YCVIU 1143 ARTICLE IN PRESS No. of Pages 17, DTD = 5.0.1
24 August 2004 Disk Used Gomti (CE) / Prabhakaran (TE)

Available online at www.sciencedirect.com
SCIENCE DIRECT®

ELSEVIER
Computer Vision and Image Understanding xxx (2004) xxx–xxx

Computer Vision and Image Understanding

www.elsevier.com/locate/cviu

Improved range image segmentation by analyzing surface fit patterns

Jaesik Min*, Kevin W. Bowyer

Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA

Received 11 June 2003; accepted 28 June 2004

Abstract

We propose a new approach to range image segmentation of planar and curved surface scenes. Our method is mainly an extended design of an existing algorithm, which was guided by a framework of performance evaluation. We choose the range segmentation algorithm developed by Jiang and Bunke as our baseline algorithm, which is fast and has shown relatively high performance in several experimental performance evaluation studies. We analyze the types of errors made by the algorithm, propose design modifications to decrease the error rate, and experimentally verify that the new approach achieves statistically significant performance improvement. Whereas the baseline algorithm applies the edge-linking uniformly to all edge pixels to segment a region, the modified algorithm selects high potential edge areas in the region by analyzing the surface fit pattern and gives priority of edge-linking to those areas. The contributions of this work are (1) an improved algorithm for segmentation of range images of both planar and curved surface scenes, and (2) a demonstration of using empirical performance evaluation to guide algorithm design and modification to achieve better performance.

Index terms: Range image; Segmentation; Performance evaluation

* Corresponding author. Fax: +1 574 631 9260.
E-mail addresses: jmin@cse.nd.edu (J. Min), kwb@cse.nd.edu (K.W. Bowyer).

1077-3142/$ - see front matter
doi:10.1016/j.cviu.2004.06.003

1. Introduction

Many algorithms have been presented for range image segmentation, and some of them have undergone experimental comparisons [7,16,9,15], and novel approaches have been presented to get better performance. Jiang and Kuhni [12] presented a contour closure algorithm for better performance in edge-based region segmentation. Cinque [5] used genetic algorithms for optimal setting of range segmentation algorithm parameters. Bellon and Silva [1] presented an improvement by using an edge detection technique. Other various techniques [3,4,13] have also been used for better range segmentation.

We present an improved version of a range image segmentation algorithm originally developed at the University of Bern (UB) [11], which uses an edge-based approach and is applicable to both planar and curved surface scenes. The UB algorithm has shown dominance both in performance and speed in an experimental comparison [16]. The basic strategy of the algorithm is to start from coarse initial segmentation based on edges of the range image and to proceed to further refinement. Since a typical edge extraction has "gaps" along true edges, most initial regions are under-segmented and need to be split into smaller ones. For each region, the baseline algorithm hypothesizes a quadratic surface. If the amount of surface fit error is greater than a predefined threshold, the region is recursively split until every sub-region satisfies its own surface hypothesis. The splitting is achieved by linking edge segments inside the region, and the linking is achieved by dilating every edge pixel of the region.

When splitting a region, the baseline algorithm discards all qualitative and quantitative information regarding the surface fit errors, and depends only on the binary edge map. Our improved approach focuses on how to use the surface fit errors to better split the initial under-segmented regions. We define three prominent patterns of surface fit error which are frequent in failed surface hypotheses and are useful in determining proper splitting actions. The experiment was performed by utilizing a range image segmentation evaluation framework [14] and the improvement of performance was verified by statistical significance tests.

The remaining sections of this paper are organized as follows. Section 2 explains how the baseline range image segmentation algorithm works and how it was trained for our experiments. Section 3 analyzes the drawback of the baseline algorithm and introduces new methods to make improvement. Section 4 briefly describes the evaluation framework that we employ for comparing the baseline and the improved algorithms. Section 5 shows the results of both algorithms and compares them, and Section 6 summarizes our work and introduces topics to be addressed in future work.

2. The baseline algorithm

The UB algorithm is a fast range segmentation algorithm that is based on edge detection along scan lines. By dealing with curve segments as the data primitives, it reduces the amount of data, thus obtaining its speed. The algorithm consists of three steps: edge extraction, edge grouping, and post-processing, each of which will be explained in the following subsections.

2.1. Edge extraction

The edge extraction method used in the UB algorithm is described in [10]. The edge detector scans the range image along four directions: horizontal, vertical, and two diagonals. Each scanned line is a three-dimensional curve. Partitioning each scan line into quadratic curve segments is performed by using the classical line fitting algorithm [6]. The end points of these segments are viewed as the potential edge points. The edge strength of the edge candidates are evaluated by computing the height difference (jump edges) and the angle difference (crease edges) between two adjacent curve segments. Each potential edge pixel can be assigned up to four edge strength values of each type from the four scan lines passing through the pixel (Fig. 1). Among these edge strength values, the maximum values are taken to define the overall edge strength of each edge type (jump or crease). Finally, the candidate edge pixel is determined as an edge point if at least one type of edge strength is greater than the corresponding threshold value.

2.2. Edge grouping

For best segmentation performance, the edge that matches the region boundary should be a closed contour. But, due to noise in the raw image and other reasons,

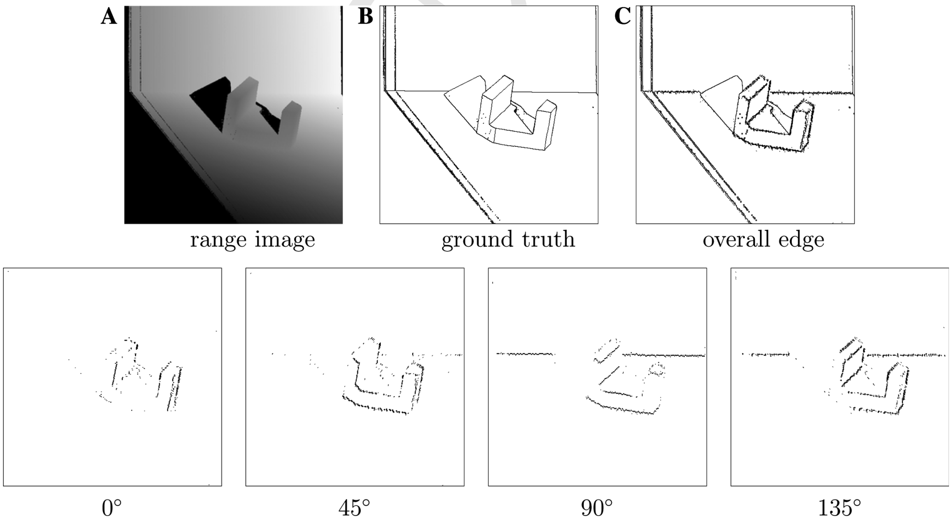
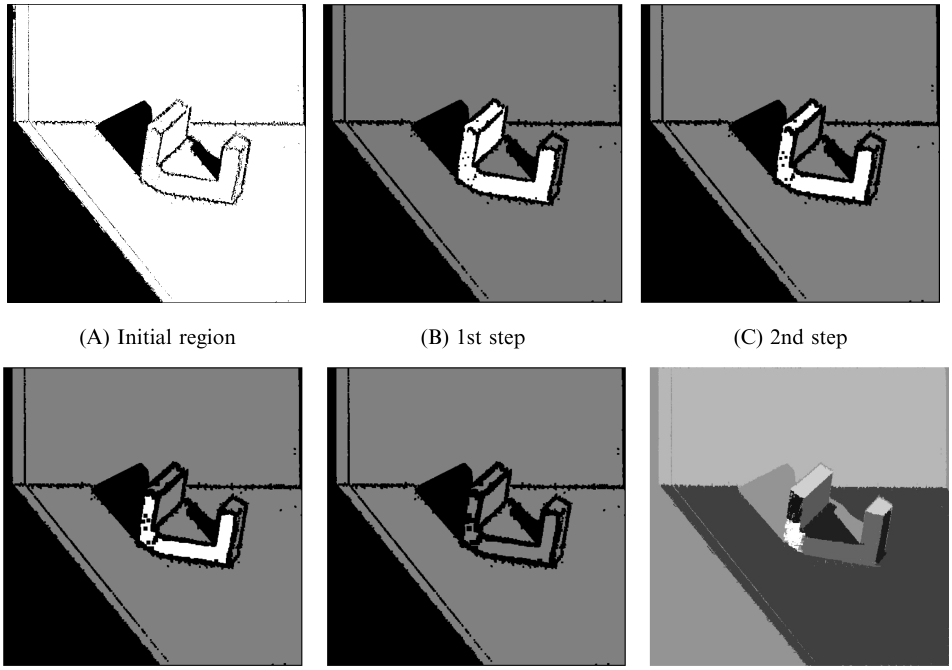


Fig. 1. Edge extractions along four scan directions. In the overall edge map, edges created due to shadow regions are also drawn.



(A) Initial region (B) 1st step (C) 2nd step

(D) 3rd step (E) 4th step (F) final

Fig. 2. Region splitting (edge grouping) of the baseline algorithm. (A) The incomplete edge extraction from Fig. 1 causes a large initial under-segmented region (white). (B–D) Each step splits the under-segmented region (white). (E) Splitting is finished. (F) After the post-processing, eroded pixels are recovered to adjacent regions.

it is common that the true edges are not in the form of closed contours. After the edge detection is completed, the algorithm performs the connected component labeling to generate the initial segmentation. Since the true edges are not fully connected at the initial step, the initial segmentation result tends to have a high ratio of under-segmented regions.

The UB algorithm makes a surface hypothesis on each segmented region and tests the hypothesis by calculating the fitting error between the segmented region and the hypothesized surface. If the surface fit errors (RMSE and average) of a region are less than predefined thresholds (T_a^c and T_r^c for curved surface, or T_a^p and T_r^p for planar surface scenes, discussed later), then the region is accepted as the final segmentation; otherwise the algorithm splits the region into subregions. The process is performed recursively until all the subregions either pass the surface hypothesis or get smaller below the predefined minimum size, which is one of the algorithm parameters.

The specific region splitting method used by the baseline algorithm is as follows. The algorithm dilates all the edge pixels inside the region—including the region boundary—in order to fill the gaps between the true edge segments. After each dilation step, new surfaces are computed for the newly shaped regions (Fig. 2). Note that surface fit error tends to get smaller as the area gets smaller, especially on curved surfaces. The edge dilation scheme is purely based on the binary edge map of the region. Therefore, the scheme requires that the binary edge map is of reasonable quality. Otherwise, linking of false edges (in a dense edge map) or excessive dilation of edges (in a sparse edge map) will occur.

2.3. Post-processing

After the edge grouping is completed, a post-processing is performed to process unlabeled pixels so far. The unlabeled pixels include those that were eroded in the process of edge dilation. Such pixels are merged to an adjacent region as long as the fit error after the merging is tolerable. Less strict values are set for the surface fit error thresholds: $T_a^{\mathrm{p}} \times T_f^{\mathrm{p}}$ or $T_a^{\mathrm{c}} \times T_f^{\mathrm{c}}$. Figs. 2E and F shows before and after of the post-processing.

2.4. Training

The UB algorithm has a total of 10 parameters as shown in Table 1. Parameters T_g, T_j, and T_c are used in the edge extraction step, and parameters T_r^{p}, T_r^{c}, T_a^{p}, T_a^{c}, and T_s are related to surface approximation, thus used in the edge grouping step. The remaining parameters T_f^{p} and T_f^{c} are used in the post-processing step and specify how the parameters T_a^{p} and T_a^{c}, respectively, can be relaxed in merging the unlabeled pixels.

As it is impractical to train the algorithm over all the parameters because of the computational load of training, we selected the most four significant parameters in training: a set of (T_g, T_j, T_s, and T_c) for the ABW images and another set of (T_g, T_j, T_c, and T_r^{c}) for the Cyberware images. A total of 69,400 executions of the baseline algorithm (65 CPU hours on a Sun Fire 880) were performed in training the algorithm over the 10 ABW training sets and 78,300 executions (48 h on a Sun Fire 880) in training the algorithm over the 10 Cyberware training sets.

Table 1
Parameters of university of bern segmenter

Name	Description
T_g	Max. distance between scan line and quadratic curve fit
T_r^{p}	Planar surface approximation RMS error: $\sqrt{\sum \mathrm{error}^2/\mathrm{RegionSize}}$
T_r^{c}	Curved surface approximation RMS error: $\sqrt{\sum \mathrm{error}^2/\mathrm{RegionSize}}$
T_a^{p}	Planar surface approximation average error: $\sum\lvert\mathrm{error}\rvert/\mathrm{RegionSize}$
T_a^{c}	Curved surface approximation average error: $\sum\lvert\mathrm{error}\rvert/\mathrm{RegionSize}$
T_j	Threshold of jump edge strength
T_c	Threshold of crease edge strength
T_s	Minimum number of pixels of a legitimate region
T_f^{p}	Tolerance of T_a^{p} in the post processing
T_f^{c}	Tolerance of T_a^{c} in the post processing

3. Improvement

Getting correctly closed edges is important for the UB algorithm to achieve a successful segmentation. The algorithm assumes that a moderately well extracted edge map is given initially and refines the initial segmentation by subsequent edge linking procedures. One thing to note here is that in the UB algorithm the surface hypothesis is performed for every region, but the qualitative result of the hypothesis is not used at all in handling the failed surface hypothesis. That is, the surface fit is used only to pick up the under-segmented regions, and refining those under-segmentations is performed without knowing *how* the surface fits the region.

For better edge linking, a new adaptive contour closing algorithm that uses a direction-guided edge grouping approach has recently been presented by Jiang [8]. It also assumes that no other information is given except for the binary edge map of the scene. In other words, the approach is considered as a stand-alone edge grouping algorithm rather than a component of a segmentation algorithm. As mentioned in the paper, its performance is highly dependent on the edge shapes and the result of contour closing is not successful in some cases. Besl and Jain [2] designed an algorithm that starts from coarse segmentation initially created by using surface curvature sign labeling and refines it by an iterative region-growing that is based on the surface fitting errors.

In this paper we present a new approach to improve the segmentation performance of the UB algorithm by using the surface fit error information in determining edge linking. The new approach is focused on the second step of the baseline algorithm, that is, *edge grouping*; other components of the original algorithm, such as edge extraction and post-processing, remain untouched.

3.1. Surface fit error map

The main potential drawback of the baseline algorithm is the unnecessary erosion of non-edge regions in the process of edge linking. Whenever the surface hypothesis fails, which means that a region turns out to be under-segmented, the baseline algorithm tries to split the region by linking the gaps between edge segments. The linking is performed by dilating all the edge pixels of the region including the boundary. In many cases, when the edge segments are disconnected only by several pixels and noise level is low, the algorithm produces very reliable and fast splitting results. In many other cases, however, this simple scheme causes several problems. When the edge map is too dense, many false edge points tend to be linked together, creating unwanted false edge contours. This results in over-segmentation, or more severely when the partitions are too small, missed regions in the final output (Fig. 3). On the other hand, when the edge map is too sparse—few edge pixels exist except for the boundary of the region—linking the edge segments takes repeated steps of edge dilation, shrinking the region as a result of excessive inward erosion from boundary edges, which sometimes is not able to be recovered even by the post-processing of the baseline algorithm (Fig. 4). In both cases, it is very likely to miss part of the region. Setting proper

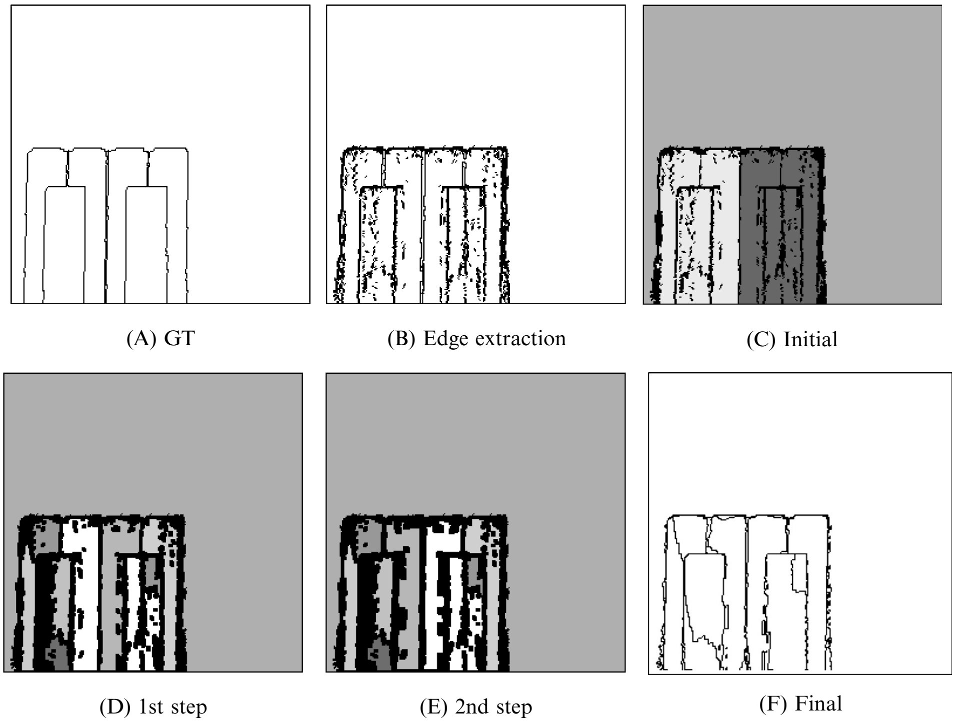


Fig. 3. Over-segmentation example of the baseline algorithm. Due to the dense edge map, noisy edge pixels are linked together, creating false contours. (C–E) Different intensities imply different regions.

values for the two edge-related thresholds (T_j and T_c) may help prevent these cases, but the threshold values usually have to be set over a number of training images of various scene quality.

The main idea of our improvement is to dilate edge pixels of an under-segmented region selectively, and the selection is based on how the hypothesized surface patch fits the region. We can categorize several patterns of surface hypothesis failure, for each of which we prescribe different action. Whenever the surface fit is made to a region, we build a surface fit error map of the region that represents the surface fit error amount of each pixel. For simpler representation, we assign grey-level value (0–255) to the fit error. That is, we assign *black* (=0) if $f(x,y) - z(x,y) \geqslant 2 * avgerr$, *white* (=255) if $z(x,y) - f(x,y) \geqslant 2 * avgerr$, where $f(x,y)$ is the surface point, $z(x,y)$ is the region point, and *avgerr* is the average surface fit error value of the region. Other fit error values in between are scaled into *grey*, so that grey-level 127 means zero fit error. The extreme fit error areas, *black* and *white*, will play a major role in the following subsections. Given an under-segmented region, instead of blindly dilating edge pixels, our new approach takes following three levels of action depending on the surface fit patterns:

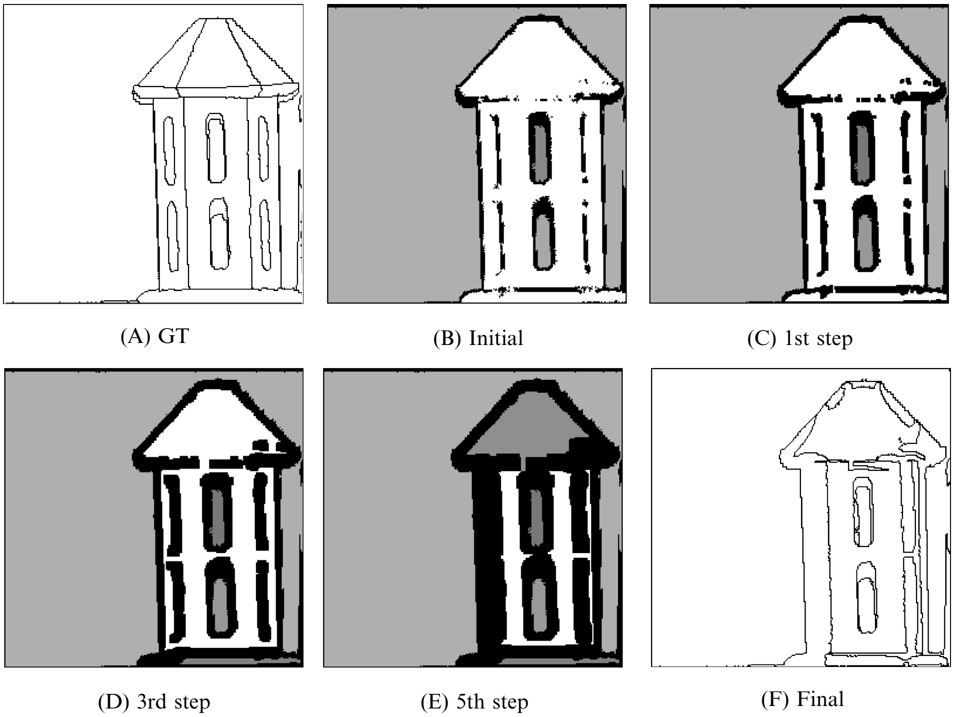


Fig. 4. Under-segmentation and missed region example of the baseline algorithm. Due to the sparse edge map, edge grouping is performed repeatedly, resulting in severe erosion of the initial region. Splitting is not successful and several regions are missed.

- *Direct split*. Whenever two opposite extreme areas (black and white) are adjoining, a border line is created along the zero-crossing line regardless of the existence of edge points.
- *Forced split*. Whenever an extreme area itself (black or white) splits the region, forced erosion is performed on that area until the split is accomplished.
- *Selective linking*. If the region has not been split at the two levels above, edge dilation is performed only on the areas with high fit error.

3.2. Direct split

It is not guaranteed that the very strong candidate of an edge is fully connected in the edge map. There are several possible reasons: too strict edge threshold, noisy inputs, or real disconnections of edge at some level. The surface fit error values around these disconnected high jump edges have patterns in which an area of highly negative (black) fit error is adjacent to an area of highly positive (white) fit error (Fig. 5C). Thus, it is intuitive to separate these areas no matter how many edge points already exist in the area. For each under-segmented region, all adjoining extreme areas with

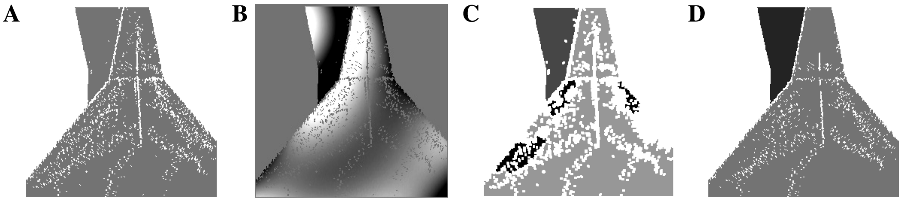


Fig. 5. Comparison of blind edge linking and direct split. (A) An under-segmented region. Due to incomplete edge detection, the upside-down funnel and the small triangular planar background are connected. (B) Surface fit error map of the region. Two extreme areas with opposite signs are adjoining in the upper part of the region. (C) Result of blind edge linking by the baseline algorithm. The funnel and background are detached, but the funnel is over-segmented at this early step. (D) Result of direct split. The funnel and the background are detached without any side-effects.

opposite signs (*black* and *white*) should be separated. As shown in Fig. 5, the split is done successfully and other irrelevant edge points remain intact, saving non-edge pixels from erosion.

3.3. Forced split

In general, crease edges are harder to extract than jump edges. The surface fit error values around the true crease edges also have patterns in which an area of extreme fit error (negative or positive) traverses the region. To determine the true crease edge region, we test every extreme area to see whether it really splits the region. After deleting the area temporarily from the region, we check if the deletion splits the region by performing connected component labeling upon the region. If the labeling shows two or more regions, we register it as a strong candidate for crease edge. Then the dilation of edge points is performed only in this area until it gets a split of the region. In the worst situation of the edge detection, we are given no edge point at all in this extreme area. Then the forced split will erode the region inwards from the boundary until the region is split. Fig. 6 shows the results both from the baseline and the improved algorithm.

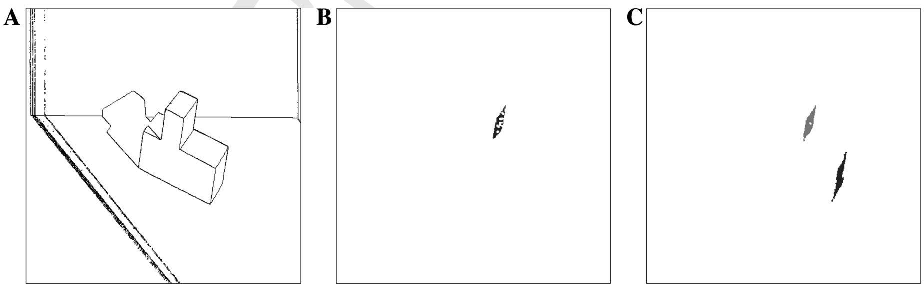


Fig. 6. Comparison of blind edge linking and forced split at the intermediate step (before post-processing). Only the regions of interest are shown. (A) Ground truth. (B) The baseline result. A small region at the rightmost side of the block is missing as result of excessive erosion of the region. Another region at the other right face of the block is eroded excessively. (C) The forced split result. The forced split has been applied. No region is missing and no region is eroded seriously.

3.4. Selective linking of edges

If the surface fit error map does not show any of the special patterns described above, or splitting the region is not completed even after applying the actions above, a selective dilation of edges is performed. This is similar to the original edge dilation scheme of the baseline, except that not all the edge pixels inside the region are selected for dilation. A typical surface fit error map of an under-segmented region has various areas of different fit errors. It is intuitive that edge points around the area with high surface fit error, regardless of the sign, are more likely—although not always true—to be near the true edges than ones with low surface fit error.

We divide the surface fitting errors into three groups, i.e., high (greater than two times of the average), medium (greater than the average), and low (less than the average), according to the relative value to the average fit error of the region. At the first step of dilation, edge points with high surface fitting error are dilated. If it succeeds in linking some edges and in splitting the region, then the dilation stops. Otherwise, dilating is performed on the edge pixels with medium surface fitting error. If it succeeds, then the same action will be taken. Otherwise, all the remaining edge pixels are dilated and performing this final step will have the same effect as the baseline algorithm. At worst, the algorithm with the new dilation approach does the same action as the baseline.

Both the baseline and the improved algorithm assume that a sufficient amount of edge pixels are provided along true edge contours. This assumption is more crucial to the baseline algorithm. Moreover, the baseline algorithm also assumes that false edge pixels are not dense in order to avoid over-segmentation. The second requirement is not crucial to the improved approach, therefore we can lower the edge thresholds without worrying about over-segmentation.

3.5. Training

We apply a different order of significance of the parameters to the baseline and the improved algorithms because they work differently. For example, for the ABW image set, the improved algorithm shows no performance improvement between 1-parameter and 2-parameter tuning when T_j is set for the second significant parameter as we did for the baseline algorithm. A new order of parameter significance was determined in a trial-and-error manner, as was done in the baseline algorithm.

We selected a set of parameters (T_g, T_r, T_j, and T_a^{p}) for the training on the ABW images, and another set of parameters (T_g, T_c, T_a^{c}, and T_j) for the training on the Cyberware images. A total of 76,446 executions of the improved algorithm (156 h on a Sun Fire 880) were performed in training the algorithm over the 10 ABW training sets and 76,428 executions (73 h on a Sun Fire 880) in training the algorithm over the 10 Cyberware training sets. The increased training time of the improved algorithm was expected because of several reasons. First, whenever a failed surface hypothesis is found, the algorithm tries to find surface fit patterns on which the direct split and/or the forced split operations are applicable. The baseline algorithm does not have this stage. Second, the pixels to be eroded by the selective erosion are a subset of the pixels to be eroded by the baseline algorithm. Therefore, the baseline algo-

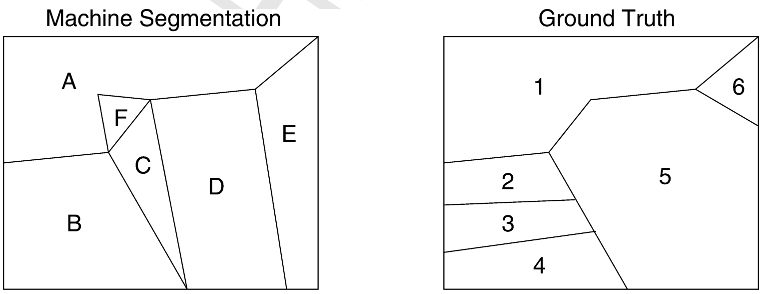
YCVIU 1143 ARTICLE IN PRESS No. of Pages 17, DTD = 5.0.1
24 August 2004 Disk Used Gomti (CE) / Prabhakaran (TE)

rithm finishes the region split much faster than the improved algorithm does, regardless of the segmentation performances.

4. Evaluation framework

The definition of the performance metrics for the segmentation is the same as used by Hoover et al. [7]. A machine segmentation (MS) of an image compares to the ground truth (GT) specification for that image to count instances of correct segmentation, under-segmentation, over-segmentation, missed regions, and noise regions. The definitions of these metrics are based on the degree of mutual overlap required between a region in the MS and a corresponding region in the GT. An instance of "correct segmentation" is recorded if and only if an MS region and its corresponding GT region have greater than the required threshold of mutual overlap. Multiple MS regions that correspond to one GT region constitute an instance of over-segmentation. One MS region that corresponds to several GT regions constitutes an instance of under-segmentation. A GT region that has no corresponding MS region constitutes an instance of a missed region. A MS region that has no corresponding GT region constitutes an instance of a noise region. Fig. 7 illustrates these definitions of the performance metrics. For the statistical test for significance, we use the number of instances of correct segmentation.

The performance evaluation framework [14] that we employ uses separate sets of images for train, validation, and test. The training step searches for the "best" parameter settings. The validation step decides how many of the segmenter's parameters should have their value learned through training versus left at the default value. The test step determines performance curves to be used in comparing different segmenters. Because the selected parameter settings may vary based on the particular set of training images, we create multiple different training sets by random sampling from a larger pool of training images. This applies to the validation and test sets, too.



MS A corresponds to GT 1 as an instance of correct segmentation.
GT 5 corresponds to MS C, D, and E as an instance of over–segmentation.
MS B corresponds to GT 2, 3, and 4 as an instance of under–segmentation.
GT 6 is an instance of a missed region.
MS F is an instance of a noise region.

Fig. 7. Illustration of definitions for scoring region segmentation results.

YCVIU 1143 ARTICLE IN PRESS No. of Pages 17, DTD = 5.0.1
24 August 2004 Disk Used Gomti (CE) / Prabhakaran (TE)

In general, typical algorithms have a number of parameters that control their operation and the default values for the parameters. This introduces the question of how many of the available parameters should be trained. After training on a given number of parameters, the parameter values for each training set are run on each validation set. If there are T_{tr} training sets and V validation sets, then $T_{tr} \times V$ performance curves are produced. If the improvement of the validation in going from $N-1$ to N parameters is statistically significant, then training is repeated using $N+1$ parameters. If there was no improvement in going to N parameters, then the $(N-1)$-parameter training result is kept.

The final trained parameter values from each training set are run on each test set, resulting in $T_{tr} \times T_{te}$ performance curves. The areas under these curves are used as the basis of a test for statistical significance of an observed difference in performance between segmenters. The performance is compared quantitatively and statistically by using the paired differences in the areas under the performance curves.

Assume that we are comparing a "challenger" algorithm to a "baseline" algorithm. The test statistic will be the difference between the areas under the performance curves. The sign test can be used to check for statistical significance without requiring the assumption that the differences follow a normal distribution. The null hypothesis is that there is no true difference in average performance between the algorithms. Under the null hypothesis, each algorithm has a 0.5 probability of generating the larger area under the performance curve on any given trial. The number of trials for which one algorithm generates a larger area than the other should follow a binomial distribution. Our framework implementation automatically reports the results of a sign test.

5. Experimental results

In Sections 2 and 3, both the baseline and the improved algorithm are trained using validation steps and 10 sets of trained parameters were applied to each of the 10 test sets. For each algorithm over each range data type, we got 100 (10 train sets $\times$ 10 test sets) performances of five different metrics (correct classification, under-segmentation, over-segmentation, missed region, and noise region) in the form of values of area under the performance curve. A paired sign test was performed on these 100 pairs of quantitative values to determine statistical significance. As is shown in subsequent subsections below, the improvement obtained by using the new algorithm is small but statistically significant (at the $\alpha = 0.05$ level) for both data sets.

5.1. Results on planar-surface image sets

The paired comparison of 100 performance values (correct classification) between two algorithms is shown in Fig. 8. The new algorithm produced slightly better performance than the baseline in 67 out of 100 instances. The improvement in correct classification mostly came from reduction of missed regions (Fig. 9), which, in the

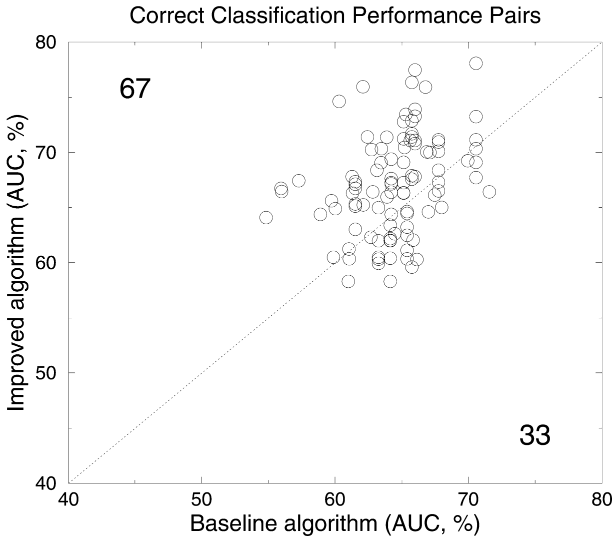


Fig. 8. Comparison of performance between the baseline and the improved algorithm on ABW test sets.

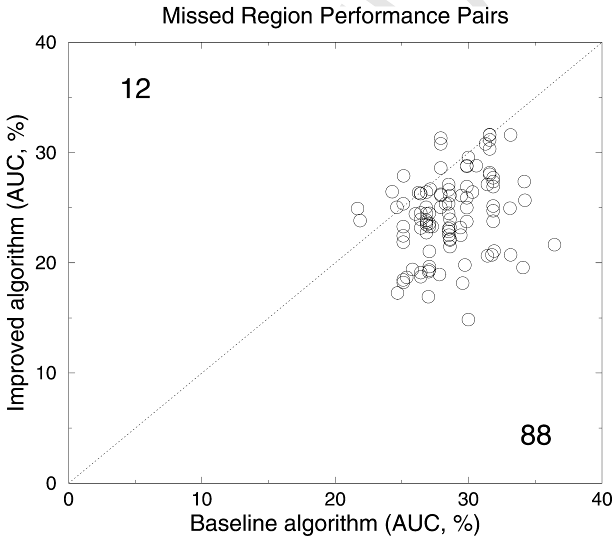
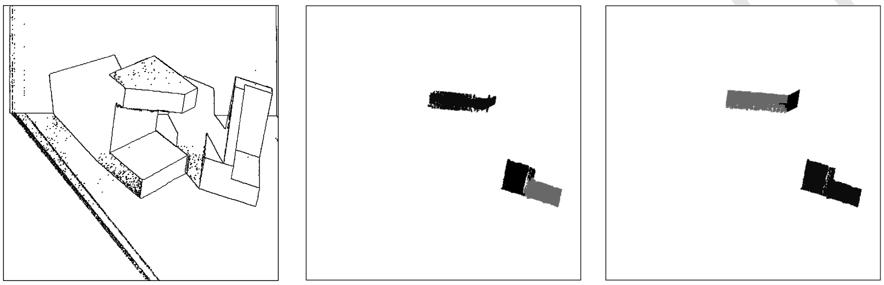


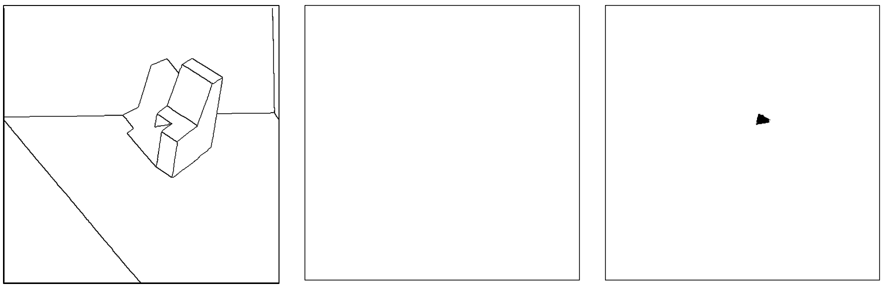
Fig. 9. Comparison of incorrect segmentations (missed regions) between the baseline and the improved algorithm on ABW test sets.

baseline algorithm, occurred mainly due to the excessive erosion of small regions. There was a small decrease in over-segmentations and a small increase in noise regions that in effect cancel each other out, and therefore did not influence the overall performance. The new algorithm produced more under-segmentations in 96 out of 100 cases, but the amounts were so small that it did not make difference.

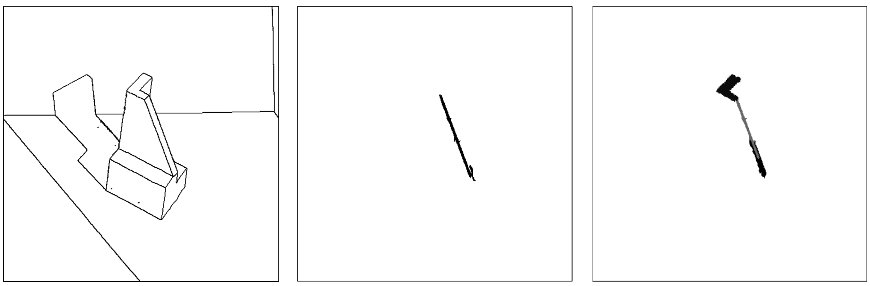
Fig. 10 shows machine segmentation samples from both algorithms. Note that setting different overlap thresholds produces different interpretations of the same segmentation result. For example, an instance of correct classification at a lower threshold switches to an error metric at a higher threshold. And an instance of over-segmentation at a lower overlap threshold (e.g., 51%) switches to an instance of missed region *plus* multiple instances of noise region at a higher overlap threshold (e.g., 95%). In the figure, we counted the number of instances of each performance metric at the fixed overlap threshold of 85%.



(A) GT (B) Baseline (C) Improved



(D) GT (E) Baseline (F) Improved



(G) GT (H) Baseline (I) Improved

Fig. 10. Segmentation comparison on ABW images at 85% overlap threshold. (A–C) Test image "abw.18." Two over-segmentations, two missed regions, and one noise region are recovered. (D–F) Test image "abw.28." One missed region is recovered. (G–I) Test image "abw.25." One missed region is recovered, but another missed region and additional two noise regions are created.

We can conclude that the new algorithm applied to the planar surface scenes improved the correct segmentation performance by relieving the excessive erosion problem of the baseline algorithm. The ABW data set contains many small planar regions which can be easily rejected from region acceptance due to small size after excessive erosion of the baseline algorithm. As all of the surfaces are planar, the new algorithm benefits from the forced split and selective erosion approaches because the boundaries of adjoining planar surfaces are more likely having extreme fit errors.

5.2. Results on curved-surface image sets

The paired comparison of 100 performance values between two algorithms is displayed in Fig. 11. The new algorithm produced better performance than the baseline in a statistically significant fraction of the results (72 out of 100 in the paired sign test).

By analyzing the performance of the baseline algorithm, we knew that its correct classification performance was largely influenced by its level of under-segmentation; on a curved surface, the algorithm tends to satisfy the fit error threshold by eroding the outer area of the under-segmented region. On the contrary, the improved algorithm is more likely to erode the inner area, which has high fit error. Thus, the new algorithm generally reduced the level of under-segmentation but did not have major effects on other performance metrics. An increase in the level of noise regions degraded performance on some images but this was often outweighed by the decrease in under-segmentation. There is no remaining predominant error tendency in the new algorithm. Fig. 12 shows machine segmentation samples from both algorithms.

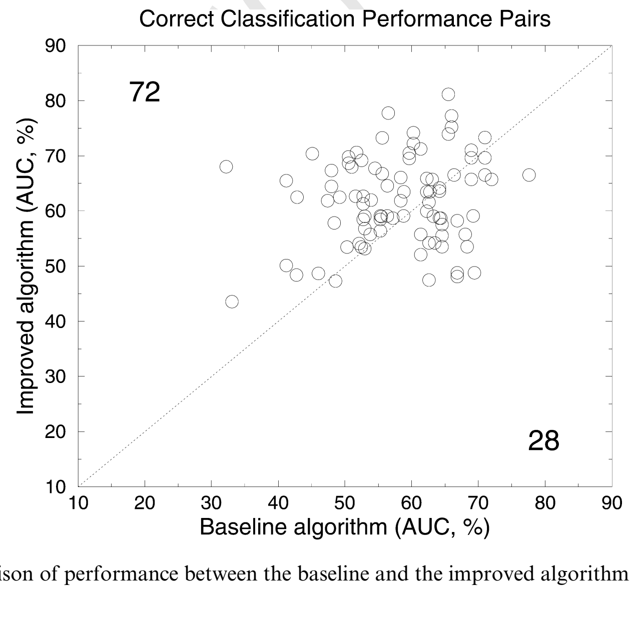


Fig. 11. Comparison of performance between the baseline and the improved algorithm on Cyberware test sets.

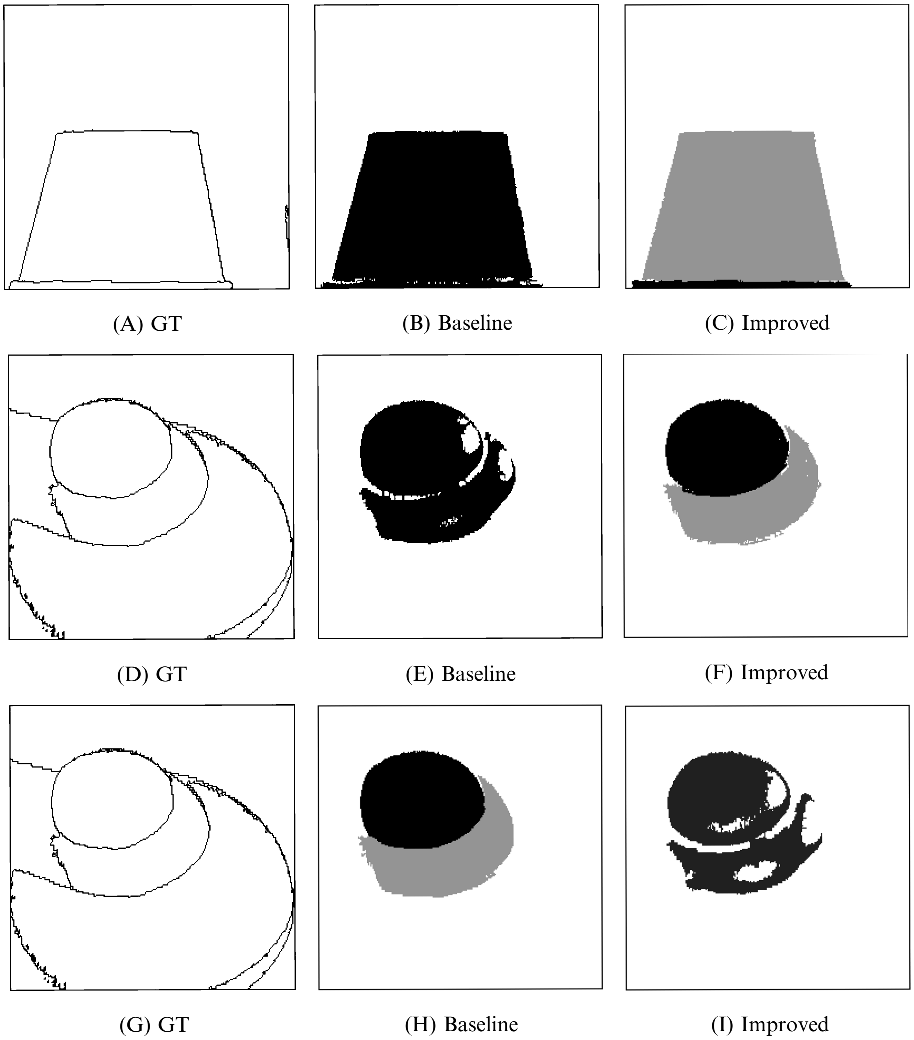


Fig. 12. Segmentation comparison on Cyberware images at 85% overlap threshold. (A–C) Test image "cone1." One under-segmentation is recovered. (D–F) Test image "snowman." Two missed regions and one noise region are recovered. (G–I) Test image "snowman" (with another parameter setting). Two missed regions and one noise region are created.

6. Summary and conclusions

An improvement in range image segmentation has been achieved by applying a new approach to handling failed surface hypothesis. Instead of linking edges blindly the new algorithm analyzes the surface fit patterns of the failed surface hypothesis. The improvement was verified by using a range image segmentation evaluation framework with sets of planar and curved surface scene images.

With image sets of uniform complexity, the produced edge maps may be of reasonable quality, so the baseline algorithm will perform well and be faster than the new approach. However, images with a wide variety in sizes of objects and/or in scene complexity will let the edge images be of low quality, so the baseline algorithm that only looks at the edge map would have difficulties in achieving a successful segmentation. By applying our novel approach, we were able to design an improved algorithm that is less sensitive to the edge extraction results.

References

[1] O. Bellon, L. Silva, New improvements to range image segmentation by edge detection, IEEE Signal Process. Lett. 9 (2) (2002) 43–45.

[2] P. Besl, R. Jain, Segmentation through variable-order surface fitting, IEEE Trans. Pattern Anal. Mach. Intell. 10 (2) (1988) 167–192.

[3] M. Bock, C. Guerra, Segmentation of range images through the integration of different strategies, in: Vision, Modeling, and Visualization, 2001, pp. 27–33.

[4] L. Cinque, F. Corzani, S. Levialdi, R. Cucchiara, Improvement in range segmentation parameters tuning. in: Int. Conf. on Pattern Recognition, vol. I, 2002, pp. 176–179.

[5] L. Cinque, R. Cucchiara, S. Levialdi, S. Martinz, G. Pignalberi, Optimal range segmentation parameters through genetic algorithms. in: Int. Conf. on Pattern Recognition, vol. I, 2000, pp. 474–477.

[6] R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley, 2001.

[7] A. Hoover, G. Jean-Baptiste, X. Jiang, P. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, R. Fisher, An experimental comparison of range image segmentation algorithms, IEEE Trans. Pattern Anal. Mach. Intell. 18 (7) (1996) 673–689.

[8] X. iang, An adaptive contour closure algorithm and its experimental evaluation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (11) (2000) 1252–1265.

[9] X. Jiang, K. Bowyer, Y. Morioka, S. Hiura, K. Sato, S. Inokuchi, M. Bock, C. Guerra, R. Loke, J. du. Buf, Some further results of experimental comparison of range image segmentation algorithms, in: Int. Conf. on Pattern Recognition, vol. IV, 2000, pp. 877–881.

[10] X. Jiang, H. Bunke, Edge detection in range images based on scan line approximation, Comput. Vis. Image Understand. 73 (2) (1999) 183–199.

[11] X. Jiang, H. Bunke, Range image segmentation: adaptive grouping of edges into regions, in: Asian Conf. on Computer Vision, 1998, pp. 299–306.

[12] X. Jiang, P. Kuhni, Search-based contour closure in range images, in: Int. Conf. on Pattern Recognition, 1998, pp. 16–18.

[13] I. Khalifa, M. Moussa, M. Kamel, Range image segmentation using local approximation of scan lines with application to cad model acquisition, Mach. Vis. Appl. 13 (5–6) (2003) 263–274.

[14] J. Min, M. Powell, K. Bowyer, Automated performance evaluation of range image segmentation algorithms, IEEE Trans. Syst. Man Cybernet. 34 (1) (2004) 263–271.

[15] J. Min, M. Powell, K. Bowyer, Automated performance evaluation of range image segmentation algorithms, in: Empirical Evaluation Methods in Computer Vision, World Scientific Press, 2002, pp. 1–22.

[16] M. Powell, K. Bowyer, X. Jiang, H. Bunke, Comparing curved-surface range image segmenters, in: Int. Conf. on Computer Vision, 1998, pp. 286–291.