neighboring pixels in the same image tend to have similar disparities.

The data term is given by equation (2)

$$E_{data}(f) = \sum D(a) \qquad (2)$$

Where, $D(a)=(I(p)-I(q))^2$ for an assignment *a* and *I* is the intensity of a pixel.

The occlusion term imposes a penalty C_p if the pixel *p* is occluded. This is given by equation (3)

$$E_{occ}(f) = \sum_{p\epsilon P} C_p . T(\ |N_p(f)| = 0\ ) \qquad (3)$$

Where, *N* is a Neighborhood system for the pixels of the left image and *T(·)* is 1 if its argument is true and 0 otherwise.

The smoothness term is given by equation (4)

$$E_{smooth}(f) - \sum_{\{a1,a2\}=N} V_{a1,a2} . T(f(a1) \neq f(a2)) \qquad (4)$$

The neighbourhood consists only of pairs {a1, a2} such that assignments *a1* and *a2* have the same disparities.

Kolmogorov and Zabih [6] stated that given the energy function one can construct a graph where the labeling with the lowest energy is equal to the minimum cut on that graph. The minimum cut on the graph yields the configuration that minimizes *E*.

IV. Block Matching Method

The block matching, a local method has been the preferred method for computing dense disparity map due to its potential for real time implementation. The method use pixels in a small window surrounding a pixel of interest. The window is selected in the reference image (normally left image) and a number of similar windows are selected in the target image (normally right image) on the corresponding scanline. On the basis of some similarity criterion, the best match of the reference window is searched in the target windows. The corresponding element is given by the window that maximizes the similarity criterion within the search region. The block matching method of OpenCV uses sum of absolute differences (SAD) as the correlation function for matching blocks of pixels. Computer vision community often uses SAD or Sum of Squared differences (SSD) as the correlation function for computing similarity. The SAD is given by equation (5). The local methods generally perform following steps [1], for example using (SAD) the steps are:

Matching cost computation - Matching cost is the absolute difference of intensity values of pixels.

Cost (support) aggregation - Aggregation is done by summing the matching cost over a square window and is given by equation (5)

$$c(x,y,d)=\sum_{i=-m/2}^{m/2}\sum_{j=-m/2}^{m/2}(I_l(x+i,y+j)-I_r(x+i-d,y+j)) \qquad (5)$$

Where, I_l and I_r are the left and right image intensities respectively in grayscale, and *c(x, y, d)* is the matching cost at disparity *d*. The disparity *d* varies between d_{min} and d_{max} and can be selected beforehand based on stereo system parameters and scene structure in the constrained environment to reduce the computational load.

Disparity computation/optimization – Disparity is computed by selecting the minimum aggregated value in the disparity range using equation (6)

$$D(x,y) = \min_{d\epsilon N}(c(x,y,d)) \qquad (6)$$

Where, *N* represents the set of all possible disparities in range (d_{min}, d_{max}) in the right image. Function *min* returns the disparity value corresponding to the minimum matching cost in the disparity range.

V. Semi Global Block Matching

Semi Global Matching [7] combines the concepts of global and local stereo methods for accurate pixel-wise matching. The algorithm is divided into three parts *Matching Cost computation*, *Cost aggregation* and *Disparity computation*. These are discussed next.

Matching cost computation - There are two ways for computing pixelwise cost, either by using mutual information or the sampling insensitive measure of Birchfielsd and Tomasi [8]. The OpenCV uses the second one. The cost *C(p,D_p)* using [8] is calculated as the absolute minimum difference of intensities at *p* and *q* in the range of half a pixel in each direction along the epipolar line.

Cost aggregation- Pixelwise cost calculation is generally ambiguous, therefore, an additional constraint is added that supports smoothness by penalizing changes of neighbouring disparities. The pixelwise cost and the smoothness constraints are expressed by defining the energy *E*(*D*) that depends on the disparity image *D* as given in equation (7). *P1* and *P2* are applied for small and large disparity changes.

$$E(D) = \sum_{p}(c(p,D_p) + \sum_{q\epsilon N_p} P_1 I(|D_p - D_q| = 1) + \sum_{q\epsilon N_p} P_2 I(|D_p - D_q| > 1)) \qquad (7)$$

where, *E*(*D*) is the energy for disparity image *D*, *p*, *q* represent indices for pixels in the image, N_p is the neighborhood of the pixel *p*, *C(p, D_p)* is the cost of pixel matching with disparity in D_p,

P_1 is the penalty for a change in disparity values of one between neighboring pixels, P_2 is the penalty passed for a change in disparity values greater than one between neighboring pixels and $I[.]$ is the function which returns one if the argument is true and zero otherwise. P_2 is always greater than P_1.

The first term is the sum of all pixel matching costs for the disparity *D*. The second term adds a constant penalty P_1 for all pixels *q* in the neighbourhood N*p* of *p*, for which the disparity changes a little bit (i.e. 1 pixel). The third term adds a larger constant penalty P2 for all larger disparity changes. The lower penalty P2 permits an adaptation to slanted or curved surfaces. The constant penalty for larger changes preserves discontinuities [7].

The problem of stereo matching is now formulated as the finding the disparity image *D* that minimizes equation (7). The cost function is optimized similarly to scanline optimization [1] which can be seen as a subset of dynamic programming. It effectively optimizes the equation in time that is linear to the number of pixels and disparities. The optimization is performed in 1D only.

The novel idea of SGBM is the computation along several paths, symmetrically from all directions through the image as shown in figure 1. Each path carries the information about the cost for reaching a pixel with a certain disparity. The aggregated (smoothed) cost for a pixel *p* and disparity *d* is calculated by summing the costs of all 1D minimum cost paths that end in pixel *p* at disparity *d* (Figure 1). By default in OpenCV, the matching algorithm aggregates costs for 5 directions. Number of directions can be set by setting the property *fulldp* of StereoSGBM to true to force the algorithm to aggregate costs for 8 directions.

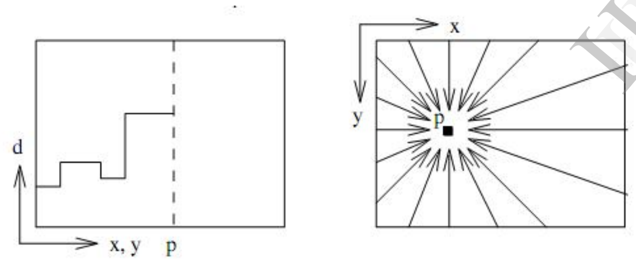


Fig. 1. Aggregation of costs[7]

Disparity computation/optimization – The disparity map is computed as in any local method by selecting for each pixel, the disparity that corresponds to the minimum cost.

VI. Experimental Performance Evaluation and Comparison

This section reports the performance evaluation and comparison of the above described methods implemented in OpenCV. All the methods are tested on testbed rectified stereo images (*Tsukuba*, *Venus*, *Teddy*, and *Cones*) provided by Vision.middlebury.edu/stereo [9]. These images are widely used by stereo vision community to test the performance of the algorithms. All experiments are performed on Intel Core i3 CPU, 2.27 GHz, 3 GB RAM. Figure 2 shows the left image of each set. The disparity range used for Tsukuba is 16 pixels, 32 pixels for Venus and 64 pixels for Teddy, and Cones. The image in the last row of figure 2 is captured by a prototype stereo camera experimental setup developed by the first author. The stereo camera calibration and image rectification program are also developed in Matlab. Before evaluation, all the captured images are rectified using the above program. The disparity range used for this image is 64 pixels.

The two criteria used for the evaluation are accuracy and computational cost. Computational cost is assessed by measuring for each method the execution time needed to process a stereo pair on the same machine. The computation time for BM method comes out to be the minimum among the three methods and graph cut methods takes the maximum.

Figure 2 shows disparity maps generated by each method. Black pixels represent invalid disparities. It can be visually seen from figure 2 that from the three methods evaluated Graph Cut generates a sharp, clear and accurate disparity map for all the images. The Graph Cut also produces much better disparity values especially near object borders. However the complexity of the algorithm is much higher. In terms of speed this method is much slower than the other two and is not suitable for real time applications. Also, the method's computation time depends on number of disparity levels. As the disparity level increases the computation time also increases.

The SGBM and BM methods have potential to compute disparity map in real time on small images. The disparity map generated by SGBM and BM are noisy and the depth discontinuity boundaries are not preserved quite well. However, the disparity map generated by SGBM performs better than BM in terms of accuracy. The results of SGBM are shown in third column of figure 2, using the default parameters for the set of all images. The runtime of the method is more as compared to BM. The last column of figure shows the result of BM. The disparity maps generated on all set of image are little bit more noisy than SGBM. However in terms of speed this method takes least time among the three methods. Table 1 gives the computation time of the three methods for all the testbed image pair and figure 3 shows the plot of computation time in seconds of three stereo methods. From the plot it is clear that GC is very expensive in computation time as compared to the other two methods.

VII. Conclusions

The paper evaluates the performance of Graph Cut, Semi Global Block Matching and Block Matching for computing dense disparity using OpenCV on testbed images and actual stereo image. It is concluded that the disparity map computed by GC method is more accurate than the other two methods however in its present form it is not suitable for real time applications. A near real time performance on small images could be achieved by SGBM and BM methods. The BM method generates disparity map in less than a second for all the testbed and self captured image. However, the disparity maps are noisy. Hence, further research is required to improve computational speed of GC and accuracy of SGBM and BM. Future work includes implementing and testing other methods in OpenCV for computing disparity map.

REFERENCES

[1] Scharstein D. and Szeliski R., “A Taxonomy and Evaluation of Dense Two Frame Stereo Correspondence Algorithms”, *Int. Journal of Computer Vision*, Vol.47, pp. 7 - 42, 2002.

[2] Brown M. Z., Burschka D. and Gregory D. Hager, “Advances in Computational Stereo”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 25, No 8, 2003.

[3] Nalpantidis L., Georgios C.S. and Gasteratos A., “Review of Stereo Vision Algorithms: from Software to Hardware”, *Int. Journal of Optomechatronics*, Vol. 2, pp. 435 - 462, 2008.

[4] Wang L., Gong M., Gong M. and Yang R. ,“How Far can We Go with Local Optimization in Real Time Stereo Matching” Proc. 3D Data Processing, Visualization and Transmission, pp. 129-136, 2006.

[5] Gong M., Yang R., Wang L. and Gong M., “A Performance Study on Different Cost Aggregation Approaches used in Real-Time Stereo Matching”, *Int. Journal of Computer vision*, Vol. 75, No 2, 2007.

[6] Kolmogorov V. and Zabih R., “Computing Visual Correspondence with Occlusions using Graph Cuts” *Proc. Int. Conf. on Computer Vision (ICCV)* , pp. 508 - 515, 2001.

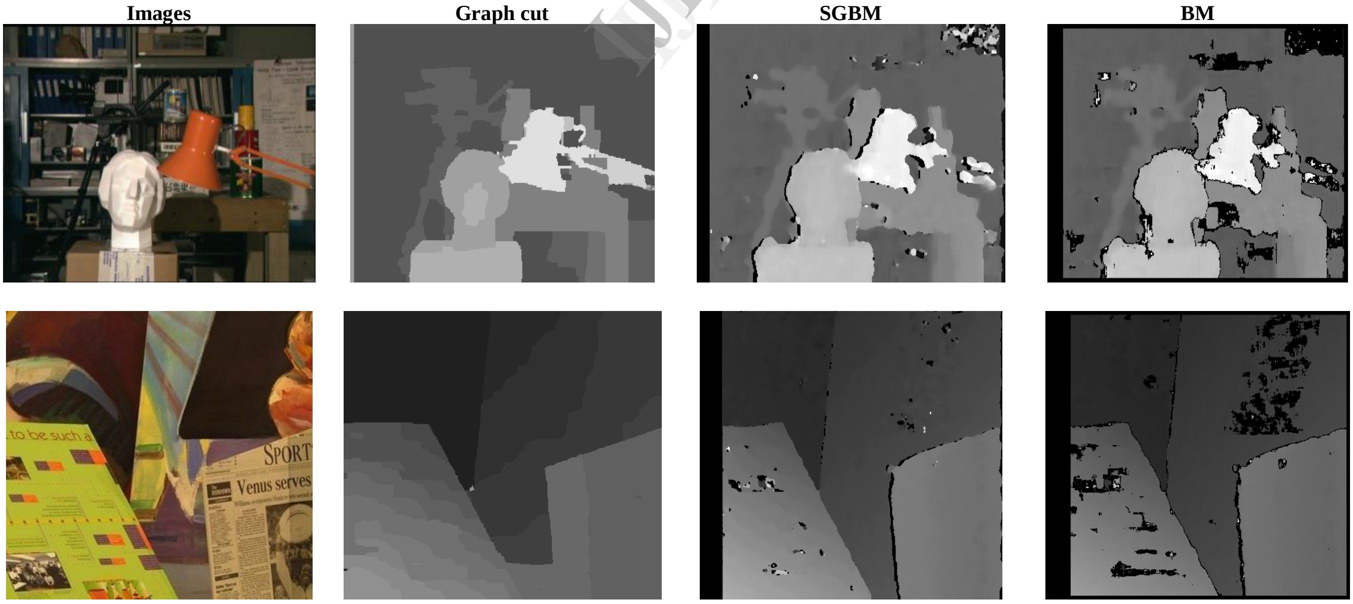
[7] Hirschmuller H, “Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information”, IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, pp 807-814, 2005.

[8] Birchfield S., Tomasi C., “A pixel dissimilarity measure that is insensitive to image sampling”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 20, No. 4, pp. 401 - 406 , 1998.

[9] http://www.middlebury.edu/stereo accessed on 15 Dec, 2013

TABLE 1: COMPUTATION TIME OF THE THREE METHODS IN SECONDS.

Computation Time (in seconds)					
Methods	**Tsukuba (288 x 384)**	**Venus (383 x 434)**	**Teddy (375 x 450)**	**Cones (375 x 450)**	**Self-captured image (396 x 555)**
Graph Cut	7.769	19.530	67.567	63.889	105.238
StereoSGBM	0.970	1.357	1.803	1.859	2.083
StereoBM	0.603	0.733	0.685	0.777	0.814



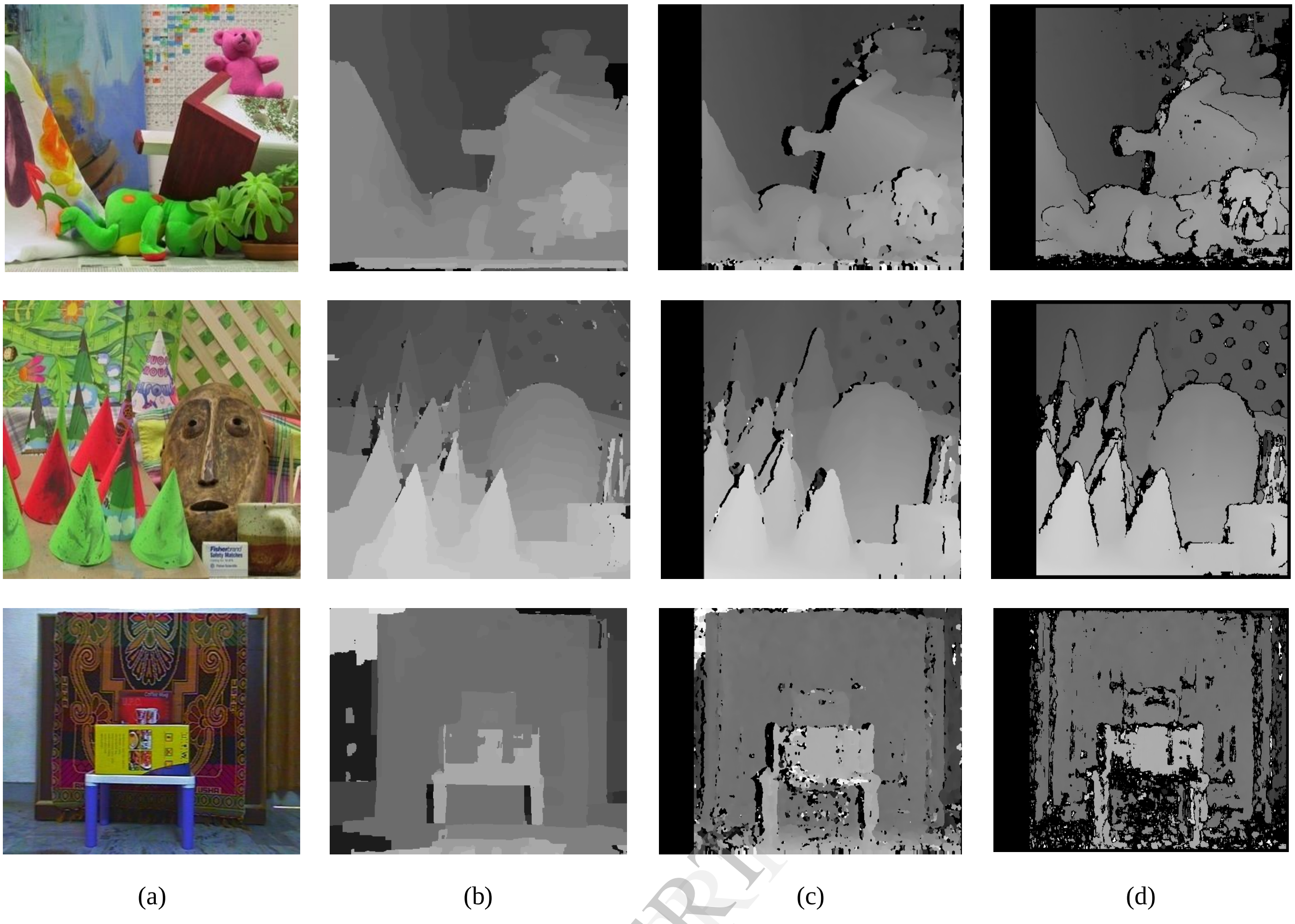
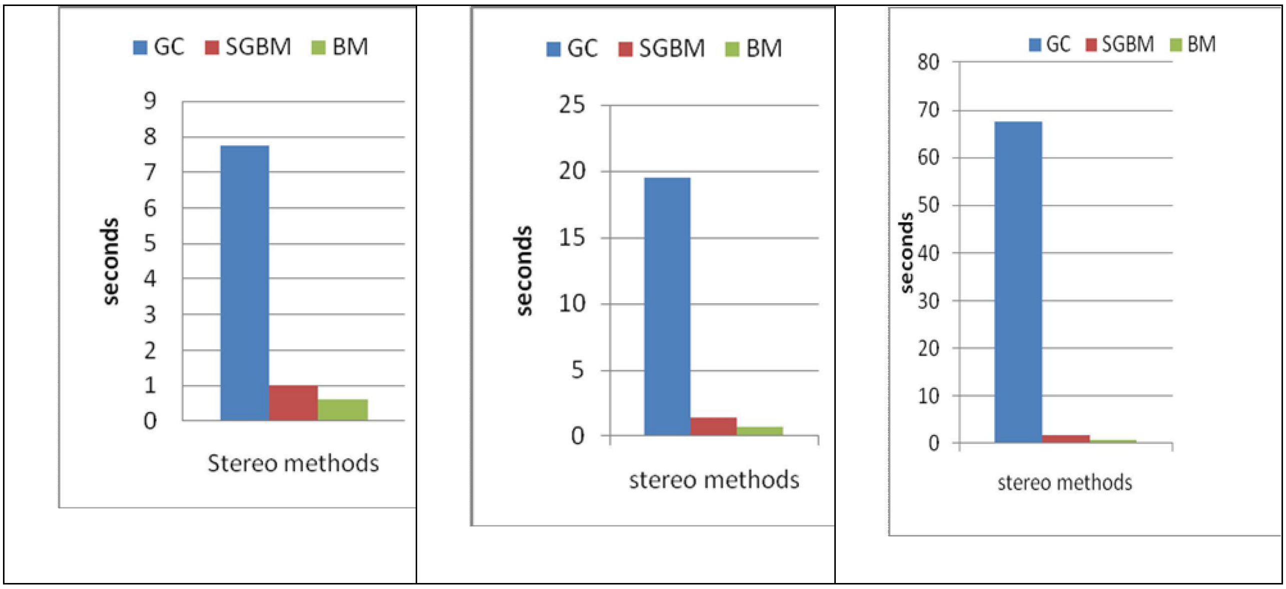
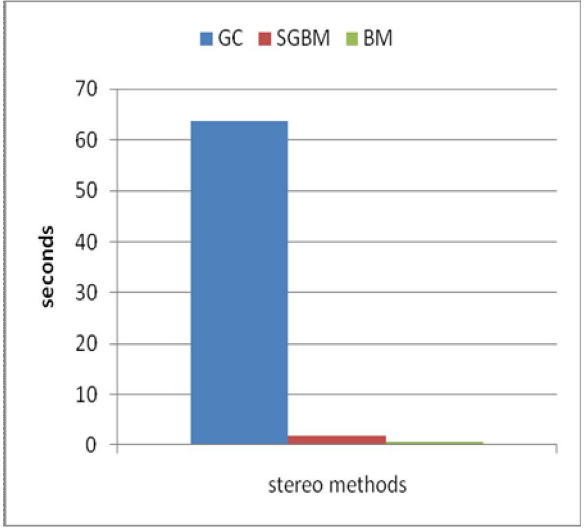
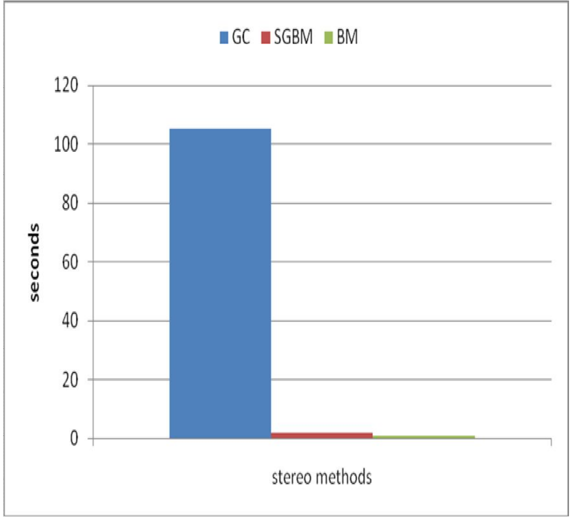


Fig. 2. Comparision of stereo Methods (a) left Image (from top to bottom Tsukuba, Venus, Teddy , Cones and self- captured image). (b) Graph cut (c) SGBM (d) BM



<table>
<tr><td>Tsukuba
(288 x 384)</td><td colspan="2">Venus
(383 x 434)</td><td>Teddy
(375 x 450)</td></tr>
<tr><td colspan="2">
</td><td colspan="2">
</td></tr>
<tr><td colspan="2">Cones
(375 x 450)</td><td colspan="2">Self-captured image
(396 x 555)</td></tr>
<tr><td colspan="4">Fig. 3. Plot of computation time (in seconds) of three stereo methods</td></tr>
</table>