An Experimental Comparison of Range Image Segmentation Algorithms

Adam Hoover[1], Gillian Jean-Baptiste[1], Xiaoyi Jiang[3], Patrick J. Flynn[2],
Horst Bunke[3], Dmitry Goldgof[1], Kevin Bowyer[1]
David Eggert[4], Andrew Fitzgibbon[4], and Robert Fisher[4]

1 Department of Computer Science & Engineering
University of South Florida
Tampa, FL 33620
hoover, jean, goldgof or kwb @csee.usf.edu

2 School of Electrical Engineering and Computer Science
Washington State University
Pullman WA 99164-2752
flynn@eecs.wsu.edu

3 Institute of Informatics, University of Bern, Switzerland
jiang or bunke@iam.unibe.ch

4 Department of Artificial Intelligence
University of Edinburgh
5 Forrest Hill
Edinburgh EH1 2QL Scotland
eggertd, andrewfg or rbf @aifh.ed.ac.uk

Abstract

A methodology for evaluating range image segmentation algorithms is proposed. This methodology involves (a) a common set of 40 laser range finder images and 40 structured light scanner images that have manually specified ground truth and (b) a set of defined performance metrics for instances of correctly segmented, missed and noise regions, over- and under-segmentation, and accuracy of the recovered geometry. A tool is used to objectively compare a machine generated segmentation against the specified ground truth. Four research groups have contributed to evaluate their own algorithm for segmenting a range image into planar patches.

Key words: experimental comparison of algorithms, range image segmentation, low level processing, performance evaluation

In general, standardized segmentation error metrics are needed to help advance the state-of-the-art. No quantitative metrics are measured on standard test images in most of today's research environments.

—NSF Range Image Understanding Workshop, 1988 [19]

The importance of theory cannot be overemphasized. But at the same time, a discipline without experimentation is not scientific. Without adequate experimental methods, there is no way to rigorously substantiate new ideas and to evaluate different approaches.

—Jain and Binford (CVGIP: Image Understanding, 1991 [20])

Comparison of segmentation results is difficult. This is because of the difficulty in implementing other people's algorithms due to [the] lack of necessary details. In many cases, we have not been able to reproduce the published results by using the author's algorithm. This is further complicated by the fact that there is no standard evaluation criterion.

—Yu, Bui & Krzyzak (PAMI, May 1994 [42])

1 Introduction

Important areas of computer vision suffer from a lack of sound experimental work [16, 19, 20, 34, 42]. An overview of the state of experimental evaluation of range image segmentation can be obtained from Table 1. Note that none of the methods listed have been evaluated using pixel-level ground truth in real images. Also note that none of the methods have been directly compared to other methods. The closest that there is to any common image data set is the "Renault part" image, the "coffee cup" image and the "MSU data set" images, each of which are mentioned in more than one paper. Two papers have used ground truth in the sense of comparing the geometry of recovered models to that of the shapes imaged [5, 31]. One paper, which emphasizes the speed of its approach, quotes execution times from papers describing other algorithms [22]. One paper, which emphasizes

Table 1: Summary of Recent Journal-Published Range Segmentation Algorithms

source	nature of algorithm	allowed region type	number of **real images** evaluated	**ground truth** used in evaluation	algorithms **compared** against
Besl & Jain '88	H-K map, iterative region growing	eight patch types based on H,K sign	6 shown	none	none
Bhandarkar & Siebert '92	hybrid edge and region based	planar patches	5 shown	angles, edge length, areas	none
Boyer, Mirza & Ganguly '94	robust sequential estimator	planar and biquadratic patches	1 shown	none	none
Ghosal & Mehrotra '93	moment-based region, edges	planar, quadratic	4 shown	none	none
Hoffman & Jain '87	clustering, statistical tests	planar, convex concave	5 shown	none	none
Jiang & Bunke '94	scan line division, merging	planar patches	3 shown	none	none
Krishnapuram & Gupta '92	edge-based	planar, curved	6 shown	none	none
LaValle & Hutchinson '95	Bayesian methods, surface fitting	polynomial patches	8 shown	none	none
Li '92	minimization framework using neural nets	eight patch types based on H,K sign	5 shown	none	none
Liou, Chiu & Jain '91	hypothesis test, region grow	biquartic patches	1 shown	none	none
Newman, Flynn & Jain '93	clustering, Hough transform and LS fitting	planes, cones, spheres, cylinders	5 shown	parameters of imaged objects	none
Sabata, Arman & Aggarwal '93	pyramidal clustering, then merging	polynomial surface patches	11 shown	none	none
Taylor, Savini & Reeves '89	split and merge	planar patches	none	none	none
Trucco & Fisher '95	diffusion, morphology and H,K thresholding	six patch types based on H,K sign	1 shown	none	none
Wani & Batchelor '94	edge masks, critical pts	edge map only	3 shown	none	none
Yokoya & Levine '89	edges and regions, H-K map	eight patch types based on H,K sign	1 shown	none	none
Yu, Bui & Krzyźak '94	"RESC" robust estimation	planar and quadric patches	5 shown	none	none

"Number of real images evaluated" is counted from figures or tables in the paper.

"Ground truth" is counted as specification of correct pixel labels and/or patch geometry.

"Algorithms compared against" is counted as side-by-side display of image results or table of results.

robust methods, compares its method with traditional least squares and least median of squares as the fitting techniques [42]. The table is not to single out any particular authors, or even the area of range segmentation. The situation is characteristic of essentially all of computer vision (e.g., edge detection). This deficiency in sound experimental work makes it difficult to assess the state of the art, particularly those aspects of a problem still requiring development. Dissemination of working theories to practitioners is also hampered.

Experimental comparisons of algorithms have recently been attempted in the areas of optical flow [2], stereo [6], and shape from shading [43]. Though these efforts represent positive steps, we feel that a guiding philosophy for the design of a comparative effort is lacking. A collective examination of these works, in addition to our own experience in range image segmentation, suggests that several factors are essential for comparative experimental efforts to have lasting value and impact:

1. *The comparative framework is itself a research issue, and so deserves appropriate conceptual energy in its development.*

   The framework centers around three elements: problem definition, performance evaluation, and data set. One surprising (and embarrassing?) thing about computer vision is that many intuitive low-level concepts have not yet come to have a rigorous, uniformly accepted definition. The example relevant here is the concept of a segmentation of an image. Highly-regarded texts give definitions which are largely similar, but which vary in the details (see Section 2.1). Similarly, subjective visual evaluation of results (which has evolved as the norm) should naturally give rise to skepticism. The evaluation procedure should be automated, and based upon objective performance measures (see Section 2.4). Finally, pre-existing or casually created imagery generally does not suffice. A thorough and challenging data set should be developed based upon a given problem definition (see Section 2.2). The effort of creating this framework is *substantial*, both in creative thought and painstaking data acquisition.

2. *Metrics are needed for error measurement, in addition to correct/valid performance.*

Just as measurements of accuracy and precision can each be useful in certain situations, there is usually more than one way to measure algorithmic performance. Some types of incorrect/invalid results might be acceptable while others are not. Thus multiple metrics are necessary for potential consumers to make intelligent decisions (see Section 2.4).

3. *The comparative study must use a "large", appropriately designed, real image data set, complete with ground truth.*

   Performance measurements based upon one or two images are generally worthless. Given the state of experimental computer vision today, "large" might mean tens of images. As experimental work becomes more common, the working definition of "large" should grow. Real images must be used. Simulated images may serve as a useful supplement when the tasks of obtaining and ground truthing sufficient real imagery is difficult. However, work that stops short of using real images inspires little confidence in its relevance. Establishing ground truth can require some ingenuity and is often painstaking, laborious and time-consuming. However, there simply is no other option.

4. *All input data, results and implementations must be made publicly available, both for potential consumers and for future incremental comparisons by others.*

   This is perhaps the single most important factor. It is bordering on unprofessional to publish results on images which are not available to other researchers. All input imagery, ground truth and results, as well as the code for the comparison tool and the segmentation algorithms presented herein, are available via http://marathon.csee.usf.edu/seg-comp/SegComp.html.

Some evaluations of intensity image segmentation algorithms (e.g., [32]) and thresholding algorithms (e.g., [25]) have been done. However, ground truth based on intensity is considerably more subjective than that based upon geometry. Previous works [26, 29] evaluate intensity image segmentations and offer a single overall goodness measure for the result. While a single measurement might seem appealing, we assert that it should be avoided. Although "valid" or "correct" results

generally warrant only one interpretation, invalid or incorrect results are not so easily evaluated, let alone weighed against each other.

This paper evaluates four segmentation algorithms on 80 real images (40 laser range finder and 40 structured light scanner) with ground truth and objective performance measures. This type of framework for a comparative effort (specific problem definition, objective performance evaluation, and large number of real images with ground truth) is essentially never used in mainstream computer vision, though it is standard practice in some related areas (e.g., optical character recognition). Besides the development of a philosophy of comparative experimental research, an important contribution here is an assessment of the state-of-the-art in planar range image segmentation. Based on our results, we assert that this problem is not "solved." This finding may be surprising and possibly controversial. We would welcome an empirical demonstration that the claim is false.

2 Comparative Framework

We restricted our work to comparison of planar segmenters. One reason is simply that developing a comparative framework for this problem seemed ambitious enough for a first step. Second, documenting the state of the art for planar segmentation seems intrinsically worthwhile. Third, the various algorithms for segmenting curved surface patches often do not allow the same set of possible surface types, making direct comparison more difficult. Lastly, there is always room for expansion of the framework in the future.

2.1 Range Image Segmentation: Problem Definition

Informally, segmenting a range image is the process of labeling the pixels so that pixels whose measurements are of the same surface are given the same label. The general problem of image segmentation is classical, and yet in four popular computer vision and image processing textbooks [1, 14, 15, 27], the formal definitions of the segmentation problem are slightly different. For instance, consider ([14], page 458):

Let $\mathbf{R}$ represent the entire image region. We may view segmentation as a process that partitions $\mathbf{R}$ into n subregions, $\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_n$, such that

1. $\cup_{i=1}^{n} \mathbf{R}_i = \mathbf{R}$,
2. $\mathbf{R}_i$ is a connected region, $i = 1, 2, \ldots, n$,
3. $\mathbf{R}_i \cap \mathbf{R}_j = \emptyset$ for all i and j, $i \neq j$,
4. $P(\mathbf{R}_i)$ = TRUE for $i = 1, 2, \ldots, n$, and
5. $P(\mathbf{R}_i \cup \mathbf{R}_j)$ = FALSE for $i \neq j$,

where $P(\mathbf{R}_i)$ is a logical predicate over the points in set $\mathbf{R}_i$ and $\emptyset$ is the null set.

Item 5 of this definition must be modified to apply only to adjacent regions, as non-bordering regions may well have the same properties; let this be called item 5a. In ([1], page 150), item 5a was advanced only as a possibile criterion. In ([27], page 388), item 5a was included, but item 2 was left out. In ([15], page 509), the formal definition was abandoned in favor of informal rules.

Besides these inconsistencies, there are technical difficulties in using this definition for range image segmentation. Some range pixels do not contain accurate depth measurements of surfaces. This naturally leads to allowing *non-surface*[1] pixels (areas), perhaps of various types. Regarding the above definition, non-surface areas do not satisfy the same predicate constraints (items 4 and 5) as regions that represent surfaces.[2] It is also often convenient to use the same region label for all non-surface pixels in the range image, regardless of whether they are spatially connected. This violates item 2 of the above definition. Finally, we also require that the segmentation be 'crisp'. No sub-pixel, multiple or 'fuzzy' pixel labelings are allowed.

2.2 Imagery Design

Given the above definition, consider the possible 'dimensions' of the range image planar segmentation problem:

1. Size (in pixels) of surface
2. Number of surfaces in the image

[1]The term "noise" is over-used and in fact not encompassingly descriptive here. For instance, triangulation-based scanners produce images containing areas where *no* range measurements were possible, due to occlusion.

[2]In essence, they satisfy the complement of predicate 4 (which is in this case joint membership to a surface); hence the term non-surface.

3. Incident angle of surface to viewpoint (angular difference between surface normal and viewpoint vector)
4. Crease edges
   (a) Angle between two surfaces of edge
   (b) Incident angle of edge to viewpoint
   (c) Edge length (in pixels)
5. Jump edges
   (a) Amount of depth discontinuity between two surfaces of edge
   (b) Edge length (in pixels)
6. #-bits/pixel (quantization level, or depth precision)
7. Amount and types of noise (besides quantization)

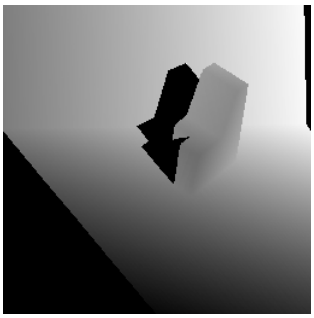
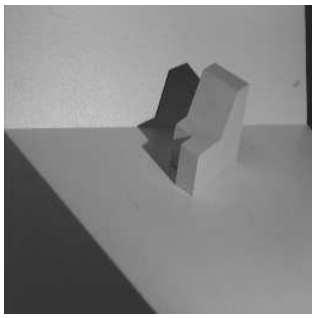
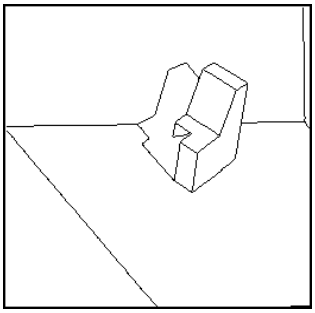
In the ideal situation, testing an algorithm on an image set that spanned the ranges of these dimensions would yield 'failure points' or 'tolerances'. However, acquiring, ground-truthing, processing and analyzing the necessary image data would require a prohibitive amount of effort. To reasonably explore the problem dimensions, we acquired 40 images (512×512 8-bit pixels) using an ABW[3] structured light scanner [36], and 40 images (512×512 12-bit pixels) using a Perceptron[4] laser range finder [33]. Although numerous methods to acquire range data have been demonstrated [3, 21, 41], these two types of sensors predominate.

Each image contains up to five polyhedral objects placed in a variety of poses and with varying degrees of inter-object spacing.[5] Although this image set does not explicitly cover all of the problem dimensions listed above, it does cover many properties. For instance, the number of surfaces generally grows as the number of objects in a scene increases. Conversely, the size of the largest surfaces (the backdrop and support planes) shrinks. There is also a general depth difference between jump edges caused by self-occlusion, and jump edges caused by inter-object occlusion. Figure 1 shows the ABW and Perceptron images which have the fewest number of surfaces (8 and 2) and the largest number of surfaces (36 and 32). Both of the image sets were randomly divided into a 10

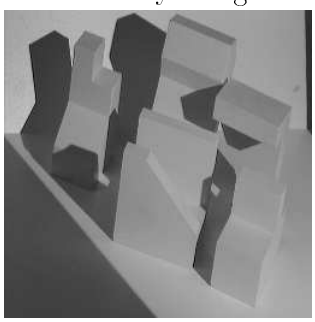
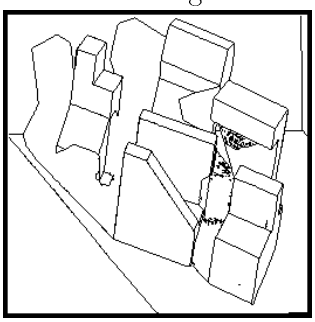
[3] address: ABW GmbH, Gutenbergstrasse 9, D-72636 Frickenhausen, Germany.

[4] address: 23855 Research Drive, Farmington Hills, MI 48335.

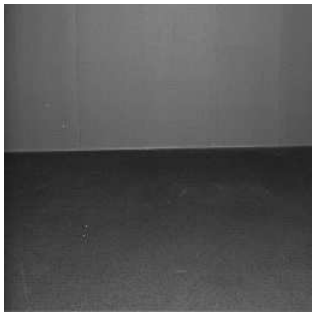
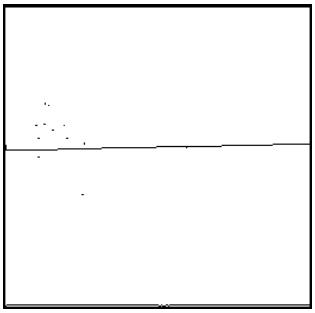
[5] The two cameras have different imaging volumes (the ABW's is table-top size while the Perceptron's is room size), so the same objects are not imaged by both. However, the two object sets exhibit similar complexity in terms of the number and spacing of surfaces.

ABW test image #28 intensity image ground truth segmentation

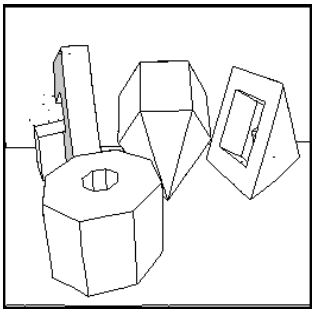




ABW train image #0 intensity image ground truth segmentation

Perceptron test image #14 reflectance image ground truth segmentation





Perceptron test image #26 reflectance image ground truth segmentation

Figure 1: Four of the eighty images used in this comparison (two of each type), and ground truths (outlines of borders of regions). The "specks" were caused by the outlining of non-surface areas. The ABW scanner uses structured light to obtain range values, so "shadow" areas are possible. Pixels in shadow areas have a value of zero and appear black. The larger a depth value the brighter the pixel.

image training set and a 30 image test set, for use in algorithm parameter setting and evaluation, respectively. There are 457 total ground truth segmented regions in the ABW test image set, and 438 total ground truth segmented regions in the Perceptron test image set.

2.3 Ground Truth

Ground truth was created for each image, consisting of a hand segmentation and a set of angles. The hand segmentations were created by a human operator outlining the boundary of each apparent surface patch in each image. The tracing is done in a magnification window so that each pixel can be considered individually in a reasonable fashion. Local contrast enhancement, the registered intensity or reflectance image, CAD models of the objects imaged, and the actual range values are all available to the operator for visualizing the regions.

Ten pixel labels were reserved for various types of non-surface pixels; at present four have been defined. A **shadow** pixel only occurs in a structured light scanner image, where the sensor is unable to make a range measurement. A **noise** pixel is an erroneous measurement of a single surface. A **cross-edge** pixel occurs when the footprint of the sensor covers more than one surface (only noticable along jump edges). Finally, we reserved the label **undiscernable surface detail** for image areas where the range readings are valid range measurements, but there is insufficient information to discern separation of surfaces (for instance, a one-pixel wide strip, or insufficient quantization). Unlike surface pixels, non-surface pixels are not considered to make up "regions", and do not contribute to the region mappings used for performance measures in this work.

Each hand segmentation was reviewed by a second human operator to catch any obvious errors. Finally, for any pair of hand segmented regions that correspond to a pair of neighboring object faces, the angle between the faces (as measured on the actual objects) was recorded.

2.4 Performance Metrics

Comparison of a machine segmentation (MS) of a range image to the ground truth (GT) is done as follows. Let M be the number of regions in the MS, and N be the number of regions in the GT. N

does not include any non-surface pixel areas (see Section 2.3). Similarly, M does not include any pixels left unlabeled (or not assigned to a surface) by the segmenter. Let the number of pixels in each machine-segmented region R_m (where $m = 1 \ldots M$) be called P_m. Similarly, let the number of pixels in each ground truth region R_n (where $n = 1 \ldots N$) be called P_n. Let $O_{mn} = R_m \cap R_n$ be the number of pixels whose same image coordinates both regions R_m and R_n occupy in their respective images. Thus, if there is no overlap between the two regions, $O_{mn} = 0$, while if there is complete overlap, $O_{mn} = P_m = P_n$.

An $M \times N$ table is created, containing O_{mn} for $m = 1 \ldots M$ and $n = 1 \ldots N$. Implicitly attached to each entry are the percentages of overlap with respect to the size of each region. O_{mn}/P_m represents the percentage of m that the intersection of m and n covers. Similarly, O_{mn}/P_n represents the percentage of n that the intersection of m and n covers. These percentages are used in determining region segmentation classifications.

We consider five types of region classifications: **correct detection**, **over-segmentation**, **under-segmentation**, **missed** and **noise**. Over-segmentation, or multiple detections of a single surface, results in an incorrect topology. Under-segmentation, or insufficient separation of multiple surfaces, results in a subset of the correct topology and a deformed geometry. A missed classification is used when a segmenter fails to find a surface which appears in the image (false negative). A noise classification is used when the segmenter supposes the existence of a surface which is not in the image (false positive). Obviously, these metrics could have varying importance in different applications. For instance, surface detection for collision avoidance would most likely require low instances of missed regions, yet be less sensitive to instances of noise regions. (It is more important to not run into anything that it is to go out of the way to avoid imaginary obstacles.) Conversely, a bin picking task would likely require low instances of noise regions, yet be less sensitive to instances of missed regions. (Given the abundance of available parts in a bin, it is more important to be sure of grabbing one of them than to be able to choose from all possible parts.)

The formulas for deciding classifications are based upon a threshold T, where $0.5 < T \leq 1.0$. The value of T can be set to reflect the strictness of definition desired. The following metrics define each classification:

1. An instance of a **correct detection** classification.

   A pair of regions R_n in the GT image and R_m in the MS image are classified as an instance of correct detection if

   (a) $O_{mn} \geq T \times P_m$ (at least T percent of the pixels in region R_m in the MS image are marked as pixels in region R_n in the GT image), and

   (b) $O_{mn} \geq T \times P_n$ (at least T percent of the pixels in region R_n in the GT image are marked as pixels in region R_m in the MS image).

2. An instance of an **over-segmentation** classification.

   A region R_n in the GT image and a set of regions in the MS image $R_{m_1}, \ldots, R_{m_x}$, where $2 \leq x \leq M$, are classified as an instance of over-segmentation if

   (a) $\forall i \in x, O_{m_i n} \geq T \times P_{m_i}$ (at least T percent of the pixels in each region R_{m_i} in the MS image are marked as pixels in region R_n in the GT image), and

   (b) $\sum_{i=1}^{x} O_{m_i n} \geq T \times P_n$ (at least T percent of the pixels in region R_n in the GT image are marked as pixels in the union of regions $R_{m_1}, \ldots, R_{m_x}$ in the MS image).

3. An instance of an **under-segmentation** classification.

   A set of regions in the GT image $R_{n_1}, \ldots, R_{n_x}$, where $2 \leq x \leq M$, and a region R_m in the MS image are classified as an instance of under-segmentation if

   (a) $\sum_{i=1}^{x} O_{m n_i} \geq T \times P_m$ (at least T percent of the pixels in region R_m in the MS image are marked as pixels in the union of regions $R_{n_1}, \ldots, R_{n_x}$ in the GT image), and

   (b) $\forall i \in x, O_{m n_i} \geq T \times P_{n_i}$ (at least T percent of the pixels in each region R_{n_i} in the GT image are marked as pixels in region R_m in the MS image).

4. An instance of a **missed** classification.

   A region R_n in the GT image that does not participate in any instance of correct detection, over-segmentation or under-segmentation is classified as missed.

5. An instance of a **noise** classification.

   A region R_m in the MS image that does not participate in any instance of correct detection, over-segmentation or under-segmentation is classified as noise.

Although these definitions result in a classification for every region in the GT and MS images, they are not unique for $T < 1.0$. However, for $0.5 < T < 1.0$ any region can contribute to at most three classifications, one each of correct detection, over-segmentation and under-segmentation. For a proof of this, see Appendix A. With any given mapping (of correct detection, over-segmentation or under-segmentation), there are two associated overall overlap metrics (computed as per the two parts of each definition). If for any given region only one mapping passes its definition, then the classification is done. When two or three mappings pass their definitions for the same region, then the mapping which has the highest average of its metric-pair is taken as the classification. On equal averages, we bias towards selecting correct detection, then over-segmentation, then under-segmentation.

Once all region classifications have been determined, a final metric describing the accuracy of the recovered geometry is computed, as follows. Any pair of regions R_{n_1} and R_{n_2} in the GT image which represent adjacent faces of the same object in the scene have their angle recorded in the truth data. Call this angle A_n. If R_{n_1} and R_{n_2} are both classified in instances of correct detection, then the angle between the surface normals of their corresponding regions in the MS image is computed. (It is assumed that the normals for each region in the MS are supplied with the segmentation.) Call this angle A_m. The absolute value of the difference between these two angles is computed, $|A_n - A_m|$. This is done for all of the correct detection classifications. The number of angle comparisons made, the average error and the standard deviation are reported. This measure gives an indirect estimate of the accuracy of the recovered geometry of the correctly segmented

portion of the image. Once again, it would be up to a consumer of the segmentations to decide on the importance of this measure. For instance, the accurate geometry might be more important for inspection (for defects) than for recognition.

We have created a tool which will automatically compare a specified ground truth and machine segmentation using these metrics. This tool was used to generate all results shown in this paper.

3 Experimental Methods

Four research groups each evaluated their own algorithm using the framework described. The algorithms are described in Section 3.1, while the parameter tuning processes (and values selected for testing) are described in Section 3.2.

3.1 Segmentation Algorithms

The four range segmentation algorithms evaluated here represent substantially different design choices. The USF and UE algorithms might be characterized as instances of the common approach to region segmentation by iteratively growing from seed regions. The WSU algorithm uses a powerful clustering algorithm to drive its segmentation. The UB algorithm uses a novel approach that exploits the scan line structure of the image. It would certainly tax most researchers to try to reason from theoretical principles which algorithm should excel on which performance metrics.

3.1.1 The USF range segmentation algorithm

This segmenter works by computing a planar fit for each pixel and then growing regions whose pixels have similar plane equations. A two-stage process is used to compute a pixel's normal. First, a growing operation is performed from the pixel, bounded by an $\mathbf{N} \times \mathbf{N}$ window. To join, a bordering four-connected pixel must be within $\mathbf{T_{perp}}$ range units. This has the effect of separating "outliers" from "inliers" (with respect to the central pixel), where the outliers could be across a jump edge, or simple noise. If less than 50% of the pixels within the window are inliers, then a single plane equation is fit to the pixels (using the eigen-method of [13, 8]). If 50% or more of

the pixels within the window are inliers, then a set of nine plane equations are computed using edge preserving sub-masks of the inliers in the $\mathbf{N} \times \mathbf{N}$ window. The nine sub-masks take the four compass directions, four diagonal directions, and the center. The plane equation from the submask which produces the smallest residual error is assigned as the normal of the pixel. For pixels close to crease edges, this procedure generally produces more accurate normals than would be obtained using a single mask. An "interiorness" measure is also found for each pixel as the residual error of the plane equation fit to the entire $\mathbf{N} \times \mathbf{N}$ window. This will generally be higher (less "interior") closer to edges.

The pixel with the smallest interiorness measure is chosen as a seed point for region growing. Criteria for pixels joining the region are (1) 4-connectivity, (2) angle between normal of pixel and normal of region grown so far within a threshold ($\mathbf{T_{angle}}^{\circ}$), (3) perpendicular distance between pixel and plane equation of region grown so far within a threshold ($\mathbf{T_{perp}}$ range units), and (4) point-to-point distance between pixel and 4-connected neighbor already in region below a threshold ($\mathbf{T_{point}}$ range units). The border of the region is recursively grown until no pixels join, at which time a new region is started using the next best available seed pixel (based on interiorness measure). Pixels are only allowed to participate in this process once. Initially, the plane equation for a region is calculated from the seed pixel's normal and point location. Once the size of the region reaches $\frac{N^2}{4}$, the plane equation for the region is calculated from all pixels in the region. If a region's final size is below a threshold ($\mathbf{T_{area}}$ pixels), then the region is discarded (and its pixels are not further considered).

3.1.2 The WSU range segmentation algorithm

The WSU range image segmentation procedure traces its origin to the dissertation work of Hoffman [17], but contains many enhancements incorporated by Flynn [11]. The technique is *not* optimized for polyhedral objects but can accommodate natural quadric surfaces as well. For the experiments described in this paper, the algorithm was modified to accept only first-order surface

fits, but no other special steps were taken to exploit the planar nature of the scenes (surfaces classified as curved are discarded before segmentation results are reported). Prior to any processing, the range points are uniformly scaled to fit within a $5 \times 5 \times 5$ cube. All distance thresholds are in these arbitrary units. The WSU segmenter works as follows:

1. Jump edge pixels are identified by thresholding the maximum change in z between the range pixel of interest and each of its 8-neighbors. If the largest z-deviation is $\mathbf{t_j}$ or greater, the pixel is labeled as a jump edge pixel.

2. Surface normals are estimated at each range pixel with no jump edges in a $\mathbf{k} \times \mathbf{k}$ neighborhood. The estimation procedure performs a principal components fit [10] to the range pixels in the neighborhood and records the principal direction with the lowest variance as the surface normal. This technique accommodates data which is contaminated with noise in all three coordinates.

3. The six-dimensional image formed by concatenating the estimated surface normals to their corresponding pixels is subsampled on a regular grid to yield one thousand or fewer 6-vectors. These vectors are fed to a squared-error clustering algorithm called CLUSTER [18], which finds groupings in the six-dimensional data set based on similarity between the data points. Since these points reflect both position and orientation, the tendency is for CLUSTER to produce clusterings consisting of connected image subsets, with pixels in each cluster having similar orientation. The internal workings of CLUSTER are quite complex. The only user-settable parameter is the maximum number $k_{\max}$ of clusters desired. For these experiments $k_{\max}$ was set to 20. CLUSTER will then produce twenty clusterings (which will correspond to initial segmentations), containing 1 to 20 segments. Clustering statistics are examined to select one clustering for further processing.

4. The selected clustering is converted into an image segmentation by assigning each range pixel to the closest cluster center in the clustering. Connected components are then found to avoid

identical labels for regions that are disjoint in the image. The resulting image is typically an oversegmentation.

5. An edge-based 'domain-independent' merging procedure identifies segments which are adjacent yet have no appreciable change in surface normal across their shared boundary. If the average angle between range pixels on one side of the edge and their neighbors on the other side is less than t_θ (7 degrees in our experiments), the patches are merged. This procedure repeats until no further merging is performed. When range images of polyhedra are processed, this step typically results in a segmentation very close to the final segmentation.

6. Each segment is classified as planar or nonplanar using a regression-based test. The principal components fitting procedure described in step 2 above is applied to all of the pixels in the segment of interest, and the RMS error of fit is calculated. If that error is greater than t_p (0.05 in our experiments), the segment is classified as nonplanar and ignored in further processing (that is, it receives a label of zero).

7. A further merging step joins segments of the same type if they are adjacent and have similar parameters. Specifically, planar segments are joined if their surface normals are within t_a degrees of one another and the distance terms in their implicit equations differ by less than t_d. In our experiments, $t_a = 7$ and $t_d = 0.25$.

8. Unlabeled pixels on the 'frontier' of each segment are merged into it if they fit the segment to a specified accuracy. This step helps to pick up pixels which were dropped from consideration because they were originally mapped to segments classified as nonplanar. For planar segments, a neighboring unlabeled pixel is attached to the segment if its fit error is t_f or less.

9. The above three steps are repeated until the segmentation stabilizes (no change in segment labels during an application of steps, 6, 7, and 8).

Small 'noise' regions can be created by the clustering procedure (due either to outlying range values or to poor estimation of the surface normal). To remove such regions, a simple connected-components procedure identifies and removes all regions with a size lower than N_s pixels, where

N_s equals 20 for each iteration through the classify-merge loop described above, and N_s equals 100 during the final processing.

An additional parameter controlled subsampling for more rapid segmentation. The range images considered in this study were usually four times the size of the images considered in earlier work with this segmenter, and the processing time associated with segmentation of such images rose dramatically. As an easily implementable modification, we added a parameter which identifies the level of subsampling t_x the image undergoes for steps 1 through 5 above. A value of $t_x = 2$ will cause the image to be decimated by two in each direction for the purposes of jump edge detection, normal estimation, subsampling for clustering, initial classification, and domain-independent merging. The first iteration through the classify-merge-grow loop is performed on the low-resolution image; subsequent iterations use the original (the pixels omitted by subsampling are picked up during the first 'grow' step since they are on the frontier of the corresponding segment and are usually picked up at that time).

3.1.3 The UB range segmentation algorithm

This segmenter is based on the fact that, in the ideal case, the points on a scan line that belong to a planar surface form a straight 3D line segment. On the other hand, all points on a straight 3D line segment surely belong to the same planar surface. Therefore, we first divide each scan line into straight line segments and subsequently perform a region growing process using the set of line segments instead of the individual pixels.

The segmentation algorithm for a range image sampled on a regular grid is described in [22]. Since neither the ABW nor the Perceptron range images have this property, the algorithm has been adapted as follows. The first step is a simple split method that recursively divides each scan line into line segments such that the perpendicular distance of the points to their corresponding line segment is within a threshold $\mathbf{T_1}$ (range units). A potential seed region for region growing is a triple of line segments on three neighboring scan lines that satisfies three conditions: (1) all three line segments have at least length $\mathbf{t_1}$ (range units), (2) the overlapping part of two neighboring

line segments has at least $t_2\%$ of the length of each line segment, and (3) every pair of neighboring points on two line segments is within a distance t_3 (range units). The candidate with the largest total line segment length is chosen as the optimal seed region. In the subsequent region growing process, a line segment is added to the region if the perpendicular distance between its two end points and the plane equation of the region is within a threshold $\mathbf{T_2} + t_4 \times size/10000$ (range units) where $size$ is the number of pixels of the region expanded so far. This dynamic threshold relaxes the expansion condition for very large regions. This process is repeated until no more line segments can be added, at which time a new region is started using the next best available seed region. If a region's final size is below a threshold t_5 (pixels), then the region is discarded.

3.1.4 The UE range segmentation algorithm

The UE segmentation algorithm is a region growing algorithm along the lines of the USF segmenter. There are four basic stages which are described as follows:

1. Normal calculation/Data smoothing

   Initial surface normals are calculated at each pixel using a plane fit to the data in a 5 × 5 window. Depth and normal discontinuity detection is performed using simple thresholds between neighboring pixels. The depth threshold is specified in range units, while the normal threshold is in degrees between normal vectors. Following this a discontinuity preserving smoothing is performed on the range data, with multiple passes possible for greater smoothing.

2. Initial H-K based segmentation

   Gaussian (H) and mean (K) curvature are estimated at each pixel using data in a window about it. Pixels can be labelled as belonging to particular surface types (elliptic, planar, etc.) based on the combined signs of the (H,K) values. Each curvature value is classified as Negative, Zero, Positive or Unknown based on the values of "inner" and "outer" thresholds. The inner threshold determines the range of values called Zero. The outer threshold determines the inner limit of the ranges of the Negative and Positive values. Between these values the pixel is labelled as Unknown. Once each pixel is labelled properly with the signs of H

and K, any 8-connected pixels of similar labelling are grouped to form initial regions. This segmentation map is then morphologically dilated and eroded in a specifiable manner to fill small Unknown areas, remove small regions, and separate thinly connected components.

3. Region growing

   For each region in the initial segmentation above a minimal size a least squares surface fitting is performed. Then each region in turn is grown (only planar regions are actually processed in this experiment). Region growing is performed through an iterative expand/refit/contract cycle. For expansion, a pixel is added to the current region if it meets the following requirements: (1) it is 8-connected to the current region, (2) the corresponding 3-D point is within a minimum perpendicular distance to the current surface, (3) the point is closer to the current surface than to the surface for which it may be labelled, (4) the estimated pixel normal is within a minimal agreement with the current surface normal at that position, and (5) the pixel normal is in better agreement with the current surface than with the surface for which it may be labelled. The boundary of the current region is extended in this manner as far as possible. Then the surface is refitted to this new data set. Finally, a contraction of the region boundary is performed. Each pixel is tested using the previous criteria against the new surface estimate. If it is not best accounted for by the new surface, the pixel is returned to the region from which it was originally taken. This expand/contract cycle is iterated until the region boundary stabilizes, or until a maximum iteration limit is reached.

4. Region boundary refinement

   After a single pass through the surfaces, the majority of pixels have been labelled, and only further boundary refinement is usually needed. This is done using the same expand/refit/contract paradigm, but with different criteria for a pixel's inclusion. In this case, a pixel is added to a region during expansion if (1) it is 8-connected to the region, (2) the 3-D point is within the minimum distance of the current surface, (3) the point is on the proper side of a decision

surface. In the case of planes, this surface is another plane passing through the line of intersection between the current plane and the plane corresponding to the current labelling of the pixel. This dividing plane is also chosen to bisect the volume of space between the two planes in question. As in the region growing step, the same criteria are used in the contraction process after surface refitting. Boundary refinement is performed on a complete pass through all of the regions. Additional passes may be performed for additional refinement.

3.2 Parameters selected by training

Each group agreed to explore the parameter space for their segmentation algorithm, once using the training set from the ABW images and once using the training set from the Perceptron images. The results of this step would yield parameter values to be used on the test sets.

3.2.1 Parameters from USF training

There are 5 parameters for this segmenter: $\mathbf{N}$, $\mathbf{T_{angle}}$, $\mathbf{T_{perp}}$, $\mathbf{T_{point}}$ and $\mathbf{T_{area}}$ (see Section 3.1.1). For the ABW imagery, seventy-two different combinations of these parameters were run on the training images (all combinations of N = [17,19,21], T_{angle} = [20.0,25.0,30.0,35.0], T_{perp} = [2.0], T_{point} = [10.0,15.0,20.0] and T_{area} = [100,250]). A table of average metrics for each set of parameters was created by running the compare tool on all ten training images using the compare thresholds [0.51,0.6,0.7,0.75,0.8,0.9,0.95]. The process of selecting the 'best' set of results is to some degree task dependent. For instance, one could desire the highest percentage of correct detection while requiring no under-segmentation, or one could desire any amount of correct detection and over-segmentation while avoiding missed regions, etc. Presumably, the particular needs of a given task would allow one to assign weights to each classification category. In the absence of such weights, we selected the set of results which scored the highest average measure in correct detections. The associated parameters were $N \times N = 21 \times 21$, $T_{angle} = 20.0$, $T_{perp} = 2.0$, $T_{point} = 10.0$ and $T_{area} = 250$.

Similar experiments were conducted on the Perceptron data set, using forty-eight combinations of parameters (N = [17,21], T_{angle} = [20.0,25.0,30.0,35.0], T_{perp} = [4.0], T_{point} = [12.0,16.0] and T_{area}

= [100,250,500]). The range of training values for T_{perp} and T_{point} differ from those used for the ABW imagery because of the difference in quantization (ABW images are 8-bit, Perceptron images are 12-bit). Slight changes were made in the training ranges for N and T_{area} based on the results from the ABW training. The parameters associated with the highest average measure of correct detection were $N \times N = 21 \times 21$, $T_{angle} = 25.0$, $T_{perp} = 4.0$, $T_{point} = 12.0$ and $T_{area} = 500$.

3.2.2 Parameters from WSU training

The WSU segmenter has many parameters, some dealing with the extraction of curved surfaces, and some whose effect on the segmentation quality is minimal for reasonable values. For that reason, we studied those parameters which had the most dramatic and positive effect on the quality of the segmentation results. These crucial parameters were:

1. the subsampling factor t_x,
2. the size k of the neighborhood used in surface normal calculation,
3. the jump edge threshold t_j, and
4. the threshold t_f used to grow planar segments after initial classification.

Initial experiments showed that $t_x = 2$ was an appropriate choice for range images with sizes on the order of 512×512, like those in this study.

Training images from the Perceptron sensor were segmented multiple times, each segmentation corresponding to parameters (t_f, k) drawn from the set [0.1, 0.2, 0.3, 0.4] $\times$ [5, 7, 9, 11]. These experiments yielded 0.4 as the best value of t_f and 7 as the best value of k. The default value of $t_j = 0.2$ was judged adequate for these images. The 'best' segmentations were determined visually and the different parameter values considered usually had a dramatic effect on the visual quality of the result. Likewise, training images from the ABW sensor were segmented multiple times using parameter vector (t_f, k, t_j) drawn from [0.1, 0.2, 0.3, 0.4] $\times$ [5, 7, 9, 11] $\times$ [0.05, 0.1, 0.15, 0.2]. These experiments yielded $t_f = 0.3$, $k = 7$, and $t_j = 0.1$ as the best values.

3.2.3 Parameters from UB training

There are 7 parameters for this segmenter: $t_1, t_2, t_3, t_4, t_5, T_1$ and T_2 (see Section 3.1.3). During training five of the parameters were fixed, namely t_1, t_2, t_3, t_4 and t_5. The other two more critical parameters were tuned based upon the training images. For the ABW images, $t_1 = 4.0, t_2 = 0.1, t_3 = 3.0, t_4 = 0.1$ and $t_5 = 100$. After some tests using arbitrarily chosen values of T_1 and T_2 we localized a good region in the parameter space, namely $R : (T_1, T_2) \in [1...1.5] \times [2...2.5]$. The goodness of this region was verified by two methods. First, nine combinations of parameters, namely $(T_1, T_2) \in [1, 1.25, 1.5] \times [2, 2.25, 2.5]$, were run on the training images and the segmentation results were compared with the ground truth through visual observation and by using the comparison tool. Secondly, tests on 100 randomly chosen parameter pairs within the region R were carried out and the segmentation results were evaluated by the comparison tool. It turned out that within R, the segmenter was very stable. For all 100 test series the average values of the six performance quantities tabulated in this paper (correct detection, angle difference, oversegmentation, undersegmentation, missed, and noise) were similar. As a matter of fact, the standard deviation of these average performance quantities over the 100 tests were 0.1, 0.1, 0.2, 0.1, 0.0, 0.1, 0.1 for an average value of 16.5, 1.6, 1.3, 1.0, 0.8, 1.0, 1.3, respectively. Finally, we selected the mean value of the region R as $T_1 = 1.25, T_2 = 2.25$.

For the Perceptron images, the fixed parameters were $t_1 = 4.0, t_2 = 0.1, t_3 = 3.0, t_4 = 0.2$ and $t_5 = 150$. The other two parameters were tuned to $T_1 = 1.75$ and $T_2 = 3.25$. The test region R in the parameter space was $(T_1, T_2) \in [1.5...2.0] \times [3.0...3.5]$ in this case and over the 100 random tests within R, the standard deviation of the average performance quantities were 0.1, 0.1, 0.1, 0.2, 0.0, 0.1, 0.1 for an average value of 10.6, 2.8, 1.9, 0.9, 0.1, 1.0, 0.5, respectively.

3.3 Parameters from UE training

There are nearly a dozen adjustable parameters for the UE algorithm. Evaluating the training data over a parameter space consisting of ranges in each of these would not have been computationally feasible. Therefore, since the results of intermediate stages are displayed, visual inspection was

used to select appropriate values of the less sensitive parameters, and refined search ranges for the others. The selection of nominal values for the less sensitive parameters was achieved as follows:

1. Depth discontinuity threshold - 15 range units

   By looking at a produced discontinuity map, the threshold was adjusted starting from a value of 5, and incremented by 5, until spurious depth edges were removed from a representative set of images.

2. Normal discontinuity threshold - 180 degrees apart

   Looking at the same maps, a set of values was tested. A large number of spurious edges existed at all normal thresholds due to the image noise level. Therefore all normal discontinuities were ignored with the given threshold rather than introduce false edges.

3. Minimum normal agreement angle for inclusion - 80 degrees

   By examining typical segmentation results, a range of values beginning at 180 degrees was checked, and the chosen value reduced the amount of under-segmentation without creating serious over-segmentation.

4. H-K outer threshold/Plane fit ratio of eigenvalues - Infinity / 0

   Setting these two values to the given values forced the system to process all regions as planar in nature, ignoring any quadric interpretations.

5. Expand/contract iterations - 30

   By examining the typical convergence of region boundaries over the training set, this value was chosen such that it would not cause premature termination.

6. Boundary refinement passes - 3

   This value was also chosen such that it would not interfere with the convergence process over the training set.

The remaining parameters more critically affected the results. In preparation for a search of the parameter space, meaningful ranges were found through visual inspection. By examining the intermediate H-K maps of sample images, ranges for the number of smoothing passes, and the inner threshold on H-K values were determined to give consistent labelings in meaningful regions. Then a set of morphology schedules based on previous experience was found that filtered these labelings to produce even smoother responses. The segmentation results on large regions such as the floor was used to find a viable range for the minimum fitting residual to produce a single region. Checking the final results for the presence of known small regions gave a potential range of values for the minimum region size. The final range of values tested for each of these parameters included:

1. Number of smoothing passes - [2 3]
2. H-K inner threshold - [.005 .006 .007 .008] for Perceptron images, [.011 .012 .013 .014 .015 .016 .017] for ABW images
3. H-K morphology schedule - [dilate/erode/dilate, dilate/erode/dilate/dilate, dilate/erode/dilate/dilate/erode]
4. Minimum fitting residual - [3.0 3.5 4.0 4.5] for Perceptron, [1.5 2.0 2.5 3.0] for ABW
5. Minimum region size - [20 25 30]

The segmentation results were computed at each point in the combined parameter space above. The major criterium used in choosing the best set of parameters was the number of correct classifications at a compare tolerance of $t = 0.8$. Choosing between the leading candidates in this category was done using secondary considerations such as the correct classifications at lower thresholds, and the amount of over/under segmentation. The final values chosen were: 2 smoothing passes, inner H-K thresholds of .006 (Perceptron) and .013 (ABW), an H-K morphology schedule of dilate/erode/dilate/dilate/erode, minimum fitting residuals of 3.5 (perceptron) and 2 (ABW), and a minimum region size of 25.

4 Experimental Results

"Perfect" performance for a segmenter would be correct detection of all regions at a compare tool tolerance of 1.0, with zero angle difference, and of course zero instances of over-segmentation, under-segmentation, missed regions and noise regions. It should be no surprise that we did not find a perfect segmenter. However, the amount of room for improvement might come as some surprise. Figures 2 and 3 show the scores of correct detection for the four segmenters, graphed against the compare tool tolerance. At the weakest tolerance (51%) the segmenters scored between 69% and 89% correct detections on the ABW imagery, and between 40% and 76% on the Perceptron imagery. At a moderate tolerance of 80%, the best scores for correct detections were 88% on the ABW imagery and 68% on the Perceptron imagery. None of the segmenters scored well when the tolerance was moved to 90% or higher.

An "across-the-board winner" in a comparison would have the highest value for average number of correct detections and the lowest value for all the error measures, for the entire compare tool tolerance range. It should come as little surprise that we did not find an across-the-board winner. For instance, the UB segmenter scored highest in correct detections on the Perceptron imagery with a tolerance of 70% and lower, while at a tolerance of 75% and higher the UE segmenter scored highest in correct detections. Table 2 presents the average results on all performance measures for all four algorithms on both test sets at 80% compare tolerance.

Figures 4 through 11 show graphs of the scores of the four segmenters on each data set for the error metrics. Three interesting results appear. First, all four segmenters scored considerably higher measures of missed and noise regions than over- and under-segmentation. Second, over-segmentation is more prevalent than under-segmentation, while missed regions are more prevalent than noise regions. Third, all four segmenters scored worse on all metrics for the Perceptron LRF data than for the ABW structured light scanner data. This last item offers at least some objective confirmation of our subjective impression, acquired in the course of this project, that time-of-flight

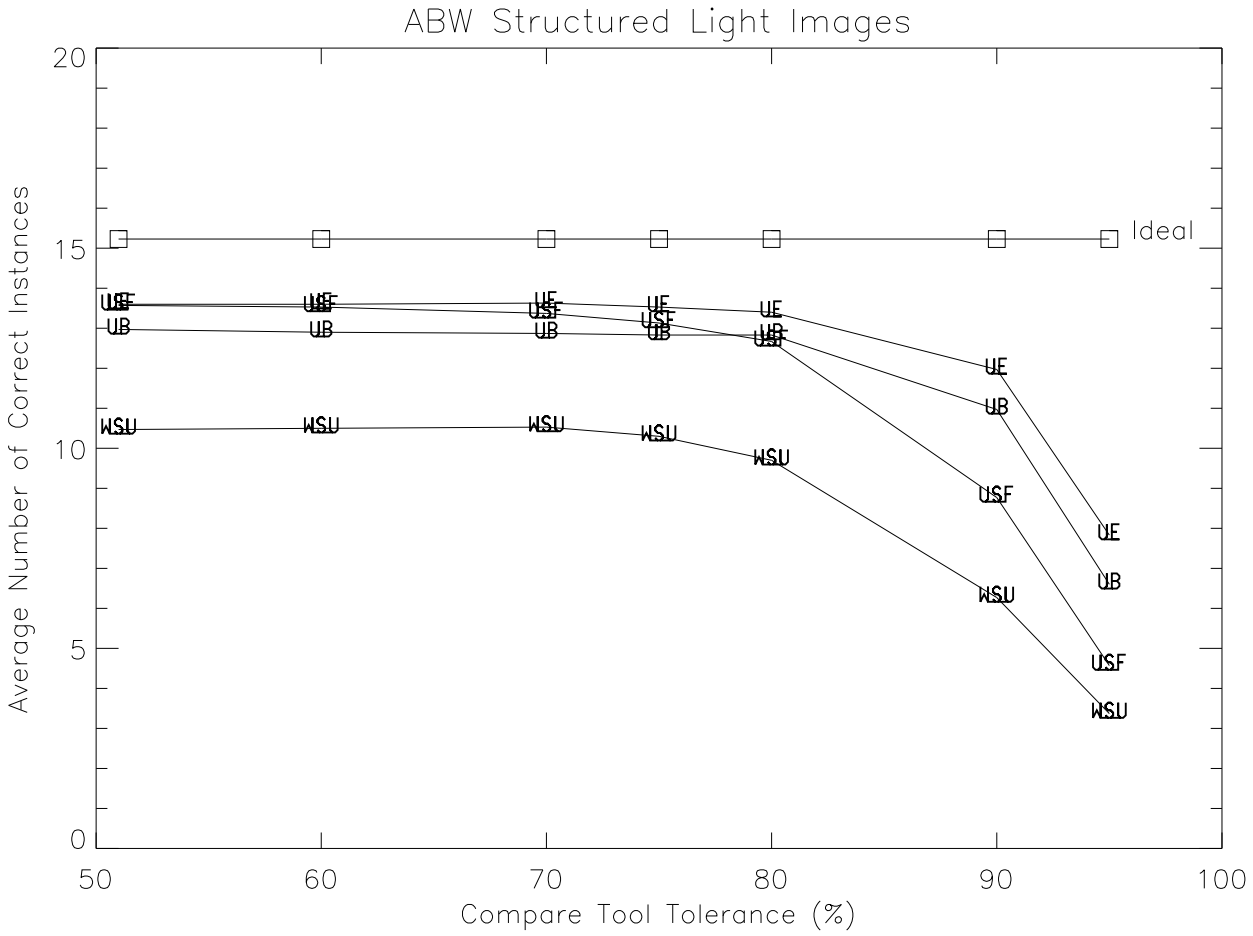


Figure 2: Average correct detections of 4 segmenters (USF, UB, WSU, UE) on 30 ABW test images.

LRF data is "noisier" than structured light scanner data. However, it must be noted that because different objects were imaged with each type of sensor, this observation is not conclusive.

None of the segmenters did worse than 2 degrees average angle difference on the ABW images, or worse than 4 degrees on the Perceptron images. The values of this performance metric were closely bunched for the different segmenters and fairly constant until the threshold T was increased beyond 0.9. At this point the numbers of correct detections diminish dramatically, making this metric less meaningful. Therefore, due to space consideraions, the graphs for this metric were omitted.

The average processing times for the algorithms on the ABW and Perceptron test sets, per image, were 78 and 117 minutes (USF) on a Sun SparcStation 20, 6.3 and 9.1 minutes (UE) on a

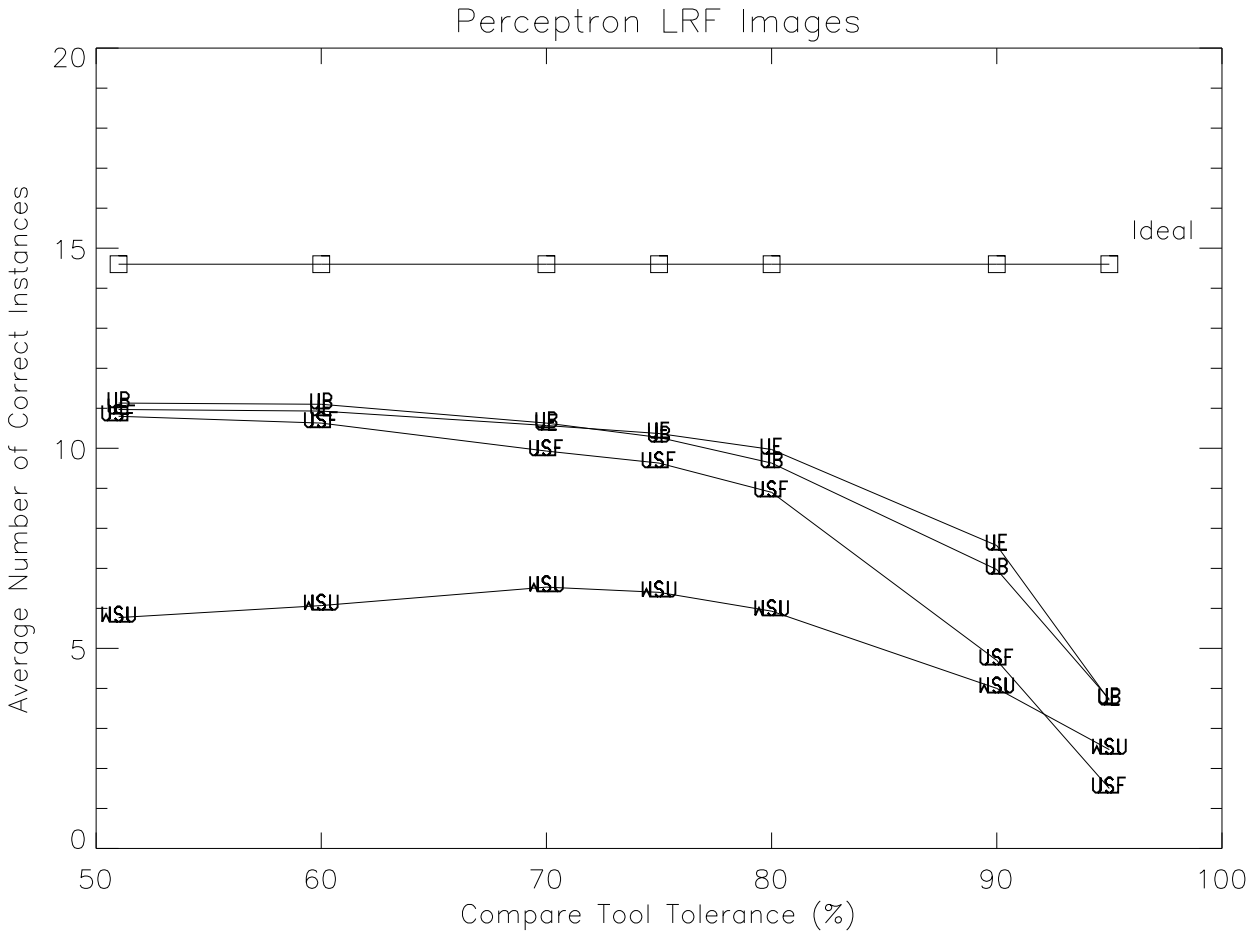


Figure 3: Average correct detections of 4 segmenters (USF, UB, WSU, UE) on 30 Perceptron test images.

Sun SparcStation 5, 4.4 and 7.7 minutes (WSU) on a HP 9000/730, and 7 and 10 **seconds** (UB) on a Sun SparcStation 20. Although the UE segmenter obtains slightly better measures of correct detections than does the UB segmenter, the difference in processing speeds is noteworthy.

5 Discussion

The two major contributions of this work are (1) the development of a rigorous framework for experimental comparison of range image segmentation algorithms, and (2) an assessment of the state of the art for planar segmentation of range images. We feel that the first contribution is of great theoretical and conceptual importance, and hope that by demonstrating a sound experi-

ABW 30 test images

research group	GT regions	correct detection	angle diff. (std. dev.)	over-segmentation	under-segmentation	missed	noise
USF	15.2	12.7	1.6° (0.8)	0.2	0.1	2.1	1.2
WSU	15.2	9.7	1.6° (0.7)	0.5	0.2	4.5	2.2
UB	15.2	12.8	1.3° (0.8)	0.5	0.1	1.7	2.1
UE	15.2	13.4	1.6° (0.9)	0.4	0.2	1.1	0.8

Perceptron 30 test images

research group	GT regions	correct detection	angle diff. (std. dev.)	over-segmentation	under-segmentation	missed	noise
USF	14.6	8.9	2.7° (1.8)	0.4	0.0	5.3	3.6
WSU	14.6	5.9	3.3° (1.6)	0.5	0.6	6.7	4.8
UB	14.6	9.6	3.1° (1.7)	0.6	0.1	4.2	2.8
UE	14.6	10.0	2.6° (1.5)	0.2	0.3	3.8	2.1

Table 2: Average results of all four segmenters on test sets at 80% compare tolerance. Units are instances of region-mappings between ground truth and machine-produced segmentations.

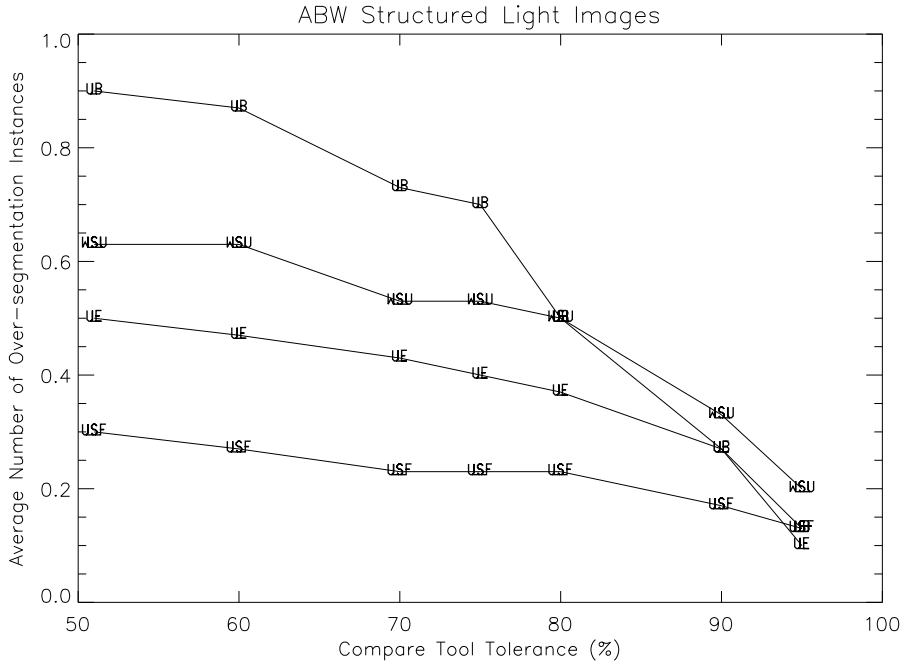


Figure 4: Average over-segmentations (USF, UB, WSU, UE) on 30 ABW test images.

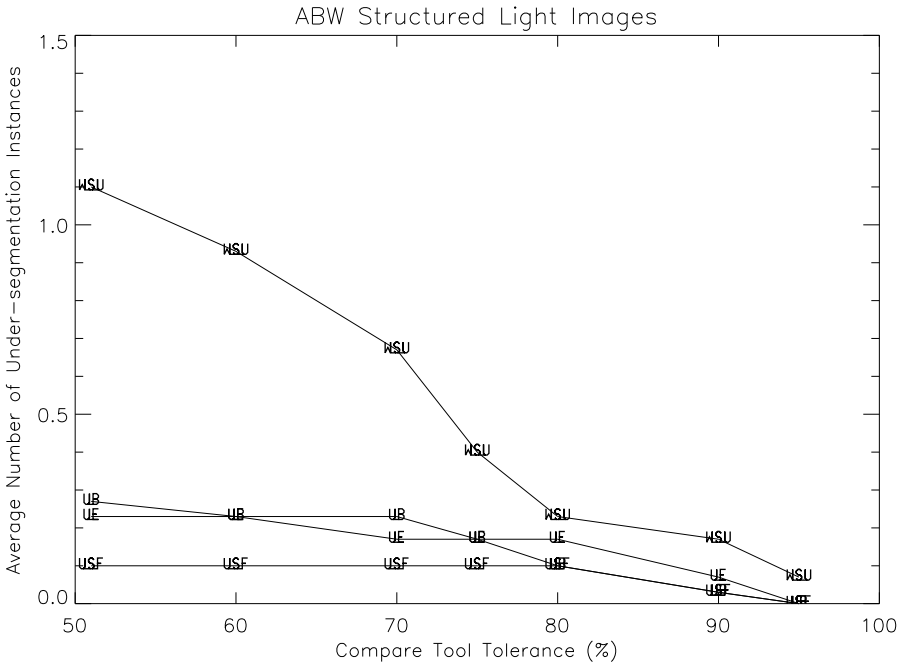


Figure 5: Average under-segmentations (USF, UB, WSU, UE) on 30 ABW test images.

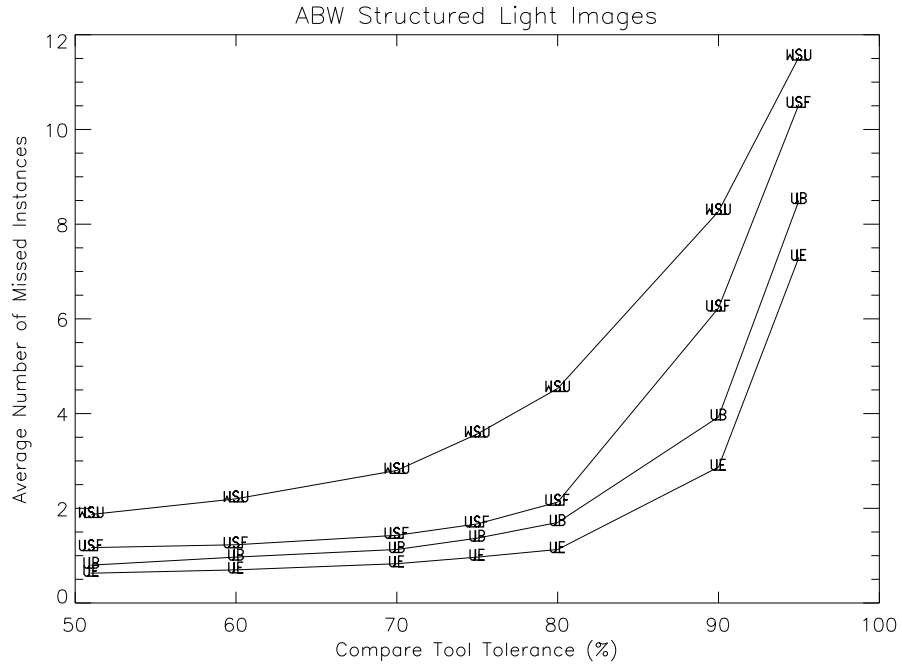


Figure 6: Average missed regions (USF, UB, WSU, UE) on 30 ABW test images.

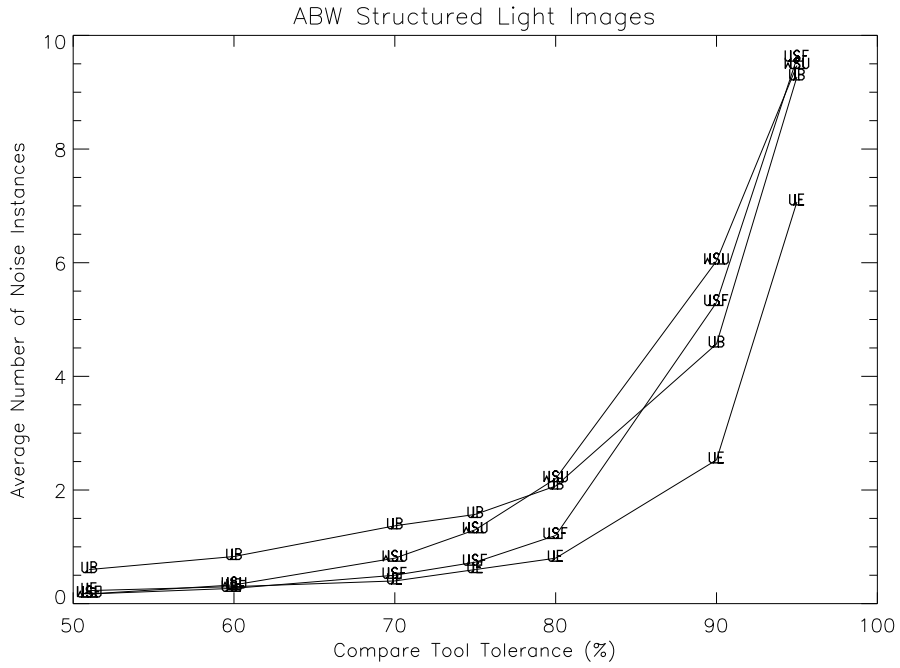


Figure 7: Average noise regions (USF, UB, WSU, UE) on 30 ABW test images.

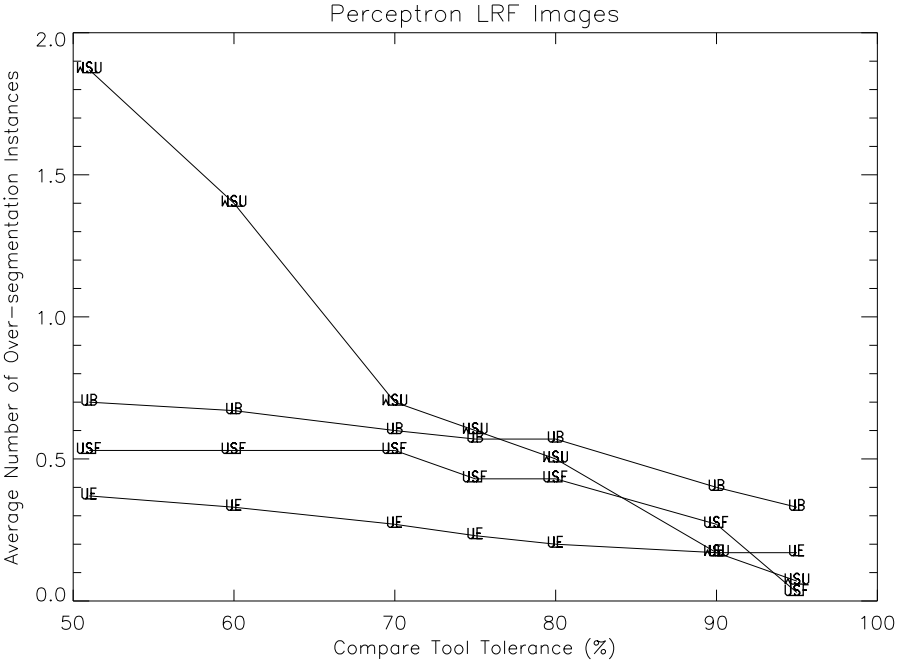


Figure 8: Average over-segmentations (USF, UB, WSU, UE) on 30 Perceptron test images.

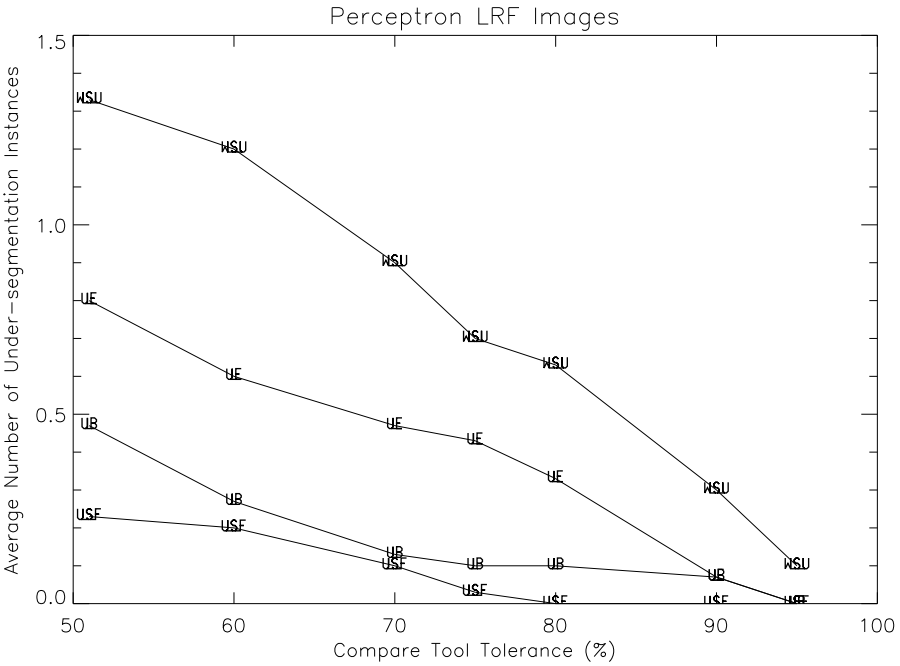


Figure 9: Average under-segmentations (USF, UB, WSU, UE) on 30 Perceptron test images.

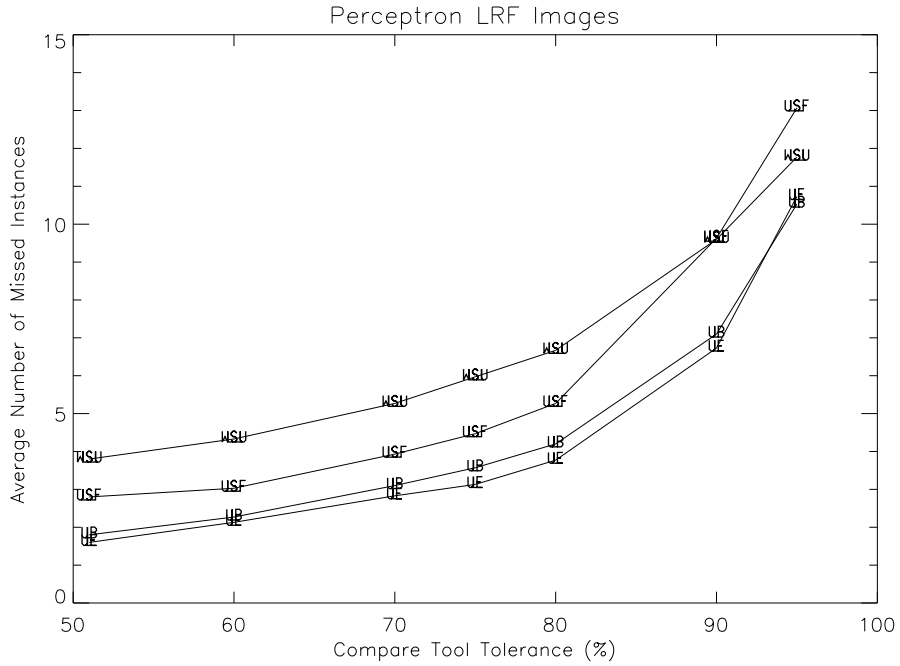


Figure 10: Average missed regions (USF, UB, WSU, UE) on 30 Perceptron test images.

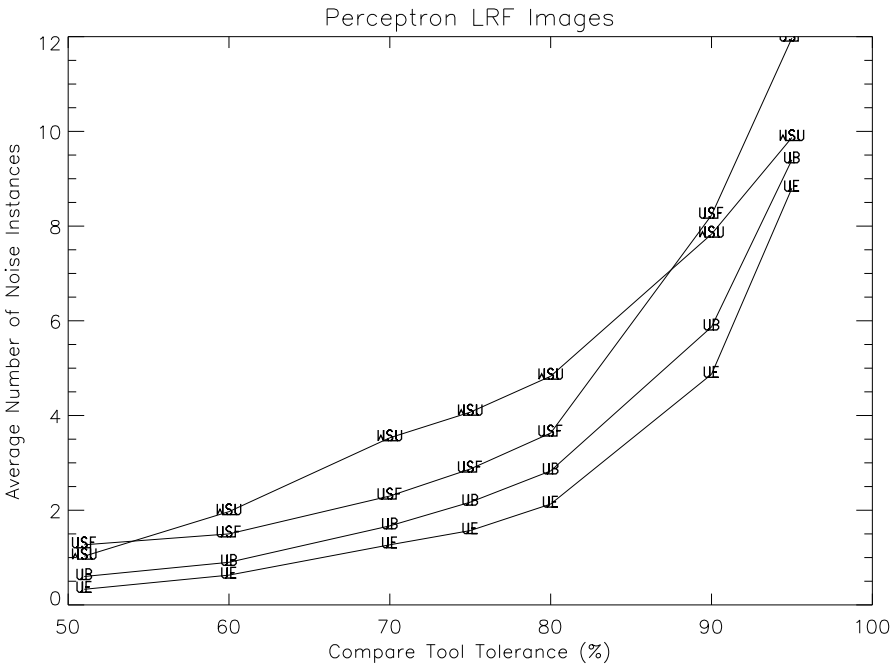


Figure 11: Average noise regions (USF, UB, WSU, UE) on 30 Perceptron test images.

mental framework, we may influence other researchers to perform more work of this type. We feel that the second contribution is of both theoretical and practical importance, largely due to the public availability of the materials involved in this work. These materials should prove valuable to researchers seeking to demonstrate an advance in the state of the art, or to practitioners seeking to utilize a range image segmentation algorithm.

A natural question that arises in reaction to the results presented herein is what specific region properties cause incorrect segmentation, yielding what types of errors? Figure 12 presents bar graphs of all GT regions incorrectly detected by the UB segmenter at an 80% compare tolerance. Each bin corresponds to 10% of the total GT regions, ordered by pixel size. (Graphs for the other three segmenters are similar. We chose to illustrate the UB segmenter by virtue of its speed and performance.) These graphs point out that missed GT regions are predominantly smaller in size than over-segmented GT regions, while under-segmentations generally involve larger and smaller GT regions. Note that this presentation of segmentation errors does not include instances of MS noise regions.

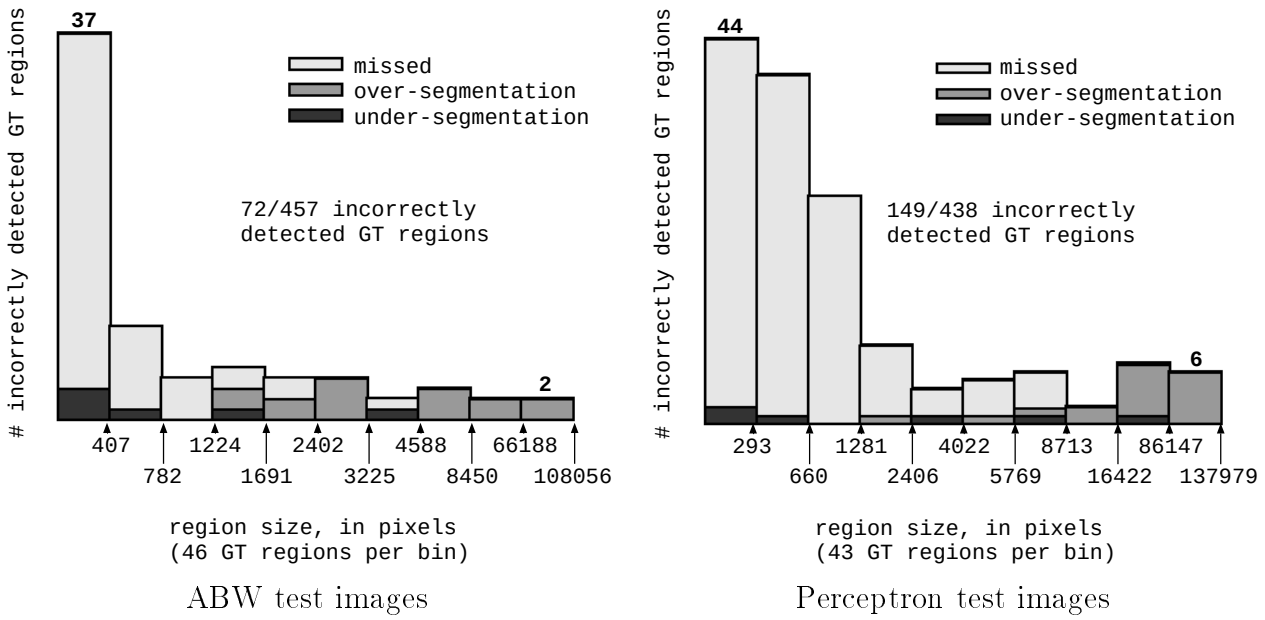


Figure 12: Size distributions of GT regions incorrectly detected by UB segmenter.

One possible perspective is to view these graphs as support for the claim that planar segmentation algorithms are performing "good enough". One can envision a segmentation consumer that is predominantly interested in large regions, and is affected less by errors in small regions. For instance, a greedy matching algorithm might be tuned in this manner. A second perspective is to view these graphs as support for the claim that there is considerable room for improvement in planar segmentation algorithms. It is not difficult to envision a segmentation consumer that can be severely affected by errors in small regions. For instance, our own experiences in CAD-based vision suggest that the geometry of small regions involved in segmentation errors is grossly worse than their large counter-parts. Of course, this entire discussion hinges on the subjectivity of what is considered "small" and "large". Regardless, Figure 12 indicates that segmentation errors occured across the spectrum of GT region size. Perfect performance, even on "large" regions, has not yet been achieved.

We would like to identify what we feel to be the most important open problems in planar patch range image segmentation:

1. Figures 4 through 11 indicate that missed and noise regions occur much more frequently than over- and under-segmentation.

2. Figure 12 illustrates that current segmenters most often miss small regions (on the order of 1000 pixels or less).

3. Figures 2 and 3 illustrate that all segmenters perform poorly when the required tolerance is 90% or higher. This suggests a need for improved refinement on the borders of segmented regions.

Regarding the particular algorithms, we make the following observations. Although the UE segmenter obtained slightly better results than the UB segmenter, the latter performs much faster, probably making it the segmenter of choice for most applications. The USF segmenter guarantees a 4-connected segmentation, which may be essential for some applications (indeed it was a design criteria for related model-building research). Finally, both the UE and WSU segmenters have the capability to segment some classes of curved surfaces.

Figure 13 presents the ABW test image which contains the largest GT region that all 4 segmenters failed to correctly detect at an 80% compare tolerance. The UB and UE segmenters over-segmented the region, while the USF and WSU segmenters missed the region. Figure 14 presents the Perceptron test image which contains the largest GT region that all 4 segmenters failed to correctly detect at an 80% compare tolerance. The UB and WSU segmenters over-segmented the region, the USF segmenter missed the region and the UE segmenter under-segmented the region, Note that results for all 40 images of each type can be viewed on the www site.

We note that we experienced phenomena similar to that reported in the JISCT stereo evaluation [6], in which only three of five research groups completed the testing of their algorithms. During the course of this project we solicited participation from a number of groups. At least four other groups actively looked at participating, but did not complete their evaluation for some reason. Similarly, all of the authors experienced some difficulty in running their algorithm implementations

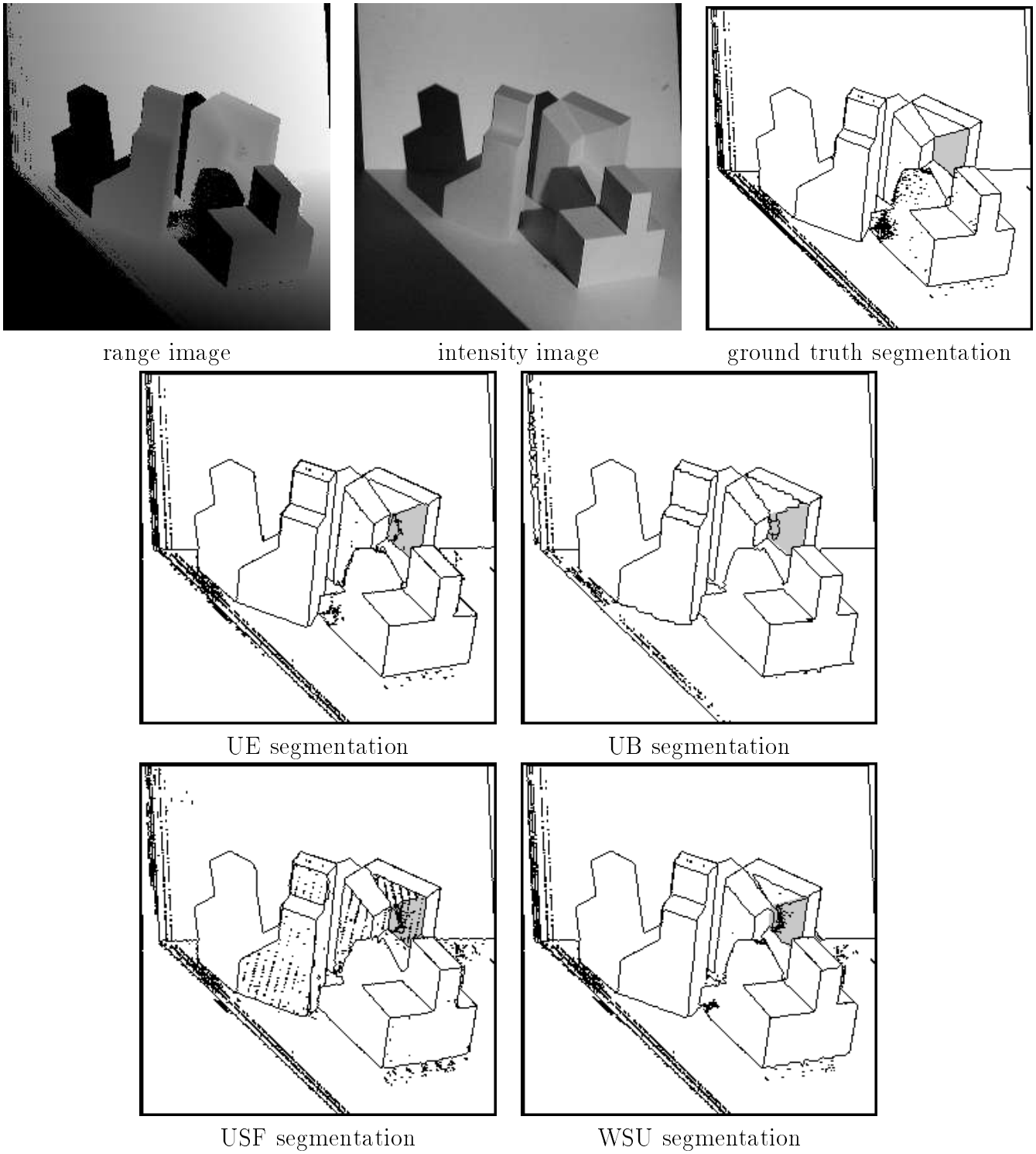


Figure 13: ABW test image #8, which contains the largest GT region (2,960 pixels) that all 4 segmenters failed to correctly detect. The GT region's area is shaded grey in the segmentations. The "specks" were caused by the outlining of isolated noise or unlabeled pixels.

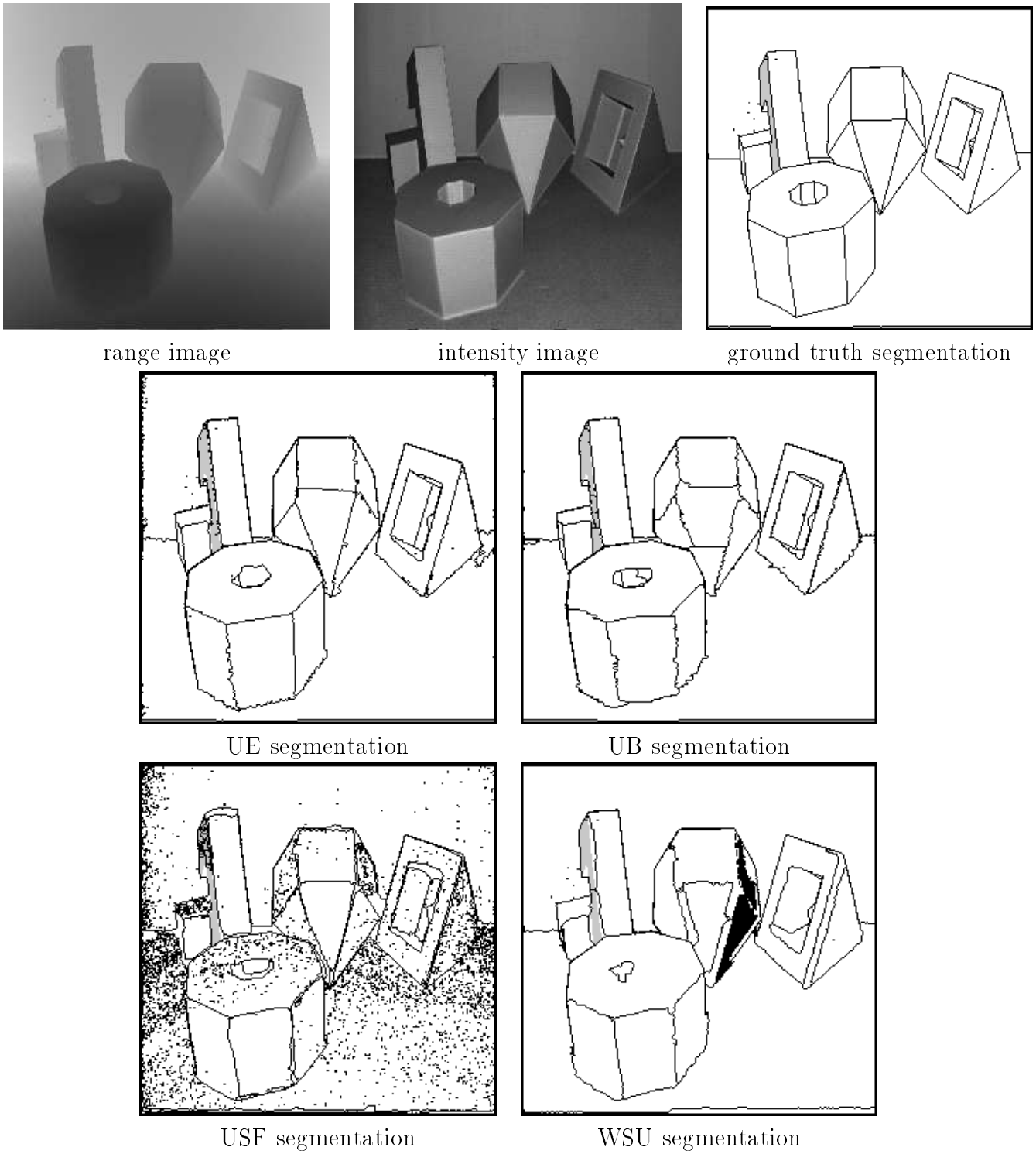


Figure 14: Perceptron test image #26, which contains the largest GT region (2,124 pixels) that all 4 segmenters failed to correctly detect. The GT region's area is shaded grey in the segmentations. The "specks" were caused by the outlining of isolated noise or unlabeled pixels.

on all of the test images. Each group discovered coding errors as well as subtle possible algorithmic improvements. This extended the time required for the evaluation.

As a final note, while we consider the current data set and evaluation methodology useful and broad, it will not capture every possible nuance of the range image segmentation problem. We purposely designed the framework to be flexible to expansion, especially where necessary to bring out certain important aspects of a new algorithm. For instance, curved surfaces represent an obvious potential area of expansion. Similarly, if one felt that some algorithm other than those presented herein would yield higher performance on either the current data set, or some expanded data set, we would welcome an empirical demonstration. We encourage such efforts by leaving all pertinent materials publically available.

6 Acknowledgments

Thanks to the CESAR lab at Oak Ridge National Labs, especially Judd Jones and Ole Henry Dorum, for making it possible for us to acquire images with the Perceptron LRF. The work at USF was supported by AFOSR grant F49620-92-J-0223, NSF grant CDA-92-00369 and a NASA Florida Space Grant Consortium graduate fellowship. The work at WSU was supported by the National Science Foundation under grants CDA-9121675, IRI-9209212, and by the Washington Technology Center. The work at UE was supported by UK EPSRC Grant GR/H/86905.

References

[1] D. H. Ballard and C. M. Brown, Computer Vision, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[2] J. L. Barron, D. J. Fleet and S. S. Beauchemin, "Systems and Experiment: Performance of Optical Flow Techniques", *Int. J. of Computer Vision*, 1994, Vol. 12:1, 43-77.

[3] Paul J. Besl, "Active, Optical Range Imaging Sensors", *Machine Vision and Applications*, 1988, Vol. 1:2, 127-152.

[4] P.J. Besl and R.C. Jain, "Segmentation Through Variable-Order Surface Fitting", *IEEE Trans. on Pattern Analysis & Machine Intelligence*, **10** (2), 1988, pp. 167-192.

[5] S. M Bhandarkar and A. Siebert, "INTEGRA – an integrated system for range image understanding", *Int. J. of Pattern Recognition and Artificial Intelligence* **6** (5), 1992, 913-953.

[6] R. C. Bolles, H. H. Baker and M. J. Hannah, "The JISCT Stereo Evaluation", in *Image Understanding Workshop*, Wash. D.C., 1993, 263-274.

[7] K.L. Boyer, M.J. Mirza and G. Ganguly, "The robust sequential estimator: a general approach and its application to surface organization in range data", *IEEE Trans. on Pattern Analysis and Machine Intelligence* **16** (10), 1994, 987-1001.

[8] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons, New York, 1973.

[9] A.W. Fitzgibbon, D.W. Eggert and R.B. Fisher, "High-level CAD Model Acquisition from Range Images", technical report, Dept. of Artificial Intelligence, Univ. of Edinburgh, 1995.

[10] P.J. Flynn and A.K. Jain, "Surface Classification: Hypothesis Testing and Parameter Estimation," *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '88)*, Ann Arbor, Michigan, pp. 261-267, June 1988.

[11] P.J. Flynn and A.K. Jain, "BONSAI: 3D Object Recognition Using Constrained Search," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **13**, 10, pp. 1066-1075, October 1991.

[12] S. Ghosal and R. Mehrotra, "Segmentation of range images: an orthogonal moment-based integrated approach", *IEEE Trans. on Robotics and Automation* **9** (4), 1993, 385-399.

[13] D. B. Goldgof, T. S. Huang and H. Lee, "A Curvature-Based Approach to Terrain Recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1989, Vol. 11, 1213-1217.

[14] R. C. Gonzalez and R. E. Woods, Digital Image Processing Addison-Wesley, Reading, Massachusetts, 1992.

[15] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, vol. 1, Addison-Wesley, Reading, Massachusetts, 1992.

[16] R. M. Haralick, "Performance characterization in image analysis: thinning, a case in point", *PRL*, vol. 13, pp. 5-12, 1992.

[17] Richard L. Hoffman and Anil K. Jain, "Segmentation and Classification of Range Images", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-9(5):608-620, September 1987.

[18] Anil K. Jain and Richard C. Dubes, Algorithms for Clustering Data, Prentice-Hall, 1988.

[19] R. C. Jain and A. K. Jain, Analysis and Interpretation of Range Images, Springer-Verlag, New York, 1990.

[20] R. C. Jain and T. O. Binford, "Ignorance, Myopia, and Naivete in Computer Vision Systems", *CVGIP*, vol. 53-1, pp. 112-117, 1991.

[21] R. A. Jarvis, "Three dimensional object recognition systems", in Range sensing for computer vision, edited by A. K. Jain and P. J. Flynn, Elsevier Science Publishers, 1993, pp. 17-56.

[22] X. Y. Jiang and H. Bunke, "Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping", *Machine Vision and Applications*, 1994, Vol. 7, No. 2, 115-122.

[23] R. Krishnapuram and S. Gupta, "Morphological methods for detection and classification of edges in range images", *Mathematical Imaging and Vision*, vol. 2, 1992, pp. 351-375.

[24] S.M. LaValle and S.A. Hutchinson, "A Bayesian segmentation methodology for parametric image models", *IEEE Trans. on Pattern Analysis and Machine Intelligence* **17** (2), 1995, 211-217.

[25] S. U. Lee, S. Y. Chung, and R.-H. Park, "A Comparative Performance Study of Several Global Thresholding Techniques for Segmentation", *Computer Vision, Graphics, and Image Processing*, Vol. 52, pp. 171-190, 1990.

[26] M.D. Levine and A.M. Nazif, "An experimental rule based system for testing low level segmentation strategies", in K. Preston and L. Uhr (eds.), *Multicomputers and image processing: Algorithms and programs*, 149-160, Academic Press, 1982.

[27] Martin D. Levine, Vision in Man and Machine, McGraw-Hill, New York, 1985.

[28] S.Z. Li, "Toward 3D vision from range images", *CVGIP: Image Understanding* **55** (3), 1992, 231-260.

[29] Y.W. Lim and S.U. Lee, "On the color image segmentation algorithm based on the thresholding and the fuzzy c-means techqniues", *Pattern Recognition*, 23(9), 935-952, 1990.

[30] S.P. Liou, A.H. Chiu and R.C. Jain, "A parallel technique for signal-level perceptual organization", *IEEE Trans. on Pattern Analysis and Machine Intelligence* **13** (4), 1991, 317-325.

[31] T.S. Newman, P.J. Flynn and A.K. Jain, "Model-based classification of quadric surfaces", *CVGIP: Image Understanding* **58** (2), 1993, 235-249.

[32] N. Pal and S. Pal, "A Review on Image Segmentation Techniques", *Pattern Recognition*, Vol. 26, No. 9, pp. 1277-1294, 1993.

[33] Perceptron Inc., "LASAR Hardware Manual", *23855 Research Drive, Farmington Hills, Michigan 48335*, 1993.

[34] K. Price, "Anything You Can Do, I Can Do Better (No You Can't)..." *CVGIP*, Vol. 36, pp. 387-391, 1986.

[35] B. Sabata, F. Arman and J.K. Aggarwal, "Segmentation of 3D range images using pyramidal data structures", *CVGIP: Image Understanding* **57** (3), 1993, 373-387.

[36] T. G. Stahs and F. M. Wahl, "Fast and Robust Range Data Acquisition in a Low-Cost Environment", in *SPIE #1395: Close-Range Photogrammetry Meets Mach. Vis.*, Zurich, 1990, 496-503.

[37] R.W. Taylor, M. Savini and A.P. Reeves, "Fast segmentation of range imagery into planar regions", *Computer Vision, Graphics and Image Processing* **45** (1), 1989, 42-60.

[38] E. Trucco and R.B. Fisher, "Experiments in curvature-based segmentation of range data", *IEEE Trans. on Pattern Analysis and Machine Intelligence* **17** (2), 1995, 177-182.

[39] M.A. Wani and B.G. Batchelor, "Edge-Region-Based Segmentation of Range Images", *IEEE Trans. on Pattern Analysis and Machine Intelligence* **16** (3), 1994, pp. 314-319.

[40] N. Yokoya and M.D. Levine, "Range image segmentation based on differential geometry: a hybrid approach", *IEEE Trans. on Pattern Analysis and Machine Intelligence* **11** (6), 1989, 643-649.

[41] Handbook of Pattern Recognition and Image Processing: Computer Vision, edited by Tzay Y. Young, Chapter 7, Academic Press, 1994.

[42] X. Yu, T. D. Bui and A. Krzyzak, "Robust Estimation for Range Image Segmentation and Reconstruction", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, May 1994, Vol. 16, No. 5, pp. 530-538.

[43] R. Zhang, P. Tsai, J. Cryer and M. Shah, "Analysis of Shape from Shading Techniques", in *Computer Vision and Pattern Recognition*, Seattle, Washington, 1994, 377-384.

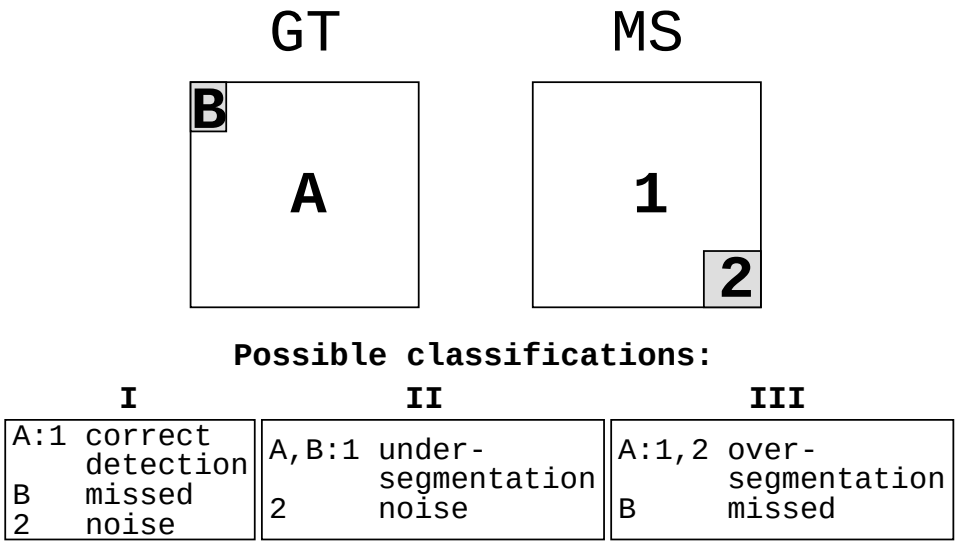


Figure 15: An example where multiple region classifications could be given.

A Proof of possibilities for multiple classifications

Although the metric definitions given in Section 2.4 result in a classification for every region in the GT and MS images, they are not unique for $T < 1.0$. Figure 15 demonstrates this. Assume that region A in the GT image and region 1 in the MS image overlap each other at least T percent of their respective areas. Then we would deduce that region A in GT and region 1 in MS are an instance of correct detection. This leaves B in GT classified as missed, and 2 in MS classified as noise (case **I** in Figure 15). However, if regions A and 1 mutually overlap at least T percent of their respective areas, then the union of regions A and B in GT and region 1 in MS would also overlap at least T percent of their respective areas. This satisfies the under-segmentation classification metric, leaving 2 in MS classified as noise (case **II** in Figure 15). Similarly, the mapping of region A in GT to the union of regions 1 and 2 in MS would yield an over-segmentation classification, leaving B in GT classified as missed (case **III** in Figure 15).

However, for $0.5 < T < 1.0$ any region can at most contribute to three classifications, one each of correct detection, over-segmentation and under-segmentation. First, consider the definition of a correct detection classification. It states that at least T percent of a GT region's pixels must overlap some MS region. This implies that only $1.0-T$ percent of the GT region's pixels can overlap any other MS region. Since $T > 0.5$, $1.0 - T$ clearly cannot also be greater than T. Therefore no other MS region can overlap the GT region sufficiently to create another correct detection classification for the GT region. This argument applies similarly for any MS region in a correct detection classification.

Now consider the definition of an over-segmentation classification. It states that for a set of MS regions to contribute to the mapping, each MS region in the set must overlap by at least T percent of its pixels the candidate over-segmented GT region. Therefore, because $T > 0.5$, each MS region can be considered in at most one mapping of over-segmentation. In the other direction, if the union of the set of MS regions overlaps the GT region by at least T percent of its pixels, then once again there is not enough left of the GT region to use in another over-segmentation mapping.

Finally, there is the possibility of considering subsets of the total possible set of MS regions that could contribute to the mapping. However, any subset causes the percentage of the GT region which is covered to be lowered. If we require the maximum possible covering (where each MS region still satisfies the metric), then we require the total set. Hence, each GT region can be considered in at most one over-segmentation mapping. Reversing the direction of arguments in this discussion between GT and MS regions proves the same for an under-segmentation mapping.