Low Cost Stereo Vision with Webcams

Kathy Kok
Department of Electrical and Computer Engineering
University of Canterbury
Christchurch, New Zealand
Email: kwk17@uclive.ac.nz

Richard Green
Department of Computer Science and Software Engineering
University of Canterbury
Christchruch, New Zealand
Email: richard.green@canterbury.ac.nz

Abstract—**Stereo vision allows for the retrieval of depth data from images taken by cameras from multiple perspectives. This depth information is valuable in computer vision as it can enable better segmentation via augmenting the raw image data with a third dimension. In many applications however, it is desirable to avoid costly stereo cameras and dedicated hardware by instead making use of inexpensive webcams and implementing stereo correspondence algorithms on a general purpose PC. In the project described in this paper, experiments are conducted with various methods of stereo matching using images obtained from both the benchmark Middlebury data sets [1] and taken from off-the-shelf webcams. The experimental results are then presented which evidence the expected trade-off of performance versus accuracy. It is found that while the Graph Cut algorithm produces a disparity map close to the ground truth image, it does not achieve a frame rate suitable for real-time applications. Of the other two algorithms evaluated, Block Matching performs the best in terms of speed, with 14.7 fps achieved for 320x240 images, when the images are captured from the webcams. Lastly, it is found that the modified Semi Global Block Matching algorithm, while only achieving about a quarter the frame rate of Block Matching, results in a superior disparity map which is comparable to that produced by the Graph Cut method.**

Keywords-stereo correspondence; calibration; rectification; webcams; graph cut; block matching; semi global block matching

I. INTRODUCTION

In this project a stereo vision system utilising relatively low-cost webcams is developed. We make use of off-the-shelf webcams in order to meet the objective of building a stereo vision system that is as inexpensive as possible, as commercial stereo cameras can cost thousands of dollars. The system incorporates two horizontally aligned Logitech Pro 9000 cameras and is implemented on CPU hardware.

Obtaining a 3D reconstruction from stereo images involves several steps. Firstly, the cameras must be calibrated by estimating their intrinsic and extrinsic parameters. The former describe properties of the camera including the focal length and distortion introduced by the lens, while the latter describe the transformation from the real-world coordinate system to the camera coordinate system. Calibration also identifies the spatial relation between the two cameras, commonly referred to as the epipolar geometry. Secondly, rectification of the captured images is performed using the knowledge of the epipolar geometry, which limits the space that must be searched when attempting to match corresponding pixels between the images. Thirdly, a disparity map is generated using a stereo correspondence algorithm. The ideal algorithm provides accurate yet fast matching of image points. From a disparity map, a 3D reconstruction can be generated.

For camera calibration and rectification, the methods of Zhang [2] and Bouguet [3] respectively are used. In this project, those correspondence algorithms available in the OpenCV 2.1.0 library [4] are considered. Specifically, the Block Matching [5], Graph Gut [6] and Semi Global Block Matching [7] algorithms are evaluated. We perform experiments to investigate both the performance and accuracy of each of these algorithms. These three stereo correspondence algorithms were first analysed using the Tsukuba stereo images from the Middlebury data set [1]. Because the Graph Cut algorithm's execution time was in the order of several seconds, it was not considered further for the development of the stereo vision system which aims to execute in near real-time.

The final developed application is able to use either Semi Global Block Matching or Block Matching. Stereo image pairs are captured from the two Logitech cameras and processed according to the specified correspondence method. For a real-world application requiring a near real-time response, the Block Matching option would need to be chosen. However, depending on the application, Semi Global Block Matching may be appropriate due to its ability to generate cleaner disparity maps than the Block Matching algorithm and perform at speeds much faster than the Graph Cut method.

II. BACKGROUND

A. Camera Calibration

The first step to obtaining a disparity map is to calibrate the two webcams by computing the intrinsic and extrinsic parameters. The intrinsic parameters are independent of what is viewed and therefore need only be calculated once as long as the zoom level remains unchanged. They describe the focal lengths and, importantly, the radial and tangential distortion introduced by imperfections in the camera lens. Particularly in cheap webcams, the radial distortion can be significant and thus the accurate identification of these parameters is critical in being able to compensate for it. Extrinsic parameter estimates can be reutilised as long as the camera positions remain the same relative to each other.

Two commonly used approaches for camera calibration are those of Zhang [2] and Tsai [8]. According to [9] Tsai's

algorithm requires difficult issues to be overcome, such as requiring high measurement accuracy of the calibration pattern and difficulties with finding calibration points. Thus it is not used here. A complete description of the pinhole camera model with radial and tangential distortion can be found in [2].

Camera calibration using Zhang's method involves capturing multiple images of a planar calibration pattern at various orientations. Feature points on the calibration pattern are identified and the camera parameters are estimated by combining a closed form solution and linear least squares. The parameters are then refined using maximum likelihood estimation. The degenerate case which occurs when images are pure translations of each other can easily be avoided by ensuring that the various pattern orientations comprise a rotational component as well as a translational component.

A measure of the quality of calibration is the reprojection error and is given by the "Euclidean distance between the identified marker point and the projection of its corresponding pattern point onto the image." [10] Zhang's method has the disadvantage that the entire calibration pattern must be in view of the camera in all shots. A suggested solution to the problem is the use of ARTag fiducial markers [10] within a calibration pattern however the potential of this to obtain a lower reprojection error and thus potentially a better disparity map is not investigated in this paper.

B. Epipolar Geometry

Rectification is a process by which a transform is applied so that the image planes are coplanar and matching points lie on the same epipolar line. If information about the epipolar geometry has been obtained in the prior camera calibration step, Bouget's rectification method can utilise the calibration data to generate images which have been corrected for both camera misalignment and intrinsics such as lens distortion [3].

In Fig.1 unrectified image pairs are shown on the left and rectified image pairs are shown on the right. In the rectified image, the matching points M1 and M2 are on the same epipolar line. After rectification, all epipolar lines are parallel and horizontal. The purpose of rectification in the context of gathering depth information is to limit the search space for finding matching pixels. This clearly helps to speed up a stereo matching algorithm as searching only needs to occur along the epipolar line that the relevant pixel lies on.

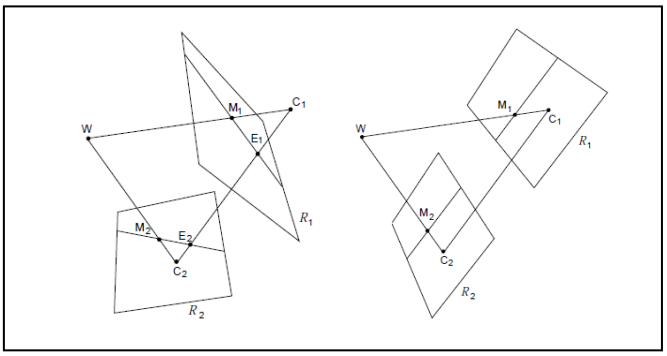

Figure 1. Visualisation of unrectified (left) and rectified (right) stereo image pairs [11]

C. Stereo Matching

Depth data is obtained from stereo images by applying a stereo correspondence algorithm. The output of these algorithms is generally a disparity map, in which the value at each pixel indicates the difference in position between the matched points in the two images [12]. The depth of an object is able to be seen because the closer it is to the cameras, the greater the disparity between the positions of matched pixels. Three methods are considered in the development of the stereo vision system described in this paper – Graph Cuts, Block Matching and Semi Global Block Matching.

Graph Cut stereo correspondence algorithms aim to find matching pixels between two images by minimising some global energy function. However finding a minimum for an energy function is usually computationally intensive to solve. The Graph Cut method developed by Kolmogorov et al [6] can produce highly accurate disparity maps but is the slowest of the three algorithms which are used in this project and are provided by the OpenCV library. With this algorithm, the authors address two issues with prior Graph Cut methods, namely the issue of handling occlusions and the issue that pixels should be matched to at most one pixel in the other image. The way this is achieved is through cross-checking pixels and marking them as occlusions in cases where the left-to-right and right-to-left matches do not correspond. In [6] it can be seen that the disparity map generated is very close to the ground truth image for the Tsukuba dataset. The running time is given to be in the order of a minute for a 384 x 288 image, which is clearly too slow for real-time applications.

Another algorithm that is available in the OpenCV library is the Block Matching stereo correspondence algorithm by Konolige. It is a much faster algorithm than Kolmogorov's Graph Cut algorithm however the accuracy of the output disparity map is much reduced to achieve this gain in speed. In applying Block Matching to a rectified stereo pair, the correlation along epipolar lines using the sum of absolute differences (SAD) is computed. Where a minimum SAD is found, this is taken to be where the matching points occur.

An improvement that can be made over Block Matching in terms of disparity map quality is to use the Semi-Global Block Matching algorithm. This method is not able to achieve as good a frame rate as the Block Matching algorithm but a more accurate disparity map is generated when compared with Block Matching. It is implemented in OpenCV as proposed by Hirshmuller [7] with a few modifications. These modifications include matching blocks rather than individual pixels, so that the SAD window size is in general greater than one. Additionally, there are pre- and post-processing steps including a Sobel pre-filter and post-filters such as a uniqueness check and speckle filtering, which come from Konolige's Block Matching algorithm [4].

The execution times of these algorithms increase with the image size. Additionally, it is possible to specify the disparity range, which will limit both how close and how far an object can be from the cameras and still have its depth computed. That is, objects whose matching points have a disparity greater that the specified maximum disparity will not be accurately detected. However, limiting the disparity range will increase

performance because fewer computations are required with a smaller number of disparities.

III. METHODOLOGY

The OpenCV computer vision library was used to implement the code for this stereo vision system. All of the functions called from this library are for CPU implementation. The two webcams were aligned horizontally. The procedure for generating a sequence of disparity maps from stereo pairs captured from webcams involved firstly calibrating the cameras with a chessboard pattern, secondly rectifying the images using the calibration data and finally calculating the depth by finding matching pixels.

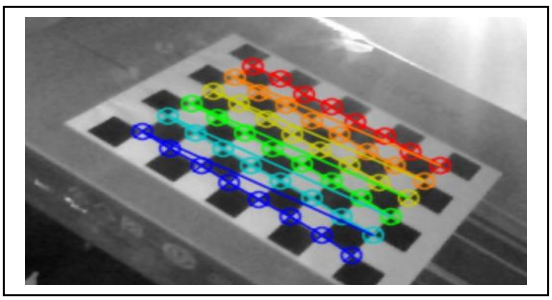

Figure 2. Corner detection using cvFindChessboardGuesses for providing feature points for intrinsic and extrinisic camera paramer estimation

The OpenCV function cvFindChessboardGuesses() finds correspondences automatically from images of a chessboard calibration pattern and the output from this can be used as input into Zhang's Algorithm, implemented as cvCalibrateCamera() and cvCalibrateCamera_64d() in OpenCV. We therefore captured twenty images of a planar chessboard pattern and calculated the calibration data from the identifiable points. Often the calibration can be poor and therefore it can be a lengthy process to obtain an accurate set of parameters.

From the calculated intrinsic and extrinsic parameters, images were rectified using the stereoRectify(), initUndistortRectifyMap() and remap() OpenCV functions. The first two of these compute the rectification transformations and thus only need to be called once. remap() takes the transformation map and applies it to the stereo pair.

The relevant OpenCV stereo matching functions are cvFindStereoCorrespondenceGC(), StereoBM() and StereoSGBM(), which implement the Graph Cut, Block Matching and Semi Global Block Matching algorithms respectively. The final implemented program only allows for the graph cut algorithm to be able to be executed on static images in this project whereas the block matching and semi global block matching algorithms are able to be run on images captured from the two webcams.

IV. RESULTS

A. *Middlebury Data Set*

We have executed the three aforementioned algorithms on a 32-bit machine with 2GB of RAM and a 2.2GHz AMD V120 single core processor. Firstly, we do a comparison using the benchmark Middlebury stereo image data sets , which are the standard basis for comparison when evaluating stereo correspondence algorithms and for which ground truth disparity maps are provided.

In Fig 3. the Tsukuba image and associated ground truth are shown. Disparity maps are generated for the Tsukuba image pair using each of the three methods. Given that we have the ground truth image, we are able to evaluate the outputs by calculating the percentage of pixels for which the difference in disparity from the ground truth is greater than one pixel. Additionally, we measure the running time of each method over a number of runs and compare the results.

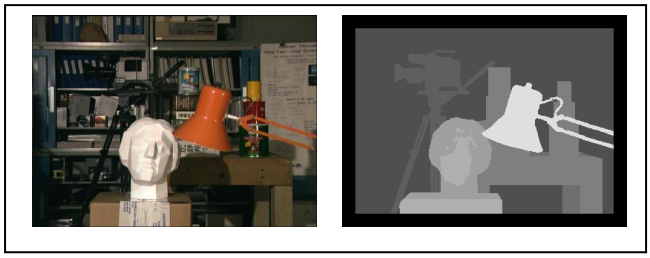

Figure 3. Original image and its ground truth disparity map from the Tsukuba Middlebury data set

Figs 4, 5 and 6 show the disparity maps generated for the Graph Cut, Block Matching and Semi Global Block Matching algorithms respectively. It can be seen that of the three, the Graph Cut method has produced the disparity map that is the closest to the ground truth. Block Matching produces the least accurate disparity map, while Semi Global Block Matching produces a relatively clean image, but is not as accurate as the Graph Cut method.

Figure 4. Disparity map for the Tsukuba data set using Graph Cut Stereo Correspondence algorithm

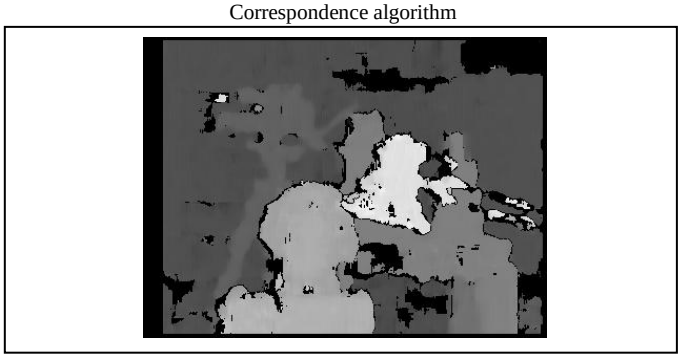

Figure 5. Disparity map for the Tsukuba data set using Block Matching stereo correspondence algorithm

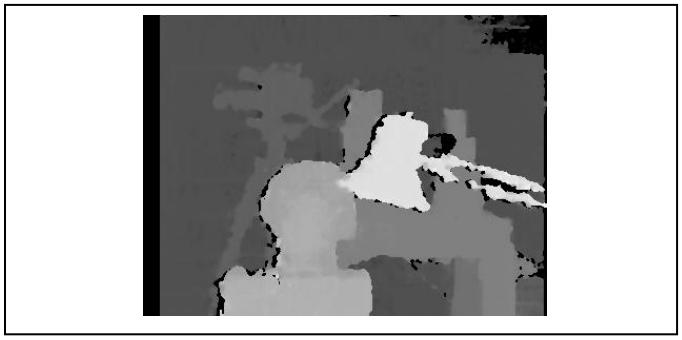

Figure 6. Disparity map for the Tsukuba data set using Semi Global Block Matching stereo correspondence algorithm

Table I shows the execution times for each of the stereo matching algorithms when used to process the Tsukuba stereo images. The image size is 384 x 288 pixels.

TABLE I. STEREO MATCHING ALGORITHM EXECUTION TIMES FOR THE TSUKUBA DATA SET

Num Disparities	Execution Time (ms)		
	Graph Cut	***Block Matching***	***Semi Global Block Matching***
16	7200	91.5	450.5
32		124.7	630.2
64		157.5	911.1

Because the Graph Cut method proved to be the slowest, taking 7.2 seconds to execute for an image size of 384x288 with a disparity range of 16, it was not included in any other experiments conducted for this project. This is because it would be impractical for use in applications requiring a response with a reasonably small lag.

TABLE II. PERCENTAGE OF PIXELS GREATER THAN 1 PIXEL DIFFERENCE IN DISPARITY FROM THE GROUND TRUTH

Num Disparities	Percentage of pixels with incorrect disparity		
	Graph Cut	***Block Matching***	***Semi Global Block Matching***
16	8.5%	26.9%	15.7%

Clearly block matching runs much faster than Semi Global Block Matching. However the percentage of pixels in the disparity map where the difference from the ground truth is greater than one pixel is almost twice that of Semi-Global Block Matching, as shown in Table II. Semi Global Block Matching is therefore more accurate than Block Matching but not as accurate as Graph Cuts.

B. Real-world Results

Image sequences captured from the webcams are able to be processed using both Block Matching and Semi Global Block Matching using the system developed in this project. The disparity maps generated for one image pair are shown in Figs. 8 and 9. Both disparity maps show the typical lack of depth information that occurs where there are plain surfaces in the raw images. In order to achieve the fastest execution times possible, the disparity range was limited to 16, which is the minimum allowed value. Limiting the maximum disparity to 16 causes the disparity map to become practically meaningless for objects closer than about 1m in this setup, however this limitation was accepted in order to achieve better speeds.

Figure 7. One image from the original real-world image pair captured from webcams

It can be seen that the disparity map generated by Semi Global Block Matching is much cleaner than that generated by Block Matching. For this particular image, the stereo matching algorithm execution time is 47ms on the AMD V120 processor. Additionally there is typically 8ms taken to rectify the image pair and this is a process that is done at each iteration. The one-off transformation computation typically takes 40ms. These captured images are of size 320 by 240.

The number of frames per second that was achieved is shown in Table III. Results for several different setups are shown, as it was found that the frame rate decreased when the images were captured and processed immediately, as opposed to being saved and loaded then processed with a stereo correspondence algorithm. As well as running the program on the AMD processor, we also tried it with a 64-bit machine with 4GB of RAM and a 3.00GHz Intel E8400 Core 2 Duo processor. Real-time performance was achieved with this more powerful machine when the images were preloaded.

TABLE III. PROCESSING SPEED FOR STEREO MATCHING ALGORITHMS

Images Captured In Real-Time or Loaded from File	Frames per second	
	Block Matching	***Semi Global Block Matching***
Captured AMD V120	14.73	3.7
Preloaded AMD V120	18.1	5.3
Preloaded Intel E8400	32.0	8.1

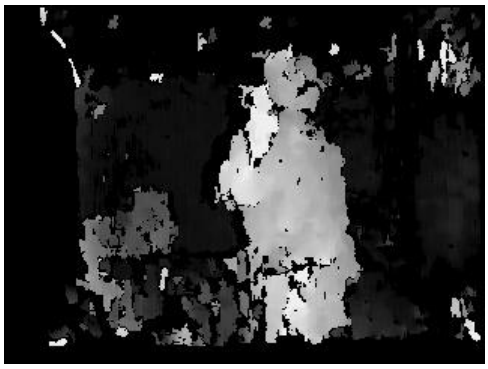

Figure 8. Disparity map using Block Matching

Figure 9. Disparity map using Semi Global Block Matching

V. CONCLUSIONS

Although this webcam stereo vision system works reasonably well given the constraints on the hardware, it does suffer from several limitations. These could prevent the system from being used in practical applications however in simple, necessarily low-cost situations, it could prove to be adequate.

Stereo correspondence algorithms require highly textured surfaces in order to differentiate between pixels and thus provide robust pixel matching between the image pairs, therefore it will not work where there are plain surfaces. This could be overcome by ensuring that the objects of interest in the scene have textured surfaces, although often this would not be possible to do. Also, this may not be a problem in many cases if the plain objects such as walls are not relevant to the application.

In order for the algorithm to execute reasonably quickly, the number of disparities was limited to 16 and therefore the objects in the scene must be a certain distance away from the cameras. The system could be improved by using different webcams. This is because the width of the cameras is such that their lenses must be quite far apart when set down horizontally next to each other. If the lenses were physically closer together then the disparity for close objects would be reduced.

A comparable project described in [13] was able to achieve 14 fps for 640x320 sized images. In that paper, a real-time stereo vision system was developed on the CPU using an optimized semi global block matching approach. This was achieved using parallelization and an intelligent image sub-sampling method. This result is clearly faster than the results obtained here as the images used for testing have almost three times the number of pixels of the images captured from the webcams in this project.

A proposed improvement to Block Matching is suggested in [14] and in that paper it is reported that similar results are often obtained to Konolige's original algorithm and in many cases the processing speeds are faster. In particular, the algorithm proposed in [14] performs better with large disparity ranges. Therefore that algorithm may be appropriate for applications where the depth of objects varies greatly. Otherwise Block Matching, as used in this project, would be the better option because it is shown that it achieves a much higher frame rate with smaller (320x240) images and a small number of disparities.

Lastly, the potential to significantly improve the performance of stereo matching algorithms is shown in [15]. The authors achieved a frame rate of 537.7 fps when implementing their census-based algorithm on a GPU, versus the 62.9 fps they achieved on an Intel Core 2 Duo CPU. It is highly likely therefore that this project could benefit from a GPU implementation.

VI. FUTURE WORK

By repeating the experiments using webcams of various quality, information would be obtained as to how much the resultant disparity map deteriorates with low resolution cameras. Cheaper cameras can be prone to greater radial distortion which manifests itself most furthest from the image centre. If it was found that the quality of the intrinsic parameter estimation was too poor to generate usable depth data, an investigation into the potential of other means of calibration to improve the correspondence could be conducted.

Often it is necessary to recalibrate due to mechanical disturbances but it is not practical to have to continually perform this task manually with a printed calibration pattern. A means of autocalibration for determining the extrinsic parameters could be implemented, which would hopefully allow for camera calibration to be performed using only natural images. This is not a trivial task, as obtaining accurate parameter estimation without using a calibration pattern is difficult. Some self-calibrating techniques have been proposed however in general they do not produce results of as good quality as calibration using a known calibration pattern [16].

Another possible improvement that could be made is to make use of parallelisation and, depending on the application, image subsampling, to achieve high quality depth information with close to real-time performance for Semi Global Block Matching. Alternatively if speed is the key metric to be optimised, and particularly if image resolution and/or the disparity range are necessarily high, the approach of [13] could be implemented to improve the Block Matching algorithm to achieve the required performance.

Finally, we have implemented the stereo correspondence algorithms on a CPU, however a GPU implementation may offer significant performance improvements over the results reported here. The stereo matching algorithms used in this project have not been ported to GPU in OpenCV 2.1 but a GPU implementation of the block matching algorithm is available in OpenCV 2.2.

ACKNOWLEDGMENT

The authors would like to acknowledge the developers of the OpenCV computer vision library, on which the implementation of this project entirely relied. Additionally, the stereo images along with the ground truth images provided by Middlebury were greatly appreciated.

REFERENCES

[1] Middlebury Stereo Vision Page <http://vision.middlebury.edu/stereo/> accessed on 18th April 2011

[2] Zhang, Z. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[3] Bouguet, J. Y. Camera Calibration Toolbox for Matlab, <http://www.vision.caltech.edu/bouguetj/calib_doc/>, accessed on 18th April 2011

[4] OpenCV 2.1 Camera Calibration and 3d Reconstruction Documentation <http://opencv.willowgarage.com/documentation/cpp/calib3d_camera_calibration_and_3d_reconstruction.html> accessed on 18th April 2011

[5] Konolige, K. Small vision systems: Hardware and implementation. *Proceedings of the International Symposium on Robotics Research*, 111-116, 1997

[6] Kolmogorov, V. and Zabih, R. Computing visual correspondence with occlusions using graph cuts. *Eighth International Conference on Computer Vision*, 508–515, 2001

[7] Hirshmuller, H. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30 (2): 328-341, 2008

[8] Tsai., R. Y, A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses, *IEEE Journal of Robotics and Automation*, 3: 323-344, 1987

[9] Tapper, M., McKerrow, P. J., and Abrantes, J., Problems Encoountered in the Implementation of Tsai's Algorithm for Camera Calibration, *Proc. 2002 Australian Conference on Robotics and Automation*, 2002

[10] Fiala, M. and Shu, C. Fully Automatic Camera Calibration Using Self-Identifying Calibration Targets, *NRC Publications Archive (NPArC)*, 2005

[11] Fusiello A., Trucco E. and Verri A. A compact algorithm for rectification of stereo pairs. *Machine Vision and Application*, 12:16-22, 2000

[12] Nalpantidis, L., Sirakoulis, G. C. and Gasteratos, A., Review of stereo matching algorithms for 3D vision, *16th International Symposium on Measurement and Control in Robotics ISMCR 2007*, 116-124, 2007

[13] Gegrig, S. K. and Rabe, C., Real-Time Semi Global Matching on the CPU, 2010

[14] Di Stefano, L., Marchionni, M., Mattoccia, S. and Neri, G., A Fast Area-Based Stereo Matching Algorithm,

[15] Humenburger, M., Zinner C., Weber M., Kubinger, W. And Wincze, M., A fast stereo matching algorithm suitable for embedded real-time systems, *Computer Vision and Image Understanding* 114 (2010) 1180-1202

[16] Illowsky, R. and Huet, L., Self-Calibration of a Pair of Webcams for Stereo Vision, unpublished, 2007