LETTER *Special Issue on Machine Vision Applications*

Finding Line Segments with Tabu Search

Concettina GUERRA[†] ***and*** **Valerio PASCUCCI**[††]**,** ***Nonmembers***

SUMMARY The problem of detecting straight lines arises frequently in image processing and computer vision. Here we consider the problem of extracting lines from range images and more generally from sets of three-dimensional (3D) points. The problem is stated as follows. Given a set Γ of points in 3D space and a non-negative constant ϵ, determine the line that is at a distance at most ϵ from the maximal number of points of Γ. The extraction of multiple lines is achieved iteratively by performing this best line detection and removing at each iteration the points that are close to the line found. We consider two approaches to solve the problem. The first is a simple approach that selects the best line among a randomly chosen subset of lines each defined by a pair of edge points. The second approach, based on tabu search, explores a larger set of candidate lines thus obtaining a better fit of the lines to the points. We present experimental results on different types of three-dimensional data (i) range images of polyhedral objects (ii) secondary structures (helices and strands) of large molecules.

key words: *3D line detection, range images, tabu search*

1. Introduction

The problem of detecting straight lines arises frequently in image processing and computer vision. Many approaches have been proposed and implemented for the case of 2D images. Successful approaches for intensity images do not necessarily represent reasonable solutions for range data. For instance methods based on the Hough transform and its variants are effective for 2D line detection for their insensitivity to noise and gaps in the lines [4]. However, since their memory and computation requirements grow with the number of parameters necessary to define a geometric primitive they do not seem adequate for the detection of lines in 3D space.

Not much work has been done on the detection of lines in range images; most of the research on the extraction of primitives from range data has focused on the segmentation of data into planar patches or into the more complex second-order degree surfaces [11]. Given an image segmented into planar regions, the line segments can be found by tracing and linking the boundary points between adjacent faces in the labeled image. However, the results of such an approach are generally not satisfactory and need to be refined due to the often imperfect results of the segmentation itself. One of the common errors of the range data segmenters is to produce over-segmentation, that is multiple detection of a single surface. Furthermore, often the detected boundaries between adjacent surfaces are distorted and noisy.

One approach to line detection in range images proposed in the literature [3] reduces the problem of line detection in 3D to a 2D problem by first segmenting the range data into planar patches and then finding the boundaries of each planar face. This is accomplished by mapping the edge points assigned to a face into a 2D binary image and applying a 2D Hough transform to find straight line segments that are the boundaries of the planar face. This method finds the same line more than once, since a line is at the boundary of more than one planar face of an object; thus further processing is needed to eliminate the duplicates.

In this paper we consider a direct 3D line detection problem stated as follows. Given a set Γ of n points in 3D space and a non-negative constant ϵ, determine the line that is at a distance at most ϵ from the maximal number of points of Γ. The above problem is solved repeatedly for the extraction of multiple lines after the removal of the points that are found close to the best line. For line detection in range images, the input set Γ is chosen as the set of edge points.

In the following we first discuss a simple approach to solve the 3D line detection problem that determines the best line among all lines defined by pairs of input points. We then present a more robust optimization technique based on *tabu search* that explores a larger set of candidate lines. Both strategies have high computational requirements and may become impractical for applications involving large datasets of points. To reduce the complexity of the algorithms *random sample consensus* (RANSAC) [2], [5] is used so that the search can restricted to a random selected subset of pairs of points.

The rest of the paper is organized as follows. In Sect. 2 we describe a simple approach to line detection in range images. We show that a proper selection of edge points that only takes into account crease edges may lead to better results and also speed up the computation. In Sect. 3 we briefly review the tabu search paradigm for solving optimization problems and show how it can be applied to the problem of detecting lines

Manuscript received March 5, 2001.

Manuscript revised June 26, 2001.

[†]The author is with Università di Padova, Italy, and Purdue University, USA.

[††]The author is with CASC, Lawrence Livermore Nat. Laboratory, USA.

in three-dimensional point sets. Section 4 describes a post-processing phase that is needed to eliminate gaps present in the lines or to remove possible duplicate lines. Experimental results conducted on several data sets are presented in Sect. 5. Finally, we conclude with future work.

2. A Simple Approach

A simple approach to line detection selects the optimal line among those that are defined by pairs of points in the dataset. It does so by determining for each line passing through a pair of edge points the number of other points lying within the expected measurement error ϵ from that line; the best line is the one that corresponds to the maximum computed value. Since for each of the $\frac{n\times(n-1)}{2}$ lines defined by pairs of points we need to examine each of the remaining $n-2$ points, the overall time complexity of the algorithm is cubic in the number of points. For any reasonable value of n the $O(n^3)$ time requirement may be prohibitive. However, in practice a number of pairs much less than the above is acceptable. A relatively small subset of pairs of points randomly chosen can yield results that are within a given bound from the optimal.

The determination of the number k of trials (pairs) that are guaranteed to produce good results with high probability has been considered for the extraction of various parametric shapes and problems [5], [15]. The value k is chosen as follows. Let w be the probability that a single randomly selected data point is within ϵ distance from the line. Let z be the probability that at least one of the random selections is error-free. z is a function of w and k:

$$z = 1 - (1 - w^2)^k$$

Thus, the value k is given by:

$$k = \frac{\log(1 - z)}{\log(1 - w^2)}.$$

In our system, the edge points are found by the scanline approximation algorithm [12], [13]. There are several edge detection algorithms developed in the literature for range images. We have chosen the scanline algorithm because of its simplicity and its high execution speed. Furthermore, this edge detector finds the jump and crease edge points. This information may be useful in reducing the amount of post-processing needed to eliminate duplicate lines. Jump edges are generated by occlusion planes. A line detection algorithm tends to produce separate lines corresponding to jump edges on both sides of the planes; the lines are adjacent on the range grid, but are distant in 3D space and therefore detected as different lines. By using the information about the location of the jump edges, this problem is overcome.

3. Tabu Search

Tabu search (TS) [6], [10] is a powerful optimization technique that has been used to solve a variety of complex combinatorial problems. TS is designed to explore the solution space beyond local optimality. It uses an operation called *move* that changes the current solution and allows to visit a neighborhood of the given current solution. One of the main components of TS is the use of adaptive memory: during the search, local choices are guided by the past history of the search. Restrictions to local moves are imposed by making reference to the memory structures, which store forbidden or tabu moves. This prevents solutions from the recent past from being revisited. Typically, instead of storing the complete solutions that correspond to tabu moves, these structures are recorded in the form of *attributes* that characterize a tabu solution. Tabu classification is not static. The number of iterations to forbid a tabu move, called *tabu tenure*, may change according to the context and vary depending on the status of the search. TS is thus based on a dynamic neighborhood: the set of neighboring solutions may be reduced to a subset of the available moves that may be different at each stage depending on the past history.

At each step of the optimization process, the procedure selects the best move in the local neighborhood and at the same time keeps track of the best solution found so far. Since the best move may result in worse value for the objective function of the current solution, this approach differs significantly from a descent method that only permits moves improving on the solution and ends when no improvement can be found.

Other important components of tabu search are *intensification* and *diversification*. Intensification encourages moves historically found good; diversification encourages moves to solutions that differ significantly from those seen in the past. Thus *diversification* allows to consider moves not in the local neighborhood. Diversification is invoked in the presence of a *critical event*; a critical event occurs for instance when the objective function does not improve for a number of consecutive iterations. In such cases the search can be restarted with a significantly different solution.

The method continues to generate solutions and over time the best known solution continues to improve until a given termination criterion is satisfied.

We have applied the basic TS paradigm to the line detection problem and obtained results of better quality than using the above simple approach. There are a number of important design decisions that have to be made when using the standard tabu search optimization procedure. A crucial aspect of TS, which is common to many other search procedures, is in the choice of an appropriate definition of the neighborhood of a solution. Another key question is the types of tabu at-

tributes to be selected for an effective use of adaptive memory and the various thresholds used for tabu classification and tenure. Finally, it has to be decided how to use randomization within the search procedure to generate candidate solutions and how to set the parameters of the random sampling.

The line detection problem is easily formulated as one objective function. The objective function is defined as number of points within ϵ distance from a line, with ϵ a given constant. The proposed method does not restrict the lines to pass through pairs of input points and therefore may find a better fit of lines to points than the naive approach especially for long line segments.

In our problem, the choice of a neighborhood is relatively straightforward. Neighbors of a given solution are obtained by moving the points that define the current line in a small 3D neighborhood. More precisely, given a solution r_{ij}, that is a line through points P_iP_j, a move operation increments or decrements by a constant amount γ the value of a coordinate of either P_i or P_j or both. TS involves exploring a dynamic neighborhood of a solution and the selection of tabu attributes. Tabu moves correspond to changes in coordinates recently applied and are excluded to avoid revisiting the same solutions. Starting from an initial random solution, the procedure repeatedly selects the best local move as the next solution to explore. Furthermore, it keeps track of the best known solution over time and updates the list of tabu moves. By allowing moves to non improving solutions, the search can continue beyond local optima.

The initial solution is chosen as a line within ϵ distance from at least t of points. Such a line is found by repeatedly selecting pairs of random points until one satisfying the above criterion is obtained. To reduce the effect of noise and improve the performance, only pairs of points with Euclidean distance within a given range (depending on the application) are examined for generating candidate lines. The parameter t is updated during the processing, it has a larger value at the beginning so that higher weight lines are considered initially and decreases towards the end of the processing.

At each step we examine the entire neighborhood of available moves and always select the best move that is not tabu (starting from the initial solution found by the above heuristics). Tabu classification occurs after the best move is found. A move "opposite" to the current best move is labeled tabu. An opposite move changes the coordinate of the point(s) affected by the move by the opposite amount. A move is kept tabu for a number iterations, *tabu-tenure*, that is the same for all tabu moves and, in our implementation, does not change during the process. Since the neighborhood is relatively small, the information about tabu tenure can be easily stored for all moves.

If for a few consecutive attempts at changing the current solution no improvement was found, we depart significantly from the current solution and randomly generate a new starting solution. The step terminates when this diversification procedure fails to produce a better solution for a number of consecutive attempts.

A skecth of the algorithm to detect one single line follows.

Set i to 0. m and z are positive constants.

1. set i to $i + 1$.

   Select an initial solution as follows. Repeatedly select a line through a pair of random edge points P_i, P_j until one is found that is within ϵ distance from at least t edge points. (t is decremented during the processing).

2. repeat

   {Select the best non tabu local move of the current solution.
   Update the current solution to the one obtained above.
   Label the corresponding opposite move tabu.
   Check tenure for all other tabu moves.
   Keep track of the best solution found so far.}

   until (the objective function did not increase for more than z iterations)

3. if $(i < m)$ go to 1.
4. end

After a new line is detected, the longest segment on the line with no large gaps on it is entered into the list of detected segments. The points at a distance less than δ from the segment are then removed from the list of points; the list consisting of the remaining edge points is used in the next iteration to detect the next best line. The constant δ is generally chosen larger than ϵ to account for errors in the measurements as well as in the edge detection process. For multiple line detection, the above procedure is repeated. The overall procedure terminates when very few points are left in the list of remaining points.

The TS for line detection has been successfully applied to reasonably small sets (hundreds of points) from biological datasets, in particular for the determination of linear segments associated to secondary structures in proteins. For applications in vision involving range data the number of boundary points is generally high (few thousands points) and the extra work required to optimize the model fitting is justified when the level of noise and distortion is high and high accuracy is required.

4. Merging Segments

The procedures described above to find the segments that best represent the data may produce an oversegmentation, that is more segments than those actually present in the image. To overcome this problem,

a post-processing phase is needed to eliminate spurious segments or to merge contiguous segments.

Next we describe how to merge two 'close' segments. The following two criteria define close segments:

- the lines containing the two segments have approximately the same slope.
- the distance between the two segments is below a given threshold.

The distance between two segments is defined as the minimum distance between the two lines containing the segments, if this minimum distance is achieved by points internal to the two segments; otherwise, it is defined as the minimum of the distances between the extreme points of the two segments.

Once two segments P_iP_j and P_kP_l are found close according to the two previous criteria, they are replaced by a single segment r' defined in terms of its midpoint P_m and orientation vector e_m as follows. Let

$$P_m = \frac{w_{ij}(P_i + P_j)/2 + w_{kl}(P_k + P_l)/2}{w_{ij} + w_{kl}}$$

where $w_{ij}(w_{kl})$ is the number of input points within ϵ distance from $P_i, P_j\,(P_k, P_l)$. In other words, the two segments P_iP_j and P_kP_l are given a weight that is the number of edge points associated to the segments. This weighting factor penalizes short or sparse segments. Let $e_{ij}(e_{kl})$ be the unit vector of $r_{ij}(r_{kl})$. The unit vector e_m of r' is given by:

$$e_m = \frac{w_{ij}e_{ij} + w_{kl}e_{kl}}{w_{ij} + w_{kl}}$$

To determine the endpoints of r' we project P_i, P_j, P_k and P_l on the line of r' and choose the two projections that are the farthest apart.

5. Experimental Results

The program LineDetection that implements the proposed approaches to line detection in 3D is written in C. It runs interactively and allows to choose either the simple approach or the tabu search, and the use of the post-processing phase for possible merging of some of the detected segments.

We have tested the program on 40 range images of polyhedral objects acquired by an ABW structures light scanner[†]. These images have been used for an experimental evaluation of several methods for segmenting range images into planar or high-order surfaces [11]. The sets of edge points, inputs to our algorithms, have been extracted from the images using the scanline approach developed in [13].

To segment the ABW test range images we have used the following values for the parameters. ϵ was set to 1.0. Recall that a point within ϵ distance from a line is considered close to the line and increases the objective function. After a line is detected the points within distance $\delta \geq \epsilon$ from it are removed from the list of available points; δ is chosen to 2.5 in ours tests. A line is defined by two points randomly selected from the set of remaining edge points. The distance between the two points should not be below 15 and not exceed 45 for the line to be considered a candidate line. Another parameter used for the selection of candidate lines specifies the minimum number of points t that have to be close to the line. The initial value of t has been set to 15.

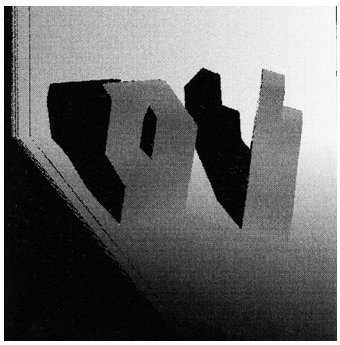


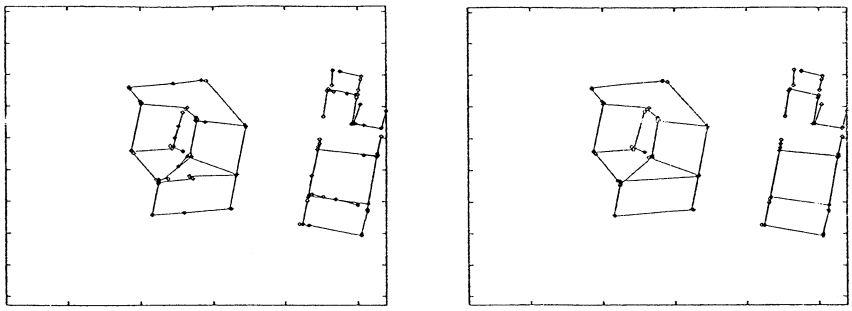
Fig. 1 The range image abw.train.9.

The values of other parameters required by TS and used with the ABW range images are listed in the following. Tha parameter γ used in a move operation to change the coordinate value of a point through the line is set to 1.0. The tabu tenure is equal to 8. A critical event occurs after the objective function does not improve for 20 iterations. The search for multiple lines terminates when fewer than 250 points are left in the list of available edge points.

Figure 2 shows the outputs of the program on the range image abw.train.9 of Fig. 1, using the simple approach and the tabu search. In both cases, the merging procedure for the resulting segments has been applied. From the figure, the improvements of the TS in terms of fewer fragmented segments are clear. The execution times on a SUN Sparcstation20 are 0.3 s and 0.6 s for the simple and TS approaches, respectively.

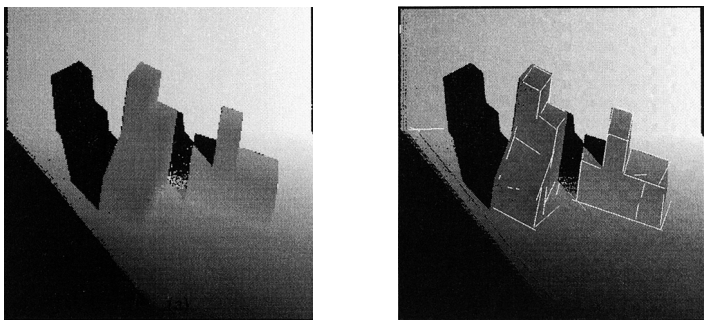
Figure 3 shows the range image abw.train.3 and the obtained segments superimposed to the input image. It appears that the results are not very satisfactory in this case: there are missing segments and segments that do not correspond to actual boundary segments in the scene. However, this seems to be due to a poor edge detection, as can be seen from Fig. 4, which in turn reflects the poor quality of the input range image. The line segments extracted by the two methods have been used for finding correspondences between sets of segments. The matching approach presented in [8], [9] computes the minimum Hausdorff distance between sets of segments and derives the best rigid transformation that maps one set into the other. The segments generated by tabu search have generally provided a bet-

[†]The images are available from http://marathon.csee.usf.edu/range/seg-comp/SegComp.html



(a) (b)

Fig. 2 The outputs of abw.train.9 with the simple algorithm (a) and TS (b).



(a) (b)

Fig. 3 The range image abw.train.3, (a) and the output of TS superimposed to the input image (b).

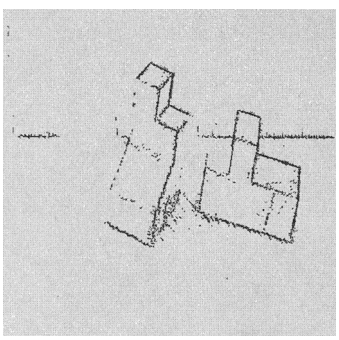


Fig. 4 The set of edge points extracted from abw.train.3.

ter input for the matching procedure that was able to find a better overlap for the sets of segments.

We have also conducted experiments on sets of biological data, more precisely on the 3D structures of proteins. Our approach has been used to determine linear segments that best approximate the secondary structures, i.e. helices and strands, of proteins. The input to the program is the set of atoms of a protein with their 3D coordinates and the specification of secondary structures. The segments obtained from several proteins have been used for proteins structure comparison and fast retrieval of substructures from protein databases based on line segment representation of proteins [7]. As an example, Figs. 5(a) and 6(a) display protein 1IFB and 1ALB, respectively, from the protein data bank PDB [1]. Figures 5(b) and 6(b) show the line segments associated to the secondary structures of proteins 1IFB and 1ALB. Each segment is displayed as a cylinder with the radius equal to 1.0. In this application ϵ has been chosen equal to 1.7 Angstrom which is a typical value for the width of a secondary structures.

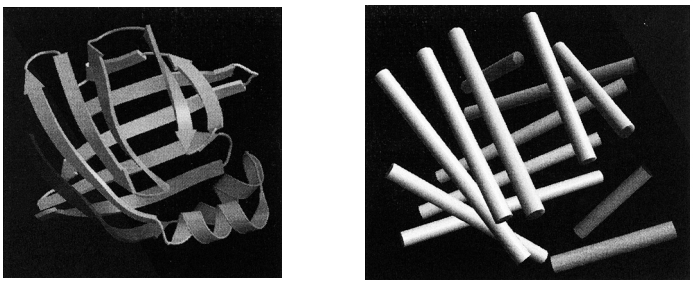
6. Conclusions

We have presented two approaches to the line detection problem in range images and shown results on range images of polyhedral objects. Although the TS requires higher execution times, the results obtained with this strategy are generally better. We have used the line segments extracted from range images as the inputs to a matching algorithm that finds correspondences between line segments in two 3D objects. When TS was chosen to generate the line segments, generally a better match was found.

As a final note, the segments extracted by this method do not generally correspond to a valid boundary representation of an object [14]. One necessary condition satisfied by a valid boundary representation is that the boundary segments either are disjoint or intersect at a common vertex. This is often violated by the output of the algorithm. To enforce these conditions, further processing is necessary. Obtaining a valid boundary is beyond the scope of this work.

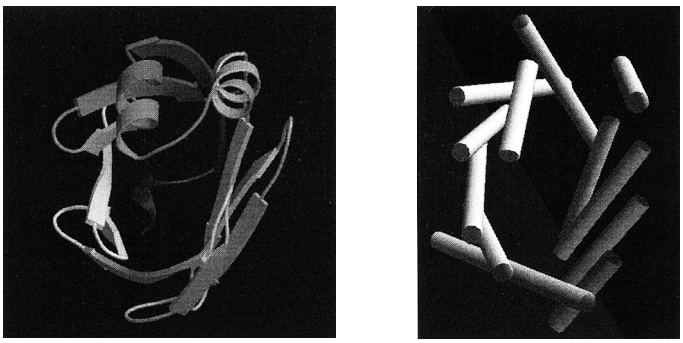
Acknowledgments

Support for Guerra was provided in part by the Italian



(a) (b)

Fig. 5 (a) The protein 1IFB from the PDB database and its secondary structures. (b) The line segments associated to the secondary structures obtained with TS. Each segment is displayed as a cylinder with the radius equal to 1.0A.



(a) (b)

Fig. 6 (a) The protein 1ALB from the PDB database and its secondary structures. (b) The line segments associated to the secondary structures obtained with TS. Each segment is displayed as a cylinder with the radius equal to 1.0A.

Ministry of University and Research under the National Project "Bioinformatics and Genomics Research," and by the Research Program of the University of Padova. The work of Pascucci was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No.W-7405-ENG-48. Gabriele Peterle helped with the programs.

References

[1] E.E. Abola, J.L. Sussman, J. Prilusky, and N.O. Manning, "Protein Data Bank archives of three-dimensional macromolecular structures," Methods Enzymol, vol.277, pp.556–571, 1997.

[2] C. Bolles and M.K. Fischler, "A RANSAC-based approach to model fitting and its applications to finding cylinders in range data," Proc. 7th Int. Joint Conf. Art. Intel., 1981.

[3] B.A. Boyter and J.K. Aggarwal, "Recognition with range and intensity data," Proc. Workshop on Computer Vision: Representation and Control, MD, pp.112–117, IEEE, 1984.

[4] R.O. Duda and P.E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Commun. ACM, vol.15, no.1, pp.11–15, Jan. 1972.

[5] M.A. Fischler and R.C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," Commun. ACM, vol.24, no.6, pp.381–395, June 1981.

[6] F. Glover and M. Laguna, Tabu search, Kluwer Academic Publishers, Boston, 1997.

[7] C. Guerra and V. Pascucci, "Segment matching for protein secondary structure comparison," (abstract), Proc. Third International Conf. Computational Molecular Biology, RECOMB99, p.30, Lyon, France, 1999.

[8] C. Guerra and V. Pascucci, "3D segment matching using the Hausdorff distance," Proc. IEEE Conf. Image Processing and its Applications, IPA99, pp.18–22, Manchester, UK, 1999.

[9] C. Guerra and V. Pascucci, "On matching sets of 3D segments," Proc. SPIE Vision Geometry VIII, pp.157–167, Denver, USA, 1999.

[10] A. Hertz, E. Taillard, and D. de Werra, "Tabu search," in Local Search in Combinatorial Optimization, pp.121–136, Wiley & Sons, 1997.

[11] A. Hoover, G. Jean-Baptiste, X. Jiang, P.J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, and R. Fisher, "An experimental comparison of range image segmentation algorithms," IEEE Trans. Pattern Anal. Mach. Intell., vol.18, no.7, pp.1–17, July 1996.

[12] X.Y. Jiang and H. Bunke, "Fast segmentation of range images into planar regions by scanline grouping," Machine Vision Applications, vol.7, no.2, pp.115–122, 1994.

[13] X.Y. Jiang and H. Bunke, "Robust and fast edge detection and description in range images," MVA'96 IAPR Workshop on Machine Vision Applications, pp.538–541, Japan, 1996.

[14] A.A.G. Requicha, "Representation for rigid solids: Theory, methods, and systems," Computing Surveys, vol.12, no.4, pp.437–464, 1980.

[15] G. Roth and M.D. Levine, "Extracting geometric primitives," Computer Vision, Graphics, and Image Processing. Image Understanding, vol.58, no.1, pp.1–22, July 1993.