# A Fast Dense Stereo Matching Algorithm with an Application to 3D Occupancy Mapping using Quadrocopters

Radouane Ait-Jellal and Andreas Zell

***Abstract*— In this paper, we propose a fast algorithm for computing stereo correspondences and correcting the mismatches. The correspondences are computed using stereo block matching and refined with a depth-aware method. We compute 16 disparities at the same time using SSE instructions. We evaluated our method on the Middlebury benchmark and obtained promosing results for practical realtime applications. The use of SSE instructions allows us to reduce the time needed to process the Tsukuba stereo pair to 8 milliseconds (125 fps) on a Core i5 CPU with 2x3.3 GHz. Our disparity refinement method has corrected 40% of the wrong matches with an additional computational time of 5.2% (0.41ms). The algorithm has been used to build 3D occupancy grid maps from stereo images. We used the datasets provided by the EuRoC Robotic Challenge. The reconstruction was accurate enough to perform realtime safe navigation.**

## I. INTRODUCTION

In binocular stereo, we are given two images captured by a pair of stereo cameras, left camera and right camera, such that one right camera is shifted (along the x-axis) with respect to the other camera with given distance $B$ called baseline. The task is to find for each point of the reference image its corresponding pixel on the target image. In the standard stereo correspondence form, the search for correspondence of a pixel $p(x, y)$ on the reference image is restricted to a 1D search along the horizontal line having the same y-coordinate on the target image. A commonly made assumption in stereo is to consider the intensity being consistent. In other words, it is assumed that the intensity of a pixel on the reference image is equal to the intensity of its corresponding pixel on the target image. Despite being a classical problem of early vision, stereo matching is a difficult problem because of half-occlusions, textureless regions, repetitive patterns, sensor noise and depth discontinuities. Half-occlusions occur when a point is seen only by one camera. Asking to determine the correspondence for half-occluded points is an ill-structured problem. The best thing to do in this case is to clearly identity them. On textureless regions, constraints made upon intensity consistency are not discriminative. There is a need to carefully use information from other parts of the image to enhance the results on these textureless regions. The existence of repetitive patterns could introduce ambiguities. Developing algorithms that try to handle all of the afore-mentioned challenges does lead to impractically slow algorithms. For applications like environment mapping using stereo cameras mounted on quadrocopters, there is a need for fast algorithms. We seek building maps on-board in real-time on these flying robots. We propose in Sec. III of this work a fast dense stereo matching algorithm. We show in Sec. IV that our stereo algorithm allows to build maps which are as accurate as the ones we get by the use of some (much slower) sophisticated stereo algorithms.

R. Ait-Jellal is with the Chair of Cognitive Systems, headed by Prof. A. Zell, Computer Science Department, University of Tübingen, Sand 1, D-72076 Tübingen, Germany {radouane.ait-jellal, andreas.zell@uni-tuebingen.de}

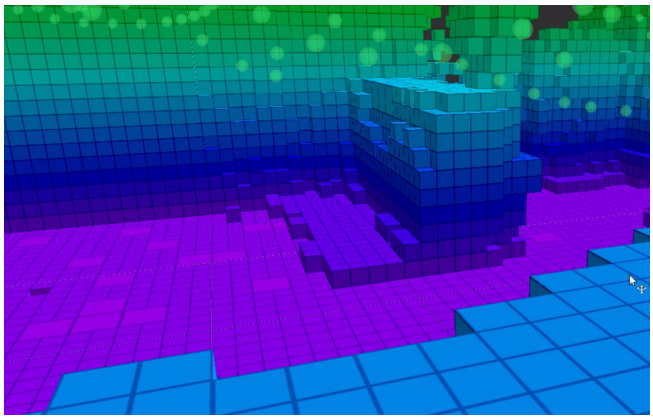


Fig. 1. Reconstruction using our stereo algorithm. The green points show the trajectory of the quadrocopter. The color of the voxels encodes the height z-axis (Best viewed in color).

## II. RELATED WORK

Most stereo algorithms are constructed from a pipeline of the following four blocks [1]: cost function, cost aggregation, disparity optimization and disparity refinement. Stereo algorithms can be divided into two categories: local stereo algorithms and global stereo algorithms [1]. While in local stereo only a local neighborhood of the pixels is used for matching, in global stereo pixels of the whole image contribute to the computation of the correspondence. Although exhaustive work has been done by the computer vision community to propose methods for the first three building blocks, the disparity refinement has been less investigated. This makes most of these algorithms perform poorly in half-occlusions and textureless regions. The very sophisticated refinement methods such as image in-painting [2] and segmentation-driven methods [3] are generally slow. Recent works [4] have shown that using a good refinement method could make the simple local matching algorithms as accurate as some sophisticated global optimization-based stereo algorithms. Refinement methods are of particular importance for local matching algorithms because the estimation of the disparity for each pixel individually leads to the violation of the

smoothness constraint of the objects.

The global optimization based stereo algorithms [5] [6] [7] model the correspondence problem as a Markov random field (MRF) problem. They design an energy function to represent the global cost associated with the selection of a disparity for all of the pixels in the image. The proposed methods differ in the way the energy function is designed. Nevertheless, the energy function for all these algorithms contains a data term and a smoothness term. The data term measures how the selected disparity fits the data. This is done by warping the reference image using as displacement vector map, the candidate disparity map. In the ideal case this will exactly generate an image similar to the target image. In practice we optimize the error between them. The smoothness term insures the coherence of the disparity map. The classical least-squares methods for solving such energy minimization problems are impractical due to the data dimension. Approximating the minimal energy by the use of graph cuts [7] makes global stereo algorithms efficient enough and suitable for some off-line applications.

Local stereo algorithms [4] [8] [9] as opposed to global stereo can be implemented very efficiently. Moreover, they are suitable for parallel computing either using software instructions such as SSE or by using special hardware such as GPUs and FPGAs. It is not always true that local stereo matching algorithms are faster than global matching algorithms. Trying to improve the accuracy of correlation based local stereo, the researchers have proposed different types of cost functions [10] and aggregation methods [11]. The basic block matching algorithm assumes the disparities to be the same within the correlation window [9]. This assumption is violated at the depth discontinuities. Correlation windows of smaller sizes could reduce the errors. Unfortunately, making the window sizes smaller can also significantly decrease the signal to noise ratio. This yields unstable disparity values. [12] proposed the variables correlation windows in which the disparity is not computed for the pixel at the center of the windows as usual but in predefined positions. The disparity associated to the windows in which the lowest cost is found is then selected.[13] combined correlation windows of different sizes. Adapting the size and the shape of the correlation windows individually [8] [9] for each pixel is the way to get the best results, but this could slow down the algorithm extremely [9]. The shape of the correlation windows can be adapted by weighting the contribution of the different pixels to the final aggregated cost. The more the disparity of a neighboring pixel is similar to the disparity of central pixel, the larger the weight should be. Color similarity is a common way for estimating the weights [8]. With regard to the design of cost functions, several methods [10] have been proposed by the computer vision community. These methods include the sum of absolute differences SAD, the sum of squared differences SSD and the normalized cross correlation NCC. Variants of these methods exist like zero-mean sum of absolute differences ZSAD. Compared to NCC and SSD, SAD does require only simple +/- arithmetic operations. As a result it can be computed very efficiently. In our algorithm we have chosen SAD not only because it is efficient but also because it is more robust against noise than the SSD.

There is a third category of stereo algorithms lying between global stereo and local stereo. These hybrid stereo algorithms are known as semi-global matching algorithms SGM. Hirschmller [14] was the first to propose this kind of algorithms. [15] has proposed a variant of SGM. Semi-global matching algorithms are fast but, they are still considerably slower than the efficient block matching algorithms.

## III. THE STEREO ALGORITHM

### A. Initial disparities computation

We compute the correspondences using the sum of absolute differences (SAD) cost function. We aggregate the SAD over a local window of fixed size. In order to keep the algorithm faster we have chosen neither to use multiple correlation windows nor to adapt the window shape and size. To make the system robust against noise we compute the SAD not on the intensities directly but on their derivatives along the x-axis. We use the Sobel operator to compute these derivatives. This way, we build a disparity space image DSI [16]. The DSI stores for each pixel of the reference image the similarity costs on the target image for the different disparities. The input pixels are coded as 8 bit characters and the computation of the costs for the different disparities can be done independently. This allows us to process 16 disparities at the same time using Streaming SIMD Extensions (SSE). The processors supporting SSE are provided with internal registers of 128 bits size. We load the registers with 16 pixels (128 bits) to be processed. Then we unpack the 128 bits register into two 128 bits registers such that each pixel is coded with 16 bits and the 8 high-order bits set to zero. We then send a signal to the processor arithmetic unit to perform the computation. Having the pixels coded in 16 bits is required in order not to overrun the buffer size. The accumulated sums of the absulte differences are stored in 128 bits variables. The use of SSE instructions makes the computation of the cost aggregation task very efficient. Cost aggregation is the most time consuming task of almost any correlation based local algorithm. Thus, making it efficient makes the overall algorithm fast. One of the traditional techniques to speed up the computation for block matching stereo is to use integral images. This makes the computation of the cost independent of the window size and very usefull when we have to deal with large windows. However, a relatively small window of size 5x5 has best fit our needs and we estimate that we do not need to use the integral images. The aggregated cost at the pixel $p(x, y)$ is given by Eq. 1.

$$Cost_{sad}(d) = \sum_{i,j \in W} |\nabla I_1(x+i, y+j) - \nabla I_2(x+i-d, y+j)| \quad (1)$$

Where $W$ is the correlation window, $\nabla I_1(x+i, y+j)$ is the gradient at the position $(x+i, y+j)$ on the first image. We perform the simple winner-takes-all strategy to select the

most likely disparity d. The selected disparity $d^*$ is given by the Eq. 2.

$$d^* = \underset{d}{argmin}(Cost_{sad}(d)) \tag{2}$$

### B. Mismatches detection and correction

We compute two disparity maps, one with the left image as reference image and the second disparity map with the right image as reference image. We then proceed with a left-right consistency check for removing the suspected mismatches. We compare the disparity of a pixel in one disparity map with its re-projection in the other disparity map. If the difference is larger than a given threshold then the pixel is set to be an outlier. This occurs particularly in regions where the pixel is occluded in the second view and in textureless areas. The result is a disparity map, which generally contains a substantial amount of missing disparities (holes). We illustrate the case of half-occlusions in Fig. 2. The scene is composed of three boxes. The orange box lies on the front and occludes some parts of the green box for the left-side camera. These occluded parts are shown in green on the line. The orange box does also occlude some parts of the blue box for the right-side camera. These parts are shown in blue on the line. We use a simple and efficient method for filling the holes. It is obvious that the required action is to fill the holes with the background disparities. The background disparities correspond to the small disparities. In our method we scan the horizontal lines of the disparity map independently, and for each line we first identify the holes. Then we compute for each hole the averaged disparity on a local neighborhood on valid pixels on the left and on the right of the hole. We compare these averaged disparities and decide accordingly which disparities to use to fill the hole. We fit a line on the hole border with smallest disparity and we compute the missing values by extrapolation. The line fitting process is illustrated in the Fig. 2.

The mismatches correction method described above does not require any time consuming preprocessing such as segmentation and plane fitting as proposed by [7]. Our method differs from method [17] used for Depth-Image Based Rendering (DIBR) and 3D Television in the way that we fit horizontal lines to extrapolate the missing values and they used a constant disparity value for the missing pixels. This method has been abandoned by the DIBR community because it can have visual artifacts for which the human eye is very sensitive. They preferably opt for more advanced and accordingly complex and time consuming refinement methods such as image in-painting. We believe that for mobile robots the use of our fast hole filling is reasonably good. Our method differs from the common method based on interpolation in the sense that interpolation methods blur the edges at the depth discontinuities. Despite being simple, intuitive and very efficient it has strongly improved the overall accuracy on the Middlebury benchmark. However, we should notice that this hole filling method can fail when there are wide regions of invalid disparities. In that case, the propagated valid disparities might differ from the correct values. This is why we fill the hole only when the width of the gap is smaller than a threshold. In our experiments we set this threshold to be equal to 1/8 of the image width.

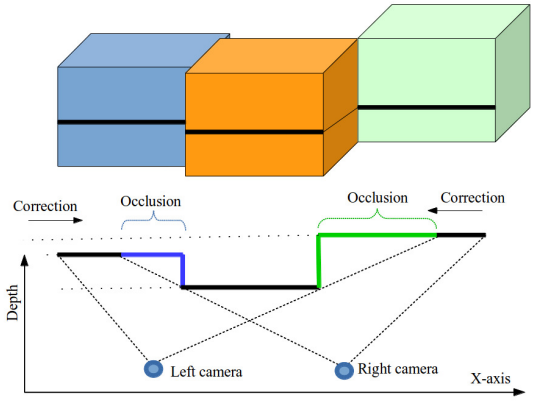



Fig. 2. Occlusion happens at depth discontinuities. The orange box is in the front and occludes some parts of the other boxes. For filling the invalid disparities we perform line fitting process based on valid neighboring disparities and propagate the disparities to the occluded regions.

### C. Edge preserving filter

Generally, objects of the scene are smooth over given areas and can have depth discontinuities near edges. The disparities that we get from the previous steps might contain some unrealistic disparity changes within smooth parts of the object. A common post-processing method to enhance the disparity map is to filter the noise with a smoothing filter. We filter the disparity map with the symmetric nearest neighbor SSN non-linear filter, which makes the disparity map smooth while it does not blur the disparity map at the discontinuities. We found that SSN performs better than the median filter.

## IV. 3D Reconstruction

Building a volumetric occupancy map of the environment is a very important task for mobile robotics. It allows the robot to be aware of its environment and helps the robot to perform a wide range of tasks, such as safe navigation by avoiding obstacles and exploration. Laser range finders are the commonly used sensors for map building because they are very accurate. Unfortunately, the use of laser range finders has many drawbacks. Most laser scanners are too heavy to be carried by Micro-Ariel Vehicles (MAV). They are active sensors with a dedicated light source, requiring additional power for lighting the scene. And due to the mechanical scanning system used for moving the beam they produce scans at lower frame rates than cameras. State of the art laser scanners that attempt to reduce some of these drawbacks are quite expensive. Alternative sensors which do not have the afore-mentioned drawbacks are stereo cameras and RGBD cameras. The Kinect-style RGBD cameras provide good range estimates in indoor environments, but they fail when the amount of infrared light is large and thus are not suitable in sunlight. Stereo cameras are becoming the sensor of choice to fulfill the afore-mentioned

requirements. Nevertheless, most sophisticated stereo algorithms are computationally expensive. This has made the simple block matching algorithms like the one we proposed widely used algorithms in real applications. Even the Kinect sensor computes the depth estimation using a block matching algorithm implemented on FPGA. The rise of small single board computers like the Intel NUC with SSE support and low power consumption (15 Watt) makes the use of stereo algorithm for computing the depth on-board flying robots a reasonably good choice. In the following we describe how we build a 3D map using our stereo algorithm. We have used our stereo algorithm to build volumetric occupancy grid maps from stereo pairs captured by a stereo system mounted on a quadrocopter. We assume that the 6DoF camera poses are known. We use the Robust Octomap which is an improved version of the original Octomap [18]. The Robust Octomap [19] has been designed to specially fit with stereo. The Octomap based map building system models the environment by dividing the volume into voxels organizing these voxels in an Octree structure. The map is updated by integrating the depth measurements (point clouds) in a probabilistic manner. We have intentionally chosen the application of building a map using Octomap because in such applications it is tolerated to use less accurate depth measurements as long as the end point is detected within the correct voxel. This does not apply for the end points which are located near the voxels boundaries. In this case, slight depth errors could lead to errors in the map.

Unlike many stereo algorithms that do not fill the invalid pixels, our algorithm produces dense environment maps by integrating fewer measurements. Those algorithms need to view the scene from other poses in order to get valid disparities for the missing points. To insert measurements from our stereo camera we need to generate point clouds from the disparity maps computed by the stereo algorithm. Since the ambiguity of stereo estimates grows quadratically with respect to the depth, in [19] they proposed a map update method which deals with this problem. We refer to [19] for more details. In order to compute the disparity map from the stereo images we first need to rectify the stereo images. The intrinsic cameras parameters including the cameras matrices and the lens distortion coefficients were estimated using the Kalibr [20] calibration method. Kalibr uses special patterns for calibration. These patterns allow for a more precise estimation of the corner positions and this way provide a more accurate estimation of the cameras parameters when compared to other methods [21]. Based on these parameters we compute the rectification transforms using the OpenCV implementation of the method [22]. To rectify the stereo pairs we remap the input images with the computed rectification transforms. The rectification step transforms the real stereo system to a virtual system in which the cameras are perfectly aligned and have the same focal length. Note that we need to compute the rectification maps only once and reuse them to rectify all the other stereo pairs. The steps for building an occupancy map are as follows:

- Rectify the stereo images to restrict the search for the correspondences from 2D to 1D.
- Perform dense stereo matching using our algorithm described in Sec. III for computing the disparity map.
- Generate point clouds by re-projecting the points of the reference image to 3D.
- Insert the point clouds (with known 6DoF poses) into the global map by probabilistically updating the voxels lying between each 3D point of the point cloud and the camera center.

The reconstruction is evaluated by computing the correlation error of our results with respect to the ground truth. The Matthews correlation coefficient MCC is used a measure for this purpose.

## V. RESULTS

We performed two sets of experiments. First, we tested our dense stereo algorithm using the Middlebury benchmark. Second, we evaluated the reconstruction results using the stereo datasets and the online evaluation tool provided by the EuRoC Challenge (http://www.euroc-project.eu/).

### *A. Evaluation of the dense stereo algorithm*

We used a window size of $5 \times 5$ and set the error threshold to 0.5 on the benchmark table. The average percent of bad pixels on the four stereo pairs (Tsukuba, Venus, Teddy and Cones) is 19.7%. Our algorithm is ranked better than the global optimization based algorithms graph cut [23] and Constant time belief propagation [6]. We get these good results compared to GC and CSBP particularly because the Cones stereo pair includes a substantial amount of occluded pixels. This stereo pair includes large occlusions (See Fig. 3) which were recovered successfully by our refinement method. Like the other algorithms we got the worst results on the Teddy stereo pair.

We have made our experiments on an Intel Core i5 2×3.3 GHz computer with 4GB RAM. We used a correlation window of size 5×5 pixels and a post-processing filter of size 5x5. The run-times of the different steps are shown in Table II. As one can expect when block matching is used, the run-time grows linearly with respect to the input image sizes and disparity ranges. Although our hole filling method does require only about 5% of matching time, it has corrected 40% of the missing disparity values. Reducing this way the average of the bad pixels from 32.70% to 19.70%. The final step of our algorithm is the filtering of the disparity map using an edge preserving filter. To achieve this task we used a median filter with a window size of 5x5. Fig. 4 shows the percentage of bad pixels before and after the refinement. We used a median filter instead of our the symmetric nearest neighbor (SSN) filter because our implementation of the SSN was not optimized and made the overall stereo algorithm slower. However, we found that

Although our algorithm is less accurate than the global optimization based algorithm TSGO and GC+Occ, our algorithm is much faster. While these iterative algorithms perform the matching in several hundreds of milliseconds our

TABLE I

PERFORMANCE EVALUATION ON THE MIDDLEBURY DATASET

| Algorithm | Avg rank | Percentage of bad pixels on all image regions | | | | |
|---|---|---|---|---|---|---|
| | | Tsukuba | Venus | Teddy | Cones | Average all images |
| TSGO [5] | 33.4 | 10.0 | 7.8 | 16.4 | 10.2 | 11.2 |
| AdaptingBP [24] | 46.3 | 9.3 | 5.1 | 16.7 | 13.2 | 13.6 |
| GC+Occ [25] | 94.2 | 7.1 | 11.3 | 30.1 | 19.2 | 17.4 |
| **Our method** | **102.4** | **16.5** | **6.5** | **25.8** | **19.7** | **19.7** |
| GC [23] | 112.5 | 9.8 | 15.0 | 38.5 | 25.5 | 21.6 |
| CSBP [6] | 122.9 | 23.8 | 9.2 | 27.8 | 24.7 | 21.9 |
| RINCensus [26] | 123.8 | 29.5 | 8.5 | 24.5 | 22.5 | 20.0 |

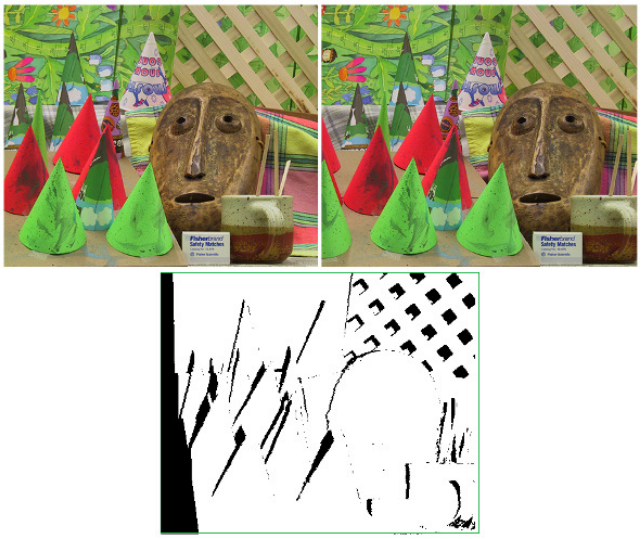


Fig. 3. Cones dataset and its occlusion map.

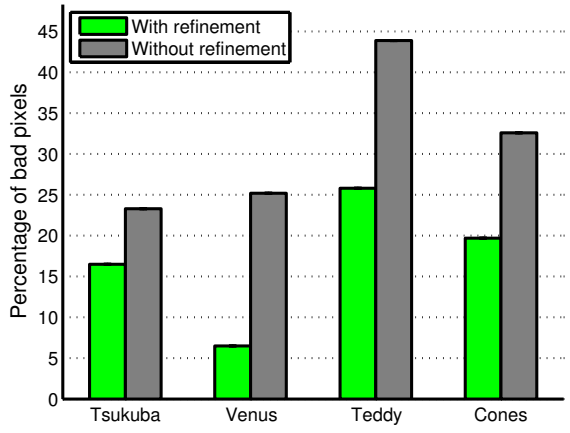



Fig. 4. Percentage of bad pixels before the refinement (grey) and after the refinement (green).

TABLE II

RUNNING TIME IN MILLISECONDS

| Dataset | Match & Consist check | Hole filling | Filtering | Max disp | Total time |
|---|---|---|---|---|---|
| Tsukuba | 5.80 | 0.41 | 1.31 | 16 | 7.52 |
| Venus | 10.19 | 0.64 | 2.41 | 32 | 13.24 |
| Teddy | 15.50 | 0.76 | 1.86 | 64 | 18.12 |
| Cones | 14.98 | 1.27 | 2.12 | 64 | 18.37 |

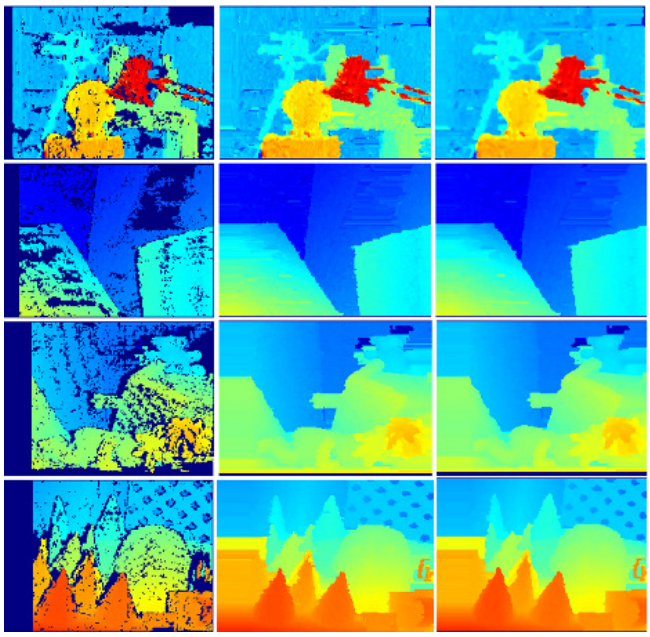


Fig. 5. Our results on the Middlebury benchmark. The left column shows the disparity maps before filling the holes. The second column shows the results after filling the missing disparities. The last column shows the disparity maps after filtering with the median filter.

algorithm performs the matching in less than 20 milliseconds for any stereo pair of the Benchmark. The Fig. 6 shows the comparison of the running times between our algorithm and the global stereo algorithms GC and CSBP and the Semi-global stereo algorithm ELAS [15]. The running times for our algorithm, ELAS and CSBP were reported for an Intel Core i5 with 2×3.3 GHz CPU. An OpenCL implementation of the CSBP was used. The running time for GC is reported for an Intel Pentium 4 with 2 GHz. Please notice that the y-axis has a logarithmic scale.

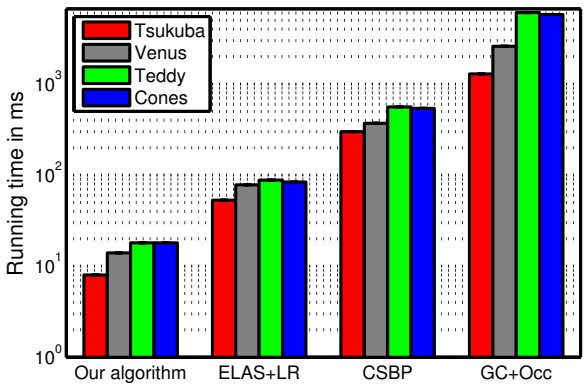
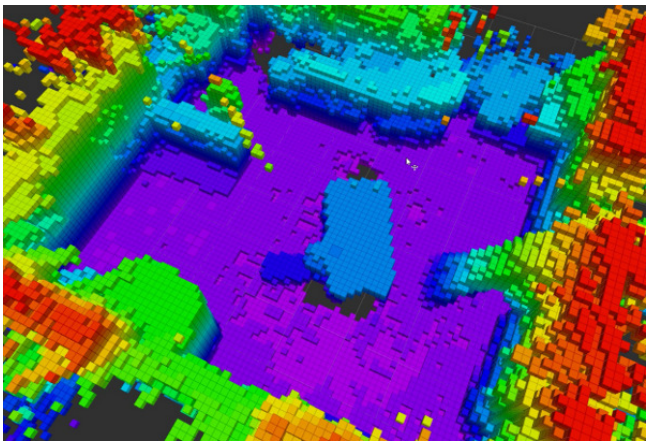
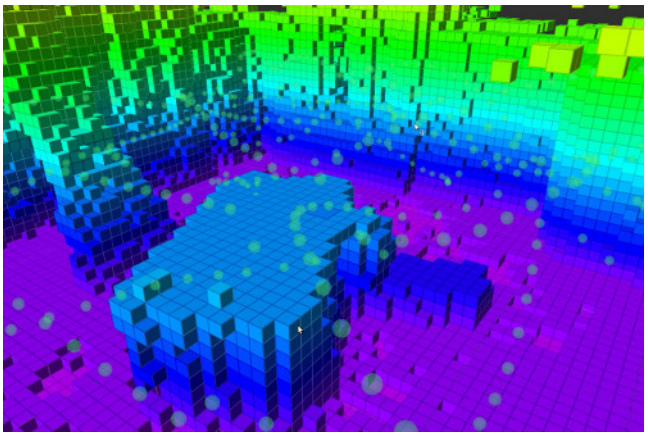



Fig. 6. Running time comparison. The y-axis has a logarithmic scale.

Fig. 7. Snapshots of the scene to be reconstructed



(a)



(b)

Fig. 8. Reconstruction using our stereo algorithm. The green points on (b) show the trajectory of the quadrocopter.

### B. Evaluation of the 3D reconstruction

We used the stereo datasets provided by the EuRoC challenge. These stereo datasets are captured using a quadrocopter. EuRoC provides three stereo datasets of the same scene with an increasing degree of difficulty. EuRoC provides the true 6DoF poses for all the stereo pairs. We used the robot operating system (ROS) to make this experiment. The accuracy of the reconstruction is evaluated using the Matthews correlation coefficient MCC. We speed-up the reconstruction by integrating new measurements only if the difference between the current pose and the previous poses has reached a given threshold. As expected, we got the best accuracy (MCC=0.81) for the dataset 1. We have done the reconstruction using the ELAS SGM algorithm [15] and obtained maps which are comparable to the ones that we have obtained with our algorithm. We believe that this is due to the fact that our algorithm is accurate enough to find in most cases the voxel which contains the end point of the ray. Fig. 7 shows some pictures of the scene to be reconstructed. We visualize the reconstructed Octomap using ROS-Rviz. Fig. 8 shows some screen-shots of the final map reconstructed using our stereo algorithm. The color encodes the height (z-axis). The violet color corresponds to lowest altitude and the red color corresponds to highest altitude.

## VI. CONCLUSION

We proposed a fast stereo algorithm which fits nicely for the task of mapping using quadrocopters. The key advantage of our algorithm is its efficiency. It is suitable to use it when real-time stereo processing is required on a medium level embeded PC on a MAV.

## REFERENCES

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *IJCV*, vol. 47, no. 1-3, pp. 7–42, 2002.
[2] L. Wang, J. Jin, R. Yang, and M. Gong, "Stereoscopic inpainting: Joint color and depth completion from stereo image," *CVPR*, 2008.
[3] D. Lima1, V. Giovani, A. Victorino1, and J. Ferreira, "A disparity map refinement to enhance weakly-textured urban environment data," *ICAR*, 2013.
[4] Z. Ma, K. He, Y. Wei, J. Sun, and E. Wu, "Constant time weighted median filtering for stereo matching and beyond," *ICCV*, 2013.
[5] M. Mozerov and J. van Weijer, "Accurate stereo matching by two step global optimization," *TIP*, 2014.
[6] Q. Yang, W. L., and N. Ahuja, "A constant-space belief propagation algorithm for stereo matching," *CVPR*, 2010.
[7] M. Bleyer and M. Gelautz, "A layered stereo algorithm using image segmentation and global visibility constraints," *Asian Conf. on Comput. Vis ACCV*, pp. 25–38, 2011.

[8] K.-J. Yoon and I.-S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE TPAMI*, vol. 28, no. 4, pp. 650–656, 2006.

[9] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," *TPAMI*, vol. 16, no. 9, pp. 920–932, 1994.

[10] H. Hirschmueller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE TPAMI*, vol. 31, no. 9, pp. 1582–1599, 2009.

[11] M. Dongbo, L. Jiangbo, and N.-D. Minh, "A revisit to cost aggregation in stereo matching: How far can we reduce its computational redundancy," *ICCV*, 2011.

[12] O. Veksler, "Fast variable window for stereo correspondence using integral images," *CVPR*, pp. 556–561, 2003.

[13] S. Adhyapak, N. Kehtarnavaz, and M. Nadin, "Stereo matching via selective multiple windows," *Journal of Electronic Imaging*, vol. 16, no. 1, 2007.

[14] H. Hirschmueller, "Stereo processing by semi-global matching and mutual information," *IEEE TPAMI*, vol. 30, no. 2, pp. 328–341, 2008.

[15] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," *Asian Conf. on Comput. Vis ACCV*, pp. 25–38, 2010.

[16] S.-S. Intille and A.-F. Bobick, "Disparity-space images and large occlusion stereo," *ICCV*, 2013.

[17] P. Lai-Man, Z. Shihang, X. Xuyuan, and Z. Yuesheng, "A new multi-directional extrapolation hole-filling method for dibr," *ICIP*, 2011.

[18] K. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and B. W., "Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems," *ICRA*, 2010.

[19] K. Schauwecker and A. Zell, "Robust and efficient volumetric occupancy mapping with an application to stereo vision," *ICRA*, pp. 6102–6107, 2014.

[20] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems IROS*, 2013.

[21] Z. Zhang, "A flexible new technique for camera calibration," *IEEE TPAMI*, vol. 22, no. 11, pp. 1330–1334, 2000.

[22] R. Hartley, "Theory and practice of projective rectification," *IJCV*, vol. 25, no. 2, pp. 115–127, 1999.

[23] R. Boykov, Veksler, and Zabih, "Graph cuts using alpha-beta swaps," *PAMI*, 2001.

[24] A. Klaus, M. Sormann, and K. K., "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," *ICPR*, pp. 15–18, 2006.

[25] V. Kolmogorov and R. Boykov, "Computing visual correspondence with occlusions using graph cuts," *ICCV*, 2001.

[26] M. Ma, "Modified census transform based on the related information of neighborhood for stereo matching algorithm," *Computer Engineering and Applications*, 2014.