Region-Based Progressive Stereo Matching

Yichen WEI Long QUAN
Department of Computer Science, Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{yichenw,quan}@cs.ust.hk

Abstract

A novel region-based progressive stereo matching algorithm is presented. It combines the strengthes of previous region-based and progressive approaches. The progressive framework avoids the time consuming global optimization, while the inherent problem, the sensitivity to early wrong decisions, is significantly alleviated via the region-based representation. A growing-like process matches the regions progressively using a global best-first strategy based on a cost function integrating disparity smoothness and visibility constraint. The performance on standard evaluation platform using various real images shows that the algorithm is among the state-of-the-art both in accuracy and efficiency.

1. Introduction

Stereo matching is inherently ambiguous in homogeneous and occluded areas. Only salient features, such as points of interest, can be matched reliably. This naturally motivates the development of progressive approaches [7, 14, 26, 20, 5, 10]. Such approaches first match those reliable features(generally pixels, and regions in our method). The earlier matched features will guide the subsequent matching in a growing-like process. An ambiguous decision is postponed until enough confidence can be accumulated to resolve the ambiguity. One main strength of progressive approaches is the low computational complexity since no global optimization is required. However, the inherent problem is the sensitivity to early wrong decisions. Since no back-tracking technique is used, a wrong decision cannot be corrected and will corrupt the subsequent matching rapidly.

A lot of region-based stereo algorithms arise recently [21, 9, 25, 3, 15, 24]. Although quite different from each other, they benefit from the common strengthes intrinsic to the region-based representation: the capability of occlusion handling around region boundaries and disparity regularization inside regions. The success of those algorithms motivates us to ask that whether it is possible to integrate a region-based approach within a progressive framework and, if so, will the two different categories of algorithms complement each other to offer a better algorithm?

We believe that the answers should be positive. An important observation is that, since regions contain much richer information than individual pixels, the possibility of making a wrong decision upon a region could be greatly reduced. Therefore, the inherent problem in a progressive approach, the sensitivity to early wrong decisions, could be alleviated.

Guided by these ideas, a region-based progressive matching algorithm is developed. It is simple, robust and efficient, compared with previous progressive or region-based algorithms. It is evaluated on the standard platform on web proposed by Scharstein and Szeliski [17, 18, 1] and also compared with other unambiguous matching algorithms[16, 22, 23, 10]. Results show that our algorithm is among the state-of-the-art, both in accuracy and efficiency.

Overview of our approach The algorithm starts from reliably matched pixels, called *ground control points*, *GCPs* [5, 10]. Regions are firstly obtained via color segmentation, then followed by a novel dynamic region splitting method. Reliable regions are firstly identified and matched using GCPs. Remaining regions are matched progressively in a growing-like process using a global best-first strategy based on a cost function that integrates disparity smoothness and visibility constraints and an ambiguity measure that is defined to be the ratio of the best and second best costs. Generally, matched regions propagate from textured areas to homogeneous and occluded areas. The progressive matching process stops if dense disparity map is obtained or a prespecified reliability threshold is reached. Figure 1 shows disparity maps produced in different stages of the algorithm.

Organization Section 2 discusses some preliminaries of the algorithm. Section 3, 4 and 5 describe the GCP computation method, the method of matching reliable regions from GCPs and the progressive matching algorithm, respectively. Experimental results are reported in Section 6. Section 7 concludes the paper.

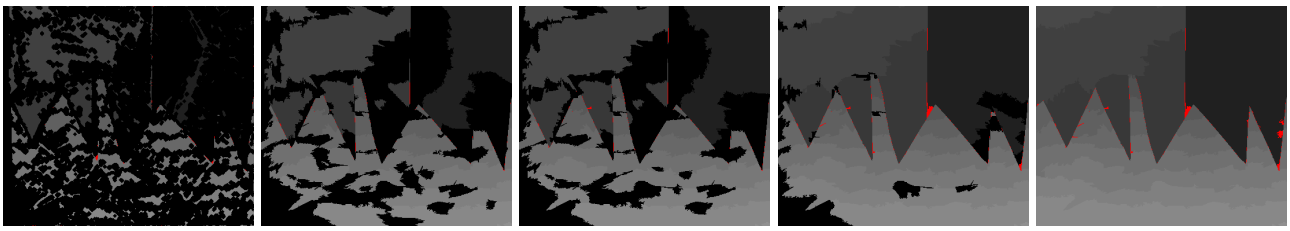


Figure 1. Disparity maps produced during different stages of the algorithm. Left most is the ground control points. Error disparities are shown in red (see the electronic version). One could see that textured regions are generally matched at first. The homogeneous and occluded regions, which are more ambiguous, are matched at last (refer to Figure 6).

2. Preliminaries

2.1. Definitions

We use a pair of horizontally rectified stereo images to ease the description of the algorithm through out the paper. While the algorithm only computes one disparity map, the generalization for multiple input images is straightforward.

Let $\mathbf{I}_0$ denote the reference (left) image and $\mathbf{I}_1$ the second (right) image. The disparity map $\mathbf{D}$ of $\mathbf{I}_0$ is a function that assigns each pixel p in $\mathbf{I}_0$ a disparity d, a horizontal displacement vector, such that p in $\mathbf{I}_0$ and $p + d$ in $\mathbf{I}_1$ correspond to the same 3D point. Starting from an empty disparity map, $\mathbf{D}(p) = undefined, \forall p \in \mathbf{I}_0$, a progressive algorithm defines $\mathbf{D}$ gradually.

A pair $(pixel, disparity)$ is called a $match$, which could be considered a 3D point. The current disparity map defines currently found matches, $\{(p, \mathbf{D}(p))|\mathbf{D}_p \neq undefined\}$. A matching cost function $C(p, d)$ measures the cost of a match based on pixel color or intensity in the images,

$$C(p, d) = f(\mathbf{I}_0(p), \mathbf{I}_1(p + d)) \quad (1)$$

This function can be the usual SSD(sum of squared difference), SAD(sum of absolute difference) or normalized correlation. We use the pixel dissimilarity measure in [2] that proves to be insensitive to the image sampling noise.

2.2. Constant Disparity Assumption

The reference image $\mathbf{I}_0$ consists of non-intersecting regions that are initially produced by a color segmentation algorithm using mean shift [8]. We assume that pixels inside the same region have the same disparity and our algorithm actually assigns each region a disparity. This assumption makes our method very simple and efficient.

The assumption seems quite restrictive since it is only valid for fronto-parallel surfaces and becomes problematic when a region represents a pronounced slanted surface or crosses surface boundaries. However, we claim that, the limitation could be significantly alleviated and the assumption becomes a very good approximation in practice by taking the following measures.

Over-segmentation Disparity discontinuities usually coincide with intensity edges that can be readily captured by a color segmentation algorithm. This is the underlying assumption of a lot of region-based approaches [21, 9, 25]. Almost all the object boundaries, i.e., the disparity discontinuities, can be detected using over-segmentation.

Dynamic region splitting After color segmentation, we take an additional region splitting method that helps capturing object boundaries missed by color segmentation and decomposes a pronounced slanted surface into small regions, therefore making the constant disparity assumption a good approximation. The method is described in Section 4.

2.3. Constrained Matching Cost

At any stage in a progressive scheme, currently found matches should constrain the subsequent matching and help to resolve the ambiguity. In this section, we describe how to efficiently integrate visibility and smoothness constraints into the pixel matching cost computation. These constraints are fully exploited in the region-based framework.

Visibility Constraint For visibility reasoning, an auxiliary disparity function, $\mathbf{D}_1$, is defined on $\mathbf{I}_1$. It is produced by projecting to the viewpoint of $\mathbf{I}_1$ all the currently found matches in $\mathbf{D}$. If multiple matches are projected to the same position in $\mathbf{I}_1$, which indicates the occurrence of occlusion, only the one with the largest disparity is recorded in $\mathbf{D}_1$. There are, of course, pixels in $\mathbf{D}_1$ that are undefined.

For a candidate match (p, d) being considered, there are three cases for its projection in $\mathbf{I}_1$. It may project to a pixel that is either undefined, occluded by a previously found match, or occluding a previously found match, according to the information stored in $\mathbf{D}_1$. Since earlier found matches are considered more reliable, their later occlusion should be prohibited. The matching cost integrating occlusion penalty

is defined as

$$\overline{C}_{vis}(p,d) = \begin{cases} C(p,d) & \mathbf{D}_1(p+d) \text{ is undefined} \\ \lambda_{occ} & \mathbf{D}_1(p+d) \geq d \\ C(p,d) + \lambda_{occ} & \mathbf{D}_1(p+d) < d \end{cases} \tag{2}$$

where λ_{occ} is a positive constant.

Smoothness Constraint Also known as *gradient limit constraint*, this constraint states that the disparity map is smooth almost everywhere and is used in a lot of stereo algorithms, explicitly or implicitly. Let $\mathcal{N}$ denote the 4-connected neighborhood system, $\mathcal{N} = \{(p,q) \mid |p_x - q_x| + |p_y - q_y| = 1\}$, the smoothness cost is defined for only pixels on the region boundary as

$$\begin{aligned} &\overline{C}_{smooth}(p,d) = \lambda_{smooth} \cdot \\ &|\{q|(p,q) \in \mathcal{N} \wedge p,q \text{ are not in the same region} \\ &\wedge (\mathbf{D}(q) = undefined \vee \mathbf{D}(q) \neq d)\}| \end{aligned} \tag{3}$$

where λ_{smooth} is a positive constant.

Intuitively speaking, function (3) will increase the cost by λ_{smooth} once for each pair of pixels whose disparities are not equal. Although the smoothness constraint is ensured inside the regions, function (3) will help to regularize the disparity map a lot since there will generally be a lot of small regions.

Constrained Matching Cost It is defined as

$$\overline{C}(p,d) = \overline{C}_{vis}(p,d) + \overline{C}_{smooth}(p,d) \tag{4}$$

It is noteworthy that eq. (2), (3) and (4) are unary functions, different from those defined in graph cut algorithms [6, 13] where the smoothness and visibility term are binary functions. While the latter is obviously more general and powerful, the former is simpler, more computationally efficient and can exploit occlusion and smoothness constraints fully within the region-based framework, since occlusion does not occur inside a region and smoothness inside a region is guaranteed.

3. Reliable Pixel Matching

Our algorithm starts by matching those highly reliable pixels, so called *ground control points(GCPs)* [5, 10]. It could be shown that carefully computed GCPs are almost free of outliers. Therefore, they can serve as a good starting point.

Let d_p^{best} be the best disparity of a pixel p in terms of the cost function (1), $d_p^{best} = \arg\min_d C(p,d)$. p is a *candidate GCP* if

$$\begin{aligned} &C(p,d_p^{best}) \leq \lambda_{amb} * C(p,d'), \forall d' \neq d_p^{best}, \text{and} \\ &C(p,d_p^{best}) \leq \lambda_{amb} * C(p',d'), \forall (p',d'), p'+d' = p + d_p^{best} \end{aligned} \tag{5}$$

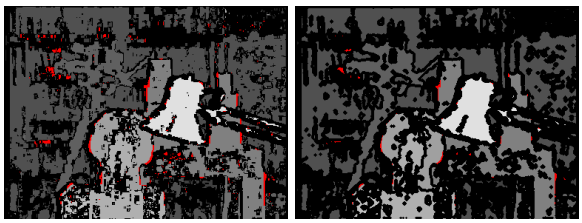


Figure 2. Computed ground control points($\lambda_{amb} = 0.4$). Outliers are shown in red (see the electronic version). Left: before post-processing. Right: after post-processing.

where $\lambda_{amb} (0 < \lambda_{amb} \leq 1)$ is the ambiguity level used through out the whole algorithm. The disparity of a candidate GCP p is d_p^{best}.

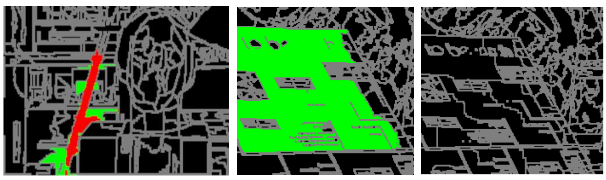
Condition (5) subsumes the commonly used winner-takes-all strategy and left-right consistency checking (λ_{amb}=1). It actually states that a reliable match (p, d_p^{best}) should have a cost value salient enough to stand out from those of all competing matches. A small amount of aggregation over $C(p,d)$ is necessary to make (5) robust. A constant 5×5 local window is used. Note that the window-based 2D aggregation is only used here and all the other computation involved in the whole algorithm is purely based on pixels and regions.

The outliers in the candidate GCPs(see Figure 2) can be roughly classified into two categories, those caused by random noise in the image and those caused by so called *fore-ground fattening* problem, which systematically distribute near the surface boundary on the side of homogeneous background. While the first category of errors can be handled simply, e.g., by eliminating isolated GCPs [5] or those with very large standard deviations in a small neighborhood [25], the second category of errors cannot be handled trivially.

Observing that a lot of small holes exist in highly reliable areas, we take a post-processing step to eliminate second category of outliers by firstly applying a morphological closing to fill the holes and then an erosion to shrink the boundaries of connected GCPs of the same disparity. Errors are reduced significantly after post-processing, at the price of sacrificing many correct GCPs (see Figure 2), and resulting GCPs are almost free of outliers.

4. Reliable Regions from GCPs

As observed, the GCPs are relatively dense in textured areas and almost free of outliers. Naturally, the more GCPs in a region of the same disparity, the more reliable it is to assign the region the disparity. Based on this observation, we



(a) (b) (c)

Figure 3. Dynamic region splitting. (a) Splitting as a bi-labelling problem (see electronic version for visualization). The boundary of camera pole missed by color segmentation is detected. (b)(c) A pronounced slanted surface(b) is split into smaller regions(c) that can be approximated by constant disparity.

first discuss two related issues and then introduce an efficient initialization algorithm that matches reliable regions.

Dynamic region splitting is necessary when a region contains GCPs with more than one disparity, when the region crosses the surface boundary or represents a pronounced slanted surface. For computational efficiency consideration, we assume that pixels in the region have only two different disparities. This is not essential since a region can be split iteratively.

The splitting is a process of assigning each pixel one of the two disparities that receives the most and second most votes from the GCPs in the region. New regions consist of connected pixels with the same assigned disparity. Figure 3 demonstrates examples.

The disparity assignment is a bi-labelling problem that can be addressed as an energy minimization problem in MAP-MRF framework and solved exactly by graph cut algorithm [11, 6]. We use the cost function (2) as the data term, instead of cost (1) as in [6]. The former is more informative in the context of a progressive framework. The smoothness term is the same as in [6]. The same smoothness penalty λ_{smooth} is used as in (3). Refer to [6] for more details.

Spatial distribution of GCPs must be considered. A region has good distribution of GCPs if the range of all GCPs is larger than half the range of the region in both horizontal and vertical directions and the density of GCPs is larger than a pre-defined threshold, $\rho_{density}$(0.25 used in the experiments). These simple heuristics work well in practice.

Initialization algorithm labels each region as *MATCHED* or *UNMATCHED*. All regions are unlabelled initially. The following marking process is applied iteratively until all regions get labelled: (1) if a region R contain GCPs of the same disparity and good distribution, label R as *MATCHED*, assign R the disparity and update disparity functions $\mathbf{D}$, $\mathbf{D}_1$ accordingly; (2) if R does not contain GCPs or contains GCPs of the same disparity value but not good distribution, label R as *UNMATCHED*; (3) if R contains GCPs with more than one disparity, split R and make new regions unlabelled. If only one region is generated, label it as *UNMATCHED*.

5. Progressive Matching Scheme

Before introducing the progressive scheme, several important issues are firstly discussed, including the definition of confidence and ambiguity, how to set ambiguity threshold and the algorithm flexibility and efficiency. The complete progressive matching algorithm is given in Figure 4

Confidence Taking into account the definition of the cost function (4), it is obvious that the more neighbors get matched, the more information is available and the more reliable matching is possible for a region. Therefore, the confidence for a region is defined as the size of already matched neighboring pixels. For no biasing towards large region, not the absolute size but a ratio is used. More specifically, let $contour(R)$ denote the set of pixel pairs relating R to its neighbors,

$$contour(R) = \{(p,q)|p \notin R \wedge q \in R \wedge (p,q) \in \mathcal{N}\},$$

the confidence of R is defined as

$$confidence(R) = \frac{|\{(p,q)|(p,q) \in contour(R) \wedge \text{p is matched}\}|}{|contour(R)|} \quad (6)$$

Ambiguity The ambiguity of a region is simply defined as the ratio of its best and second best matching cost, the same as in the GCP computation(see Section 3).

The matching cost of a region R is simply the summation of the constrained pixel matching cost (4), $\overline{C}(R,d) = \sum_{p \in R} \overline{C}(p,d)$. Let d_R^{best} and d_R^{second} be the best and second best disparity of R in terms of $\overline{C}(R,d)$, the ambiguity of R is

$$ambiguity(R) = \overline{C}(R, d_R^{best}) / \overline{C}(R, d_R^{second}).$$

Ambiguity Threshold Setting The ambiguity threshold λ_{amb}(as in Section 3) is set empirically initially. In each iteration, only those regions with ambiguities lower than the threshold will be matched. To achieve dense matching, it has to increase after each iteration. However, it is apparently not favorable to set the increasing amount a constant.

Instead, we take a simple adaptive strategy. A fixed-size array A_{amb} is used to store the expected ambiguity thresholds generated in the current iteration and the new threshold will be set accordingly before next iteration. Intuitively speaking, only $|A_{amb}|$ regions with smallest ambiguities exceeding the current threshold will have their ambiguity value stored in the array. λ_{amb} is set to the maximum value

1. Arrange all *UNMATCHED* regions in descending order of confidence (6). Set A_{amb} empty.
2. For each region R with $confidence(R) \neq 0$
   (a) If $ambiguity(R) \leq \lambda_{amb}$, label R as *MATCHED* and assign it the disparity d_R^{best}. Update disparity functions $\mathbf{D}, \mathbf{D}_1$ accordingly and go to Step 2.
   (b) If R contains GCPs with more than one disparity, split R, label new regions as *UNMATCHED* and go to Step 2.
   (c) If there is an entry in A_{amb} empty or with a value larger than $ambiguity(R)$, set the value of the entry to $ambiguity(R)$.
3. If there are new regions labelled as *MATCHED*, go to Step 2.
4. If there are *UNMATCHED* regions, set λ_{amb} to the maximum value in A_{amb}, go to Step 1; or exit when a pre-defined condition is met in case that dense matching is not desired.

Figure 4. Progressive matching algorithm. We use a global best-first strategy which has two-fold meaning: (1) Regions with larger confidence are attempted earlier; (2) Regions with lower ambiguity are accepted earlier.

in the array after the current iteration and those regions will become matched in the next iteration.

Flexibility Dense matching is obtained when λ_{amb} increases to 1. If reliable sparse matching is desired, the algorithm can be terminated in Step 4 when a pre-defined condition is met, e.g., λ_{amb} reaches a upper bound or desired matching density is achieved. The size of A_{amb} controls the matching speed smoothly. The algorithm accuracy is insensitive when $|A_{amb}|$ is small(10 in the experiments).

The scheme allows combination with an *ad-hoc* post-processing algorithm. For example, we can run the graph cut algorithm [6] for all unmatched pixels using the constrained cost function (4) as the data term to handle occlusion, or compute the disparity for unmatched regions using the greedy algorithm in [21]. While this is beyond the scope of this paper, it remains as future work.

Efficiency The whole algorithm allows an efficient implementation due to the constant disparity assumption and the efficient computation of constrained matching cost (4).

Stereo algorithms are roughly classified as global or local in [17]. Our algorithm does not involve frequent time-consuming aggregation of support, compared with a local algorithm, and does not optimize a global function, compared with a global algorithm. Actually, the most time-consuming component is the color segmentation and the progressive step is very fast.

6. Experiment

We evaluated the algorithm on the test bed proposed by Scharstein and Szeliski [17] that is available on web [1]. The evaluation metric is the percentage of error disparities differing from the true value more than 1 pixel. Four data sets, Tsukuba, Sawtooth, Venus, Map, are used for quantitative evaluation while two data sets, Teddy, Cones, are used for qualitative evaluation.

Parameter setting All the parameters are fixed for the first four data sets, including those in the color segmentation. λ_{amb} is empirically chosen as 0.4 initially (also used in GCP computation). The algorithm consistently performs well for all data sets when λ_{smooth} and λ_{occ} are less than 6 and we believe this is due to the cost function (1) we used [2]. We use $\lambda_{occ} = 4$ and $\lambda_{smooth} = 3$ to produce the results shown in other figures. For Cone and Teddy, due to the large disparity range, we set λ_{amb} to 0.5 initially to obtain higher GCP density.

Running performance Table 1 summarizes the running performance obtained on a Pentium IV 1.9G PC. The progressive part is quite efficient, while most time is consumed in segmentation step. Since we only tried one segmentation algorithm currently, it is not clear how the algorithm will perform using other simpler and faster segmentation algorithms. This remains as future work.

Comparison with Unambiguous Matching Algorithms Due to the difficulty of dense matching, some researchers turned to investigate the problem of unambiguous dense matching [16, 22, 23, 10]. A progressive algorithm naturally performs such a task. Figure 5 shows our intermediate results as λ_{amb} increases gradually.

Setting $\lambda_{amb} = 0.8$ as a threshold upper bound, table 2 compares our results with those produced by other algorithms. For the first three data sets, our results are obviously better than [16, 22] and comparable with [23, 10].

Overall Comparison Figure 6 shows the computed dense disparity maps as well as the ground truth. We obtained very good results for Tsukuba, Sawtooth and Venus data sets, but less satisfactory for the very textured Map that can be in fact easily handled by local methods. Even for the difficult Cones and Teddy, due to their large disparity range, our algorithm works quite well.

The overall evaluation and comparison on the standard platform [1] is given in Table 3. The error metric is calculated over three different areas in the image, classified as untextured(untex), discontinuous(disc) and the entire image(all). Our overall rank is 6^{th} out of about 30 algorithms. There is actually little difference between the top ten algo-

rithms. Although it is not possible to perform a comparison of running time, we believe that our algorithm should be one of the fastest between those methods ranking at top since most of them are global methods.

7. Conclusions

In this paper, we presented a novel stereo matching algorithm that integrates a region-based representation into a progressive framework. A robust, flexible and efficient progressive matching algorithm is developed. It is relatively insensitive to the parameter setting and performs consistently well for almost all the data sets tested in the experiments. Evaluation and comparison result shows that our algorithm is among the state-of-the-art.

Acknowledgements

This work is supported by the Hong Kong RGB grant HKUST 6188/02E.

References

[1] http://www.middlebury.edu/stereo/.

[2] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. IEEE *Trans. on Pattern Analysis and Machine Intelligence*, 20:401–406, 1998.

[3] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *Proceedings of the 7th Int. Conf. on Computer Vision, Kerkyra, Greece*, pages 489–495, September 1999.

[4] M. Bleyer and M. Gelautz. A layered stereo algorithm using image segmentation and global visibility constraints. Submitted to ICIP 2004.

[5] A. F. Bobick and S. S. Intille. Large occlusion stereo. *International Journal of Computer Vision*, 33(3):181–200, 1999.

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. IEEE *Trans. on Pattern Analysis and Machine Intelligence*, pages 1222–1239, 2001.

[7] Q. Chen and G. Medioni. A volumetric stereo matching method: Application to image-based model. In *Proceedings of the Conf. on Computer Vision and Pattern Recognition, Fort Collins, Colorado, USA*, 1999.

[8] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. IEEE *Trans. on Pattern Analysis and Machine Intelligence*, Vol.24:603–619, May 2002.

[9] F. Ernst, P. Wilinski, and K. V. Overveld. Dense structure-from-motion: An approach based on segment matching. In *Proceedings of the 7th European Conf. on Computer Vision, Copenhagen, Denmark*, 2002.

[10] M. L. Gong and Y. H.Yang. Fast stereo matching using reliability-based dynamic programming and consistency constraints. In *Proceedings of the 9th Int. Conf. on Computer Vision, Nice, France*, 2003.

[11] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51:271–279, 1989.

[12] L. Hong and G. Chen. Segment-based stereo matching using graph cuts. In *Proceedings of the Conf. on Computer Vision and Pattern Recognition*.

[13] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proceedings of the 7th European Conf. on Computer Vision, Copenhagen, Denmark*, 2002.

[14] M. Lhuillier and L. Quan. Match propogation for image-based modeling and rendering. IEEE *Trans. on Pattern Analysis and Machine Intelligence*, vol 24:pages 1140–1146.

[15] M. H. Lin and C. Tomasi. Surfaces with occlusions from layered stereo. In *Proceedings of the Conf. on Computer Vision and Pattern Recognition*, 2003.

[16] R. Sara. Finding the largest unambiguous component of stereo matching. In *Proceedings of the 7th European Conf. on Computer Vision, Copenhagen, Denmark*, 2002.

[17] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *International Journal of Computer Vision*, volume 47, pages 7–42, April 2002.

[18] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *Proceedings of the Conf. on Computer Vision and Pattern Recognition*, 2003.

[19] J. Sun, H. Y. Shum, and N. N. Zheng. Stereo matching using belief propagation. In *Proceedings of the 7th European Conf. on Computer Vision, Copenhagen, Denmark*, 2002.

[20] R. Szeliski and D. Scharstein. Symmetric sub-pixel stereo matching. In *Proceedings of the 7th European Conf. on Computer Vision, Copenhagen, Denmark*, 2002.

[21] H. Tao and H. Sawhney. A global matching framework for stereo computation. In *Proceedings of the 8th Int. Conf. on Computer Vision, Vancouver, Canada*, pages 532–539, July 2001.

[22] O. Veksler. Dense features for semi-dense stereo correspondence. *International Journal of Computer Vision*, 47(1-3):247–260, April 2002.

[23] O. Veksler. Extracting dense features for visual correspondence with graph cuts. In *Proceedings of the Conf. on Computer Vision and Pattern Recognition*, 2003.

[24] Y. Wei and L. Quan. Fast segmentation-based dense stereo from quasi-dense matching. In *Proceedings of the Asian Conf. on Computer Vision, Jeju, Korea*, 2004.

[25] Y. Zhang and C. Kambhamettu. Stereo matching with segmentation-based cooperation. In *Proceedings of the 7th European Conf. on Computer Vision, Copenhagen, Denmark*, 2002.

[26] Z. Zhang and Y. Shan. A progressive scheme for stereo matching. *Second European Workshop on 3D Structure from Multiple Images of Large-Scale Environments(SMILE2000)*, July 2000.

	Tsukuba	Sawtooth	Venus	Map	Cone	Teddy
image size	384×288	434×380	434×383	284×216	450×375	450×375
max disparity	15	21	19	28	55	52
regions initially(finally)	828(966)	1459(1692)	981(1236)	819(1124)	1822(1822)	1636(1636)
iterations	15	25	23	16	37	18
time(segmentation)	21	40	41	13	37	34
time(progressive)	1.07	2.99	4.99	0.73	9.88	5.88

Table 1. Running performance. Time is in seconds.

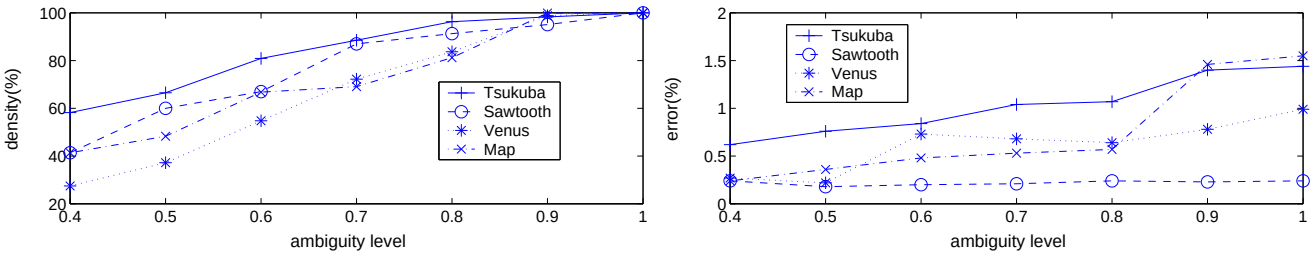


Figure 5. Matching density and error rate plotted versus λ_{amb} as the algorithm runs progressively.

algorithm	Tsukuba		Sawtooth		Venus		Map	
	D(%)	e(%)	D(%)	e(%)	D(%)	e(%)	D(%)	e(%)
our method($\lambda_{amb} = 0.8$)	96.3	1.07	91.3	0.24	83.7	0.64	81.2	0.57
sara02 [16]	45.7	1.4	52	1.6	40	0.8	74	0.3
veksler02 [22]	66	0.38	76	1.62	68	1.83	87	0.22
veksler03 [23]	75	0.36	87	0.54	73	0.16	87	0.01
gong03 [10]	85.7	1.07	85	0.41	67.1	0.51	60.8	0.09

Table 2. Comparison of our method with other unambiguous matching algorithms in terms of matching density(D) and error rate(e).

	Tsukuba			Sawtooth			Venus			Map	
algo	all	untex	disc	all	untex	disc	all	untex	disc	all	disc
Segm.-based GC[12]	1.23	0.29	6.94	0.30	$\mathbf{0.00_1}$	3.24	$\mathbf{0.08_1}$	$\mathbf{0.01_1}$	$\mathbf{1.39_1}$	1.49	15.46
Segm.+glob.vis.[4]	1.30	0.48	7.50	$\mathbf{0.20_1}$	$\mathbf{0.00_1}$	$\mathbf{2.30_1}$	0.79	0.81	6.37	1.63	16.07
Layered[15]	1.58	1.06	8.82	0.34	$\mathbf{0.00_1}$	3.35	1.52	2.96	2.62	0.37	5.24
Belief prop.[19]	$\mathbf{1.15_1}$	0.42	$\mathbf{6.31_1}$	0.98	0.30	4.83	1.00	0.76	9.13	0.84	5.27
GCMulCam[13]	1.85	1.94	6.99	0.62	$\mathbf{0.00_1}$	6.86	1.21	1.96	5.71	0.31	4.34
Our method	$\mathbf{1.44_6}$	$\mathbf{0.55_6}$	$\mathbf{8.18_7}$	$\mathbf{0.24_2}$	$\mathbf{0.00_1}$	$\mathbf{2.64_2}$	$\mathbf{0.99_5}$	$\mathbf{1.37_8}$	$\mathbf{6.40_8}$	$\mathbf{1.49_{20}}$	$\mathbf{17.11_{27}}$

Table 3. Evaluation table(incomplete) on the web [1]. Algorithms are in order of their ranking. Our algorithm ranked 6^{th} out of about thirty algorithms. Subscript number is relative rank in each column.

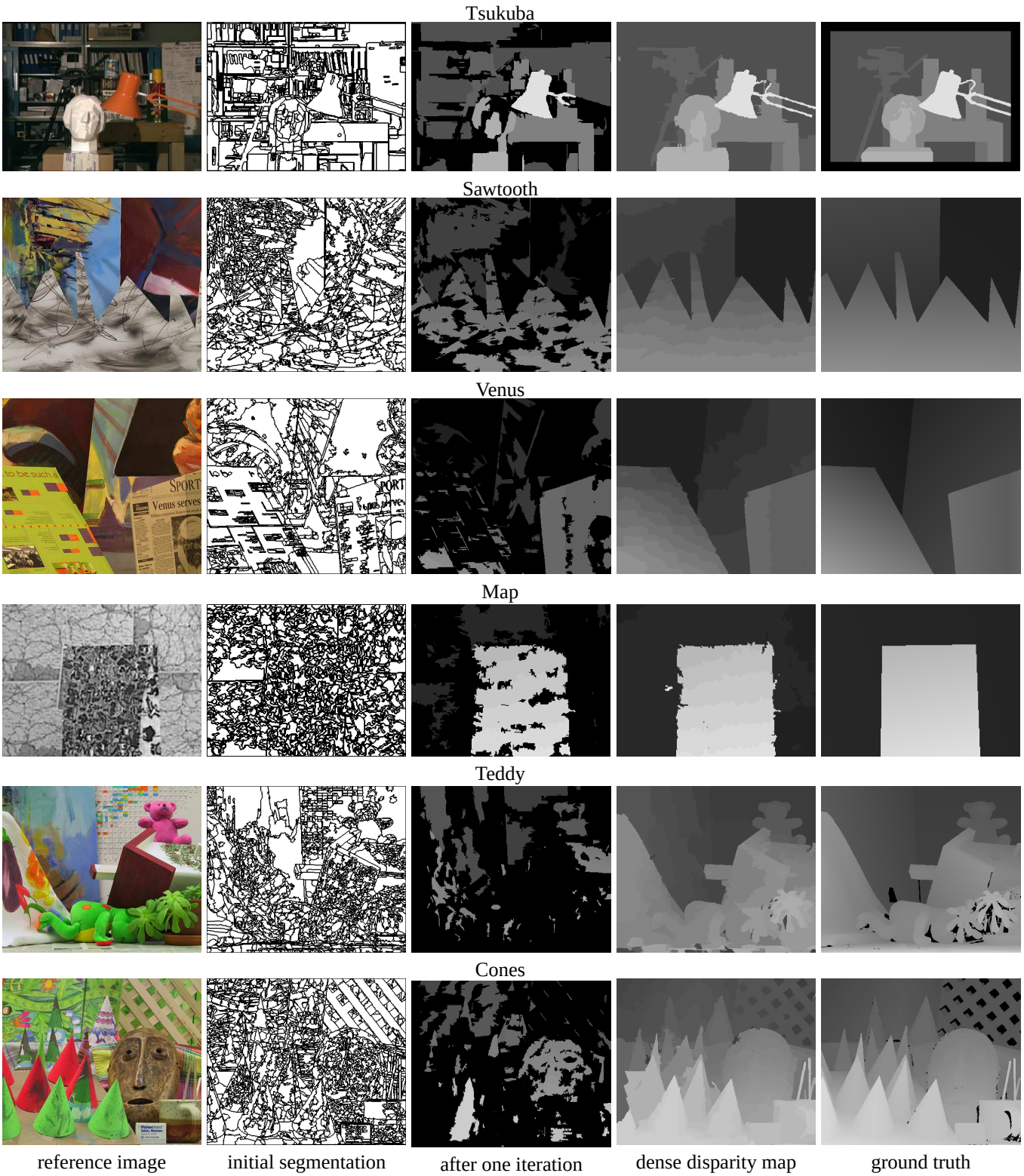


Figure 6. Dense disparity maps produced by our method. We obtained very good results for almost all data sets except the very textured Map which can be actually easily handled by local methods. For Teddy and Cones, although very limited initial reliable regions are found, the final results are very visually satisfactory. Most object details and surface boundaries are finely recovered, including the bear and flowers in Teddy, and thin sticks and most cones in Cones, mainly due to the reasonable segmentation and the robust progressive scheme. The only obvious error observed is in the middle left of Cones, caused by an early wrong decision.