Fast Range Image Segmentation Using High-Level Segmentation Primitives

X. Y. Jiang
Dept. of Computer Science
Univ. of Bern, Switzerland
jiang@iam.unibe.ch

U. Meier
GfAI, Herrenschwanden
Switzerland

H. Bunke
Dept. of Computer Science
Univ. of Bern, Switzerland
bunke@iam.unibe.ch

Abstract

In this paper we present a novel algorithm for very fast segmentation of range images into both planar and curved surface patches. In contrast to other known segmentation methods our approach makes use of high-level features (curve segments) as segmentation primitives instead of individual pixels. This way the amount of data can be significantly reduced and a very fast segmentation algorithm is obtained. The proposed algorithm has been tested on a large number of real range images and demonstrated good results. With an optimized implementation our method has the potential to operate in quasi real-time (a few range images per second).

1 Introduction

Today range scanners are able to acquire high-resolution and high-quality range images in (quasi) real-time [1]. The range scanner described in [16], for instance, needs only 0.3 seconds to acquire a range image of 512 × 256 pixels. With 30 images of 420 × 512 pixels per second, the range scanner developed by Beraldin *et al.* [2] is even possible to operate in real-time. On the other hand, however, range image analysis and interpretation in a robust and fast way is still an open research topic. Here one of the most important problems is that of segmentation.

Generally speaking, the segmentation algorithms known from the literature can be classified into general and dedicated approaches. In an approach of the former category, general (typically polynomial) surface functions are used to compute a complete segmentation and reconstruction [3, 11]. In contrast, dedicated approaches search for particular structures in range data, such as planes, cylinders, cones or solids of revolution [8, 12, 17, 18]. In this case the specific properties of the surfaces under consideration can be fully utilized. In the present paper we consider the segmentation of range images into quadrics of the form

$$\sum_{i+j+k\leq 2} a_{ijk}x^i y^j z^k = 0 \qquad (1)$$

and our method thus falls into the category of general approaches.

Segmentation of range images is considered a difficult problem. A recent experimental comparison[1] [6, 7] demonstrated that even the seemingly simple task of planar range image segmentation cannot be regarded as solved. There is still considerable room for improvement with respect to both quality and speed. Naturally, segmentation into more complex surface structures is even less matured.

We believe that one major problem in achieving fast range image segmentation is due to the use of individual pixels as segmentation primitives. In our efforts to make use of high-level segmentation primitives instead of individual pixels, we have proposed a fast and robust algorithm for segmenting a range image into planar regions based on a scan line grouping technique [8]. In the experimental comparison described in [6, 7], this technique has demonstrated not only a very high computational efficiency, but also good segmentation results. The present paper reports the extension of our earlier algorithm to handling both planar and curved surfaces, retaining the advantage of very high speed. Although this new algorithm is based on the same scan line grouping idea, it is not a straightforward generalization of the former method and requires a number of essential modifications and extensions.

Because of the general difficulties in dealing with the quadrics defined in Eq. (1), our segmentation algorithm is divided into two phases. We begin with an

[1]This comparison is based on a number of objective performance metrics and two large range image sets acquired by a laser range finder and a structured light scanner, respectively, that have manually specified ground truth.

initial segmentation into quadratic surface patches:

$$z = f(x,y) = \sum_{i+j\leq 2} a_{ij}x^i y^j \qquad (2)$$

In a second phase, these surface patches are merged into quadrics.

The rest of this paper is organized as follows. First, we give a description of the fundamental scan line grouping technique. Then, the actual implementation for the segmentation task under consideration and the two phases of our segmentation algorithm are presented. This is followed by experimental results. Finally, some discussions conclude the paper.

2 Scan line grouping technique

We assume a dense range image $z(x,y)$ regularly sampled in at least one of the coordinate directions. Notice that this assumption is satisfied by many range scanners. A special case is range images sampled on a regular grid [14], i.e., in both coordinate directions. In the subsequent discussion we will assume that the y-axis be the direction of regular sampling.

Let $z = f(x,y)$ be the surface of an object to be segmented. Then, that part of the surface on a scan line $z(x,y_0)$ with a constant y_0 is simply a two-dimensional curve $z = f(x,y_0)$ in the $x-z$ plane. For example, a three-dimensional plane

$$z = ax + by + c$$

results in a straight line

$$z = ax + \hat{b}, \quad \hat{b} = by_0 + c$$

on the scan line $y = y_0$. Generally, curve segments $z = f(x,y_0)$ corresponding to different surfaces have different function parameters. So, a simple and powerful grouping technique is to partition a scan line into a set of curve segments. On the other hand, all points on a smooth curve segment definitely belong to the same surface patch. Instead of individual pixels, we can thus treat a complete curve segment as an atomic entity and consider the set of curve segments as high-level segmentation primitives.

For the development of a fast algorithm the following three aspects are of crucial importance:

- the number of segmentation primitives should be small,
- the access to the neighborhood of segmentation primitives should be efficient, and
- the handling costs of segmentation primitives during the segmentation process should be small.

With individual pixels as segmentation primitives the last two requirements are satisfied, but the main bottleneck lies in the huge number of pixels. Using curve segments, on the other hand, the number of segmentation primitives can be significantly reduced. As will be discussed later in this paper, our method fulfills the other two requirements, too. Thus, the use of curve segments provides a sound basis for an efficient segmentation algorithm.

3 Extraction and representation of curve segments

On a scan line $y = y_0$, a quadratic surface according to Eq. (2) generates a quadratic curve of the form

$$z = ax^2 + bx + c.$$

As a preprocessing step for the actual segmentation, we therefore divide each scan line into a set of quadratic curve segments. For this purpose we have adapted the classical splitting algorithm proposed by Duda and Hart [4]. A quadratic approximation function is first determined for a whole scan line based on the midpoint and the two endpoints. Then, whenever the largest error $e_{\max}$ between the approximation function and the scan line is greater than a preset threshold, the scan line is split into two parts. The splitting is performed at the location where $e_{\max}$ occurs. The splitting algorithm proceeds recursively until the approximation error $e_{\max}$ doesn't exceed the threshold.

We denote a curve segment by $(y_0, (x_1, x_2), g(x))$ where y_0 is the y-coordinate of the underlying scan line, x_1 (x_2) is the x-coordinate of the leftmost (rightmost) pixel of the curve segment and $g(x)$ represents the related quadratic curve function. All curve segments of a scan line are stored in a double-linked list in their natural order. In addition we use an array, each entry y_0 of which points to the first curve segment of the scan line $z(x,y_0)$.

Among other factors, a simple and efficient access to the neighborhood of segmentation primitives has an essential influence on the performance of a segmentation algorithm. In a regular image grid this access is guaranteed in a natural way. For instance, the 8-neighborhood of a pixel (x,y) is simply the set $\{(x\pm 1, y\pm 1)\} - \{(x,y)\}$. However, by using curve segments as segmentation primitives, this natural neighborhood relationship is lost and we need some additional pointers to establish the neighborhood relationship.

Two curve segments s and s' are said to be neighbors if there exist points $p \in s$ and $p' \in s'$ such that p and p' are neighbors in a 4-neighborhood. Clearly, each curve segment $s = (y_0, (x_1, x_2), g(x))$ has exactly one left neighbor (except the first segment of a scan line) and one right neighbor (except the last segment of a scan line). These neighbors are simply the left and right neighbor in the double-linked list. The number of neighbors of s in the scan lines $z(x, y_0 \pm 1)$, however, is not constant. We note that a segment $s' = (y_0 \pm 1, (x_1', x_2'), g'(x))$ is a neighbor of s if and only if $[x_1, x_2] \cap [x_1', x_2'] \neq \emptyset$ holds. This condition can be easily tested. Moreover, if s' and s'' on the same scan line are both neighbors of s, then all curve segments between s' and s'' are also neighbors of s. Thus, all neighbors of s in the scan line $z(x, y_0 \pm 1)$ build a continuous sequence in the double-linked list, and we need only two pointers to the first and last segment of this sequence. (Note that the first and last segment can be the same.) Totally, six pointers – tow for the actual scan line and two for the scan line above and below, respectively – are used for each curve segment to establish the neighborhood relationship. By this data structure, the neighbors of a curve segment can be found by simply following the pointers. Thus a fast access to the neighborhood of a curve segment is achieved.

4 Region growing using curve segments

The actual segmentation is based on region growing. Basically, it consists of a seed extraction step followed by iterative region growing. These two steps are repeated until no more seeds can be found.

In our case a seed contains a single curve segment s_0 and a quadratic surface function hypothesis $z = f(x, y)$. A simple strategy has been followed to establish seeds. We always choose the longest unused curve segment s_0. The surface function is hypothesized based on s_0 and the two neighboring curve segments above and below s_0, respectively, that have the largest overlap with s_0 among all neighboring segments of s_0 in the respective scan line.

The region growing step takes place around the seed curve segment s_0. A segment $s : (y_0, (x_1, x_2), g(x))$ adjacent to the current region R is tested against the surface function $z = f(x, y)$ in the following way to decide whether s is added to R. If s belongs to R, then $g(x)$ should have the expected form $f(x, y_0)$ in the ideal case. We measure the actual difference between the expected curve $f(x, y_0)$ and the actual curve $g(x)$ by means of the maximal and average difference:

$$d_{\max} = \max\{|f(x, y_0) - g(x)|,\ x \in [x_1..x_2]\}$$

$$d_{\text{avg}} = \frac{1}{|x_2 - x_1|} \int_{x_1}^{x_2} |f(x, y_0) - g(x)| dx$$

Moreover, the angle difference between the normal vectors at the midpoint $\overline{x} = (x_1 + x_2)/2$ of s and the expected curve $f(x, y_0)$,

$$\cos^{-1} \frac{(-g_x(\overline{x}), 1) \cdot (-f_x(\overline{x}, y_0), 1)}{||(-g_x(\overline{x}), 1)|| \cdot ||(-f_x(\overline{x}, y_0), 1)||}$$

is computed. The curve segment s is merged to R only if all three difference terms are within a respective tolerance value. Notice that we can compute these quantities efficiently in an analytical manner.

There is always some uncertainty in choosing a seed for region growing. In our case we have tried to minimize this uncertainty by using the longest segment. While this simple strategy works well in many cases, there is no guarantee that we always find an appropriate seed for the region growing process. Consequently, it may happen that the hypothesized seed cannot be reasonably expanded. To remedy this problem, we accept a region only if it is larger than a minimal size. Otherwise, the region is cancelled and the seed curve segment is specially marked so that it will not be used later for this purpose.

5 Quadrics approximation

A number of surfaces like cylinder and sphere cannot be reasonably represented by quadratic functions. As a consequence, the region growing procedure usually generates an over-segmentation for such surfaces. One potential solution to this problem is to use polynomial functions of higher degree [3]. In our work we have chosen quadrics of the form (1) as more powerful surface function. Based on the regions found by the region growing procedure described above we test pairs of neighboring regions. For each pair we compute a quadric function of the union of the two involved regions. If the pair is well approximated by the quadric, i.e., the maximal and average approximation error are within a respective threshold, then they are merged. For safety we take only those pairs of regions into account where there is a smooth boundary between the two regions. For a boundary we consider all pairs of neighboring points $P_1(x_1, y_1, z_1)$ and $P_2(x_2, y_2, z_2)$, one from each region, represented by the quadratic function $z = f_k(x, y)$, $k = 1, 2$. Then, the angle difference between the local normals at P_1 and P_2 can be computed analytically. A boundary is regarded smooth if the average angle difference for all point pairs along the boundary is smaller than a threshold.

Generally, the merge operation described above is computationally relatively expensive because of the quadric function approximation and the determination of the approximation error terms. Especially in the latter case we leave the level of curve segments and go back to the individual pixels. Fortunately, the merge operation is only carried out on a small number of quadratic surface patches resulting from the first phase of our algorithm. Moreover, the boundary test described above allows a further reduction of the possible candidates.

6 Experimental results

We have implemented the segmentation algorithm in C++ on a Sun Sparcstation 20. It has been tested on a large number of real range images and demonstrated good results. Concretely, we have used 141 range images from the popular range image set (test set 1) maintained by Patrick Flynn [5] and 38 other range images (test set 2) described in [10]. Both image sets were acquired at Michigan State University using a Technical Arts scanner. For each image in test set 2 an additional intensity image in full registration is provided. For all test images, our segmentation method used the same parameters determined experimentally. Figure 1 shows the segmentation results for two range images from test set 1 where different grey levels are used to color the regions so that no two neighboring regions get the same grey level. The two images contain mainly curved surfaces (top) and planar surfaces (bottom), respectively, and illustrate that our segmentation algorithm is able to handle both types of surfaces in a unified framework. The average segmentation time for the 141 range images of test set 1 is 0.7 seconds. Note that the computation time given in this section includes all steps of our segmentation algorithm, i.e., extraction of curve segments, determination of the relationships between the curve segments, and region growing. For the two range images shown in Figure 1, 1.1 and 0.9 seconds were needed, respectively. Another segmentation result for a range image from test set 2 is presented in Figure 2. The computation time amounts to 0.5 seconds. In average 0.7 seconds are needed for the images of test set 2. Finally, Figure 3 demonstrates the two phases of our segmentation algorithm for a range image from test set 1. An over-segmentation resulting from the first phase can be successfully resolved by the subsequent quadrics approximation.

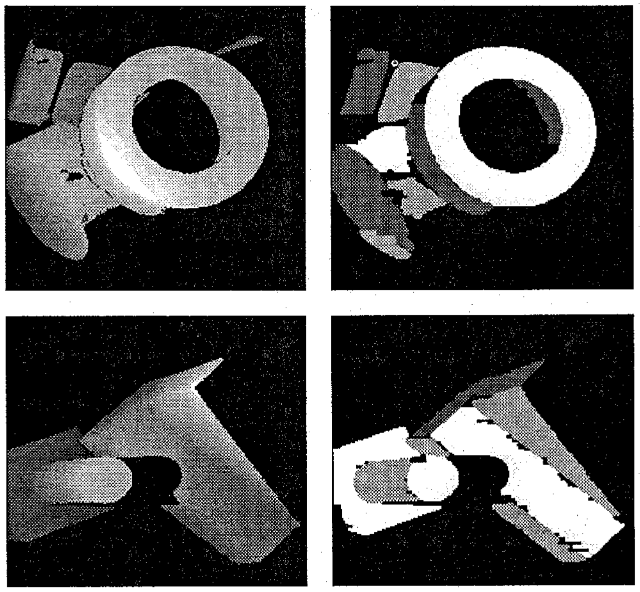
Figure 1: Two range images from test set 1 (left) and their segmentation result (right).

A (qualitative) visual assessment reveals that the segmentation results achieved by our algorithm are comparable to those reported in the literature on the same test image set (see for instance [15]). However, our algorithm is unique in its high computational efficiency. As a comparison example, the segmentation method described in [19] requires for range images of comparable complexity more than 60 seconds on a Silicon GTX 220 Workstation. No direct comparison of computation efficiency with [15] is possible since no actual segmentation time was given there. So far we have not paid much attention to the optimization of the program code. Also, the scan line approximation can be done in parallel. Thus, we expect that the segmentation method described in this paper will finally be able to operate in quasi real-time (a few range images per second).

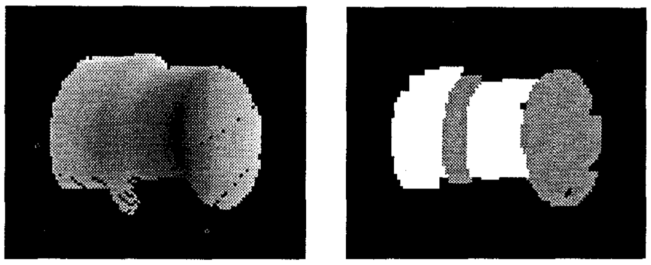
Figure 2: A range image from test set 2 (left) and its segmentation result (right).

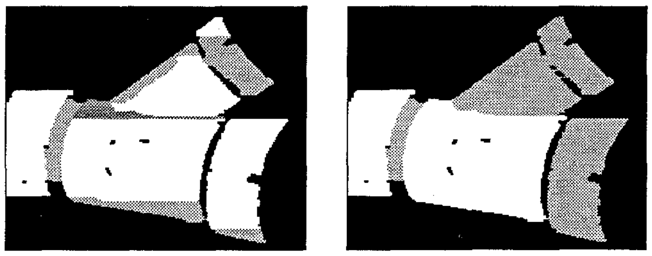
Figure 3: Segmentation results of a range image from test set 1 after the first (left) and second phase (right) of our segmentation algorithm.

7 Discussions

We have proposed a novel algorithm for very fast segmentation of range images into both planar and curved surface patches in a unified framework. In contrast to other known segmentation methods our approach makes use of high-level features (curve segments) as segmentation primitives instead of individual pixels. This scan line grouping technique significantly reduces the amount of data the segmentation process is faced with. The first range image in Figure 1, for instance, has 18096 valid pixels, but only 962 curve segments. Hence, a reduction ratio of 19 has been achieved[2]. As high-level segmentation primitives, on the other hand, the curve segments can be easily handled and aren't computationally more costly.

The robustness of our segmentation algorithm has been demonstrated by applying it with identical parameter values to a large number of real range images. Further improvements can be expected through two combination strategies. In our current implementation we have applied the scan line grouping technique to the image rows. But we could also proceed by exchanging the role of the image rows and columns. This way two segmentation results can be obtained. A combination of both may be useful to achieve a more reliable segmentation. From the general point of view, the combination of results achieved by different segmentation methods is worth investigation and has been widely ignored in the literature so far.

Recently, we have developed a fast edge detection and description method [9] that precisely marks jump and crease edge points in range images and also supplies a rich description of edge structures, including detailed edge types and the local surface characteristics. Edge information of this kind should be able to provide additional guidance to region-based segmentation schemes. This second combination strategy is currently under study.

[2]As a matter of fact, the scan line grouping technique provides the basis for a powerful image compression method. Pham and Abdollahi [13] reported such an approach by partitioning image rows into straight line segments. Our interest here, however, lies not in efficient storage and transmission of pictorial information but in an image representation which allows fast segmentation.

Acknowledgements

We thank the Pattern Recognition and Image Processing Lab of Michigan State University of making their database of range images for public usage.

References

[1] C.M. Bastuscheck, Techniques for real-time generation of range images, Proc. of CVPR, 262–268, 1989.

[2] J.A. Beraldin, M. Rioux, F. Blais, L. Cournoyer, J. Domey, Registered intensity and range imaging at 10 mega-samples per second, Optical Engineering, 31(1), 88–94, 1992.

[3] P.J. Besl, Surfaces in range image understanding, Springer-Verlag, 1988.

[4] R.O. Duda, P.E. Hart, Pattern classification and scene analysis, Wiley, New York, 1972.

[5] P.J. Flynn, A.K. Jain, BONSAI: 3-D object recognition using constrained search, IEEE Trans. on PAMI, 13(10), 1066–1075, 1991.

[6] A. Hoover, G. Jean-Baptiste, X.Y. Jiang, P.J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, Range image segmentation: The user's dilemma, Proc. of Int. Symposium on Computer Vision, Coral Gables, 323-328, 1995.

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

[8] X.Y. Jiang, H. Bunke, Fast segmentation of range images into planar regions by scan line grouping, Machine Vision and Applications, 7(2), 115–122, 1994.

[9] X.Y. Jiang, H. Bunke, Robust and fast edge detection and description in range images, Proc. of IAPR Workshop on Machine Vision Applications, Tokyo, 1996.

[10] G.C. Lee, G.C. Stockman, Obtaining registered range and intensity images using the Technical Arts Scanner, Technical Report CPS-91-08, Dept. of Computer Science, Michigan State University, East Lansing, 1991.

[11] A. Leonardis, A. Gupta, R. Bajcsy, Segmentation of range images as the search for geometric parametric models, Int. Journal of Computer Vision, 14, 253–277, 1995.

[12] T. Lozano-Pérez, W.E.L. Grimson, S.J. White, Finding cylinders in range data, Proc. of IEEE Conf. on Robotics and Automation, 202–207, 1987.

[13] D.T. Pham, M. Abdollahi, Image compression using polylines, Pattern Recognition, 21(6), 631–637, 1988.

[14] P. Saint-Marc, J.-L. Jezouin, G. Medioni, A versatile PC-based range finding system, IEEE Trans. on Robotics and Automation, 7(2), 250–256, 1991.

[15] N.S. Raja, A.K. Jain, Obtaining generic parts from range images using a multi-view representation, CVGIP: IU, 60(1), 44–64, 1994.

[16] Y. Sato, M. Otsuki, Three-dimensional shape reconstruction by active rangefiner, Proc. of CVPR, 142–147, 1993.

[17] L. Shao, R. Volz, Finding cones from multi-scan range images, Proc. of Intelligent Robots and Computer Vision X: Neural, Biological and 3-D Methods, SPIE Vol 1608, 378–384, 1991.

[18] N. Yokoya, M.D. Levine, Volumetric shapes of solids of revolution from a single-view range image, CVGIP: IU, 59(1), 43–52, 1994.

[19] X. Yu, T.D. Bui, A. Krzyzak, Robust estimation for range image segmentation and reconstruction, IEEE Trans. on PAMI, 16(5), 530–538, 1994.