Identifying Well-Covered Minimal Bounding Rectangles in 2D Point Data

Marc van Kreveld* Thijs van Lankveld* Remco Veltkamp*

Abstract

Laser range imaging is an upcoming data source for many fields of research. Although many methods are proposed for finding surfaces and classifying points, finding the bounds of these surfaces is still a difficult problem. We propose a method for finding, in a point set of size n, a rectangle that contains a predefined subset of the points and none of the other points in $O(n \log n)$ time. Further, there may not be places in the rectangle that are too far from all points in the predefined subset.

1 Introduction

A type of data that has lately seen an increase in use is the laser range scan, also known as light detection and ranging (LiDAR) data. This data consists of three-dimensional points that are scanned from surfaces. The accuracy and resolution of the measurements is high. Currently, many scanners also include a photosensitive sensor that assigns a color value to each point. A more complete description of the properties of LiDAR data is given in [9, 10].

When this data is studied interactively it is easy to form a mental image of the geometry of the scene. The collection of 3D points gives insight into the surfaces that were sampled, and many methods have been proposed to automatically detect these surfaces [12, 14]. One influential group of methods is the methods based on RANSAC [3], which uses a minimal random sample to define a surface and iterates to find the surface containing most points. Another group of methods is based on region growing [14], which initializes a bounded surface at a certain point and tries to expand this while maintaining some constraint like maximal deviation from the surface in location or normal. Finally, Funke, Malamatos, and Ray [4] give an approximation algorithm for finding a connected component with comparable normals in a triangulation. In the last two methods, constraints on local planarity might not hold globally.

Though surfaces can be found in the data, these are usually either unbounded or of an arbitrary and overly complex shape. Computing the correct bounded shapes from these surfaces is far from trivial. Problems arise because the measurements are not regularly distributed over the surfaces or not well suited for detecting edges. In many cases it is possible to determine which points are on a surface, but it is still difficult to find the original boundary of this surface. A similar problem arises when we are not interested in the whole surface, but in smaller sections that should be separated from the surface. An example of this is a known surface of a facade in which we want to find the regions that make up windows or shutters based on their color. Comparable work has been done using LiDAR data when the topology of the original object is known [11].

Within the remainder of this paper, we assume the surface has already been extracted using the method of Schnabel, Wahl, and Klein [12]. This shifts the problem from 3D to 2D. Schnabel's method also filters out the points that are not on the surface. An example of LiDAR data after surface detection is shown in Figure 1. We will assume it is known which points are

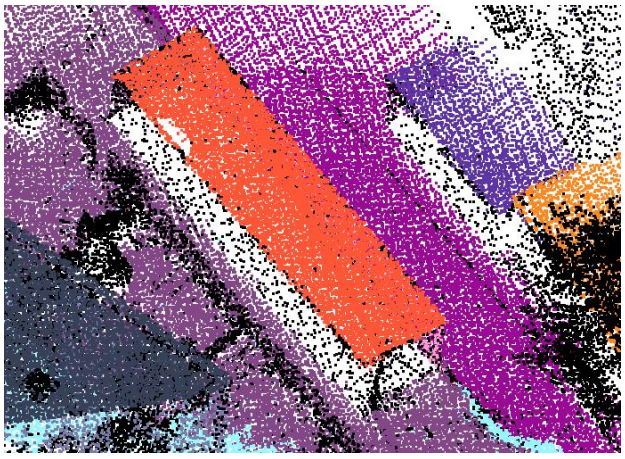

Figure 1: The LiDAR data, in which points close to the same plane have been given the same color. In the rest of the paper, we will call these points blue.

inside the shape we want to find, for example a window or shutter, and we will call these the blue points. All others will be called red points. This distinction may be made by a so-called superpixel method [2].

Many objects have rectangular shapes and our method is constrained to finding these. Usually, if there is one rectangle that is valid because it contains all blue points and no red points, then there are many of these rectangles with different orientations

*Department of Information and Computing Sciences, Utrecht University, marc@cs.uu.nl, thijsvl@cs.uu.nl, Remco.Veltkamp@cs.uu.nl

and sizes. We constrain the problem to finding the angles $\theta \in [0, \frac{\pi}{2})$ for which there is some *blue* rectangle $\mathcal{R}_\theta$ whose edges are rotated by θ from axis-aligned.

Definition 1 *A rectangle $\mathcal{R}$ is blue with respect to point sets B and R if and only if it contains all points in B and no points in R.*

For our method we use the minimal oriented bounding box at angle θ to check if there is a blue $\mathcal{R}_\theta$. This is valid because to decide if a blue $\mathcal{R}_\theta$ exists, we can test the smallest one.

Generally, we are mostly interested in rectangles that give a good indication of the distribution of a point set. To incorporate this, the problem is extended to finding rectangles that are well covered by the blue points. For a rectangle $\mathcal{R}$ to be well covered, we require that any point inside $\mathcal{R}$ has a blue point within a pre-specified distance δ.

Definition 2 *A rectangle $\mathcal{R}$ is δ-covered by point set B if and only if the union of all disks with radius δ and center $c \in B$ covers $\mathcal{R}$.*

Once the angles θ are identified for which there is a blue rectangle $\mathcal{R}_\theta$ that is δ-covered by B, these rectangles can be searched for the smallest area or diameter rectangle, or for rectangles with a parallel edge within (other) surfaces. However, these applications fall outside the scope of this paper.

Problem The problem solved algorithmically in this paper is as follows. Let R and B be point sets of combined size n, and let δ be a scalar. Determine all angles $\theta \in [0, \frac{\pi}{2})$ for which there is a blue rectangle $\mathcal{R}_\theta$ that is δ-covered by B.

To solve this problem, we assume the data has two properties. Firstly, we assume the data does not contain noise, unlike the original sensor data. Secondly, as stated earlier, we assume it is known which points fall inside and outside the rectangle, that is, which points are blue and which are red. This distinction may be determined beforehand based on the color of each point, or some other property.

Our problem is a red-blue separability problem; see [1, 5, 7, 8] for other papers. Our problem's most distinguishing feature is the extra requirement that the rectangle that is the separator must be δ-covered by the points that lie inside.

The algorithm we present for solving the problem uses ideas of sweep and scan algorithms. The method most closely related is rotating calipers [13], most notably when rotating calipers is used to find the minimal bounding box.

In the next section an algorithm for solving the problem without coverage requirements is given. Section 3 extends this algorithm to solve the complete problem. Finally, we discuss the results and future research.

2 A simple separation algorithm

This section introduces the basic framework for solving the problem of finding a blue rectangle. The approach is to rotate a rectangle $\mathcal{R}$ around point set B and handle important events as they occur. There are two causes for events.

Firstly, because $\mathcal{R}$ is minimal, all of its edges contain at least one point of B, and an event occurs when these points change. These points and the order in which they change can be determined from the boundary of the convex hull $\mathcal{CH}_B$ of B. The events are the same as those used for determining the oriented minimal bounding box with rotating calipers.

Secondly, an event occurs when a red point r enters or exits $\mathcal{R}$. The angle of these events can be determined from the two lines through r, tangent to $\mathcal{CH}_B$. Note that a red point r will never be inside $\mathcal{R}$ if it is too far from $\mathcal{CH}_B$. Let v_1, v_2 be vertices of $\mathcal{CH}_B$ on the tangents through r. Then r can only enter $\mathcal{R}$ if $\angle v_1 r v_2 \geq \frac{\pi}{2}$.

The events are inserted into a queue sorted on angle of rotation at which they occur. After determining which red points are inside $\mathcal{R}$ before rotation, the events are handled in order. The algorithm keeps track of the number of red points inside $\mathcal{R}$ and whenever $\mathcal{R}$ becomes or stops being blue, this information is stored, together with the current angle. When all events have been handled, all angles θ for which $\mathcal{R}_\theta$ is blue are known.

Theorem 1 *For point sets B and R with total size n, all oriented bounding boxes of B that do not contain any point in R can computed in $O(n \log n)$ time.*

Proof. Constructing $\mathcal{CH}_B$ takes $O(n \log n)$ time. This also gives the angles when the points on the edges of $\mathcal{R}$ change. Using a binary search on the vertices of $\mathcal{CH}_B$, all lines through a red point and tangent to $\mathcal{CH}_B$ can be computed in $O(n \log n)$. Each edge of $\mathcal{R}$ will start containing a vertex of $\mathcal{CH}_B$ once and there are two lines tangent to $\mathcal{CH}_B$ through each red point. This means that there are $O(n)$ events. Sorting the events by angle takes $O(n \log n)$ time. During rotation, after each event it is checked if the rectangle has become empty or non-empty. This takes $O(1)$ time per event. □

3 Requiring δ-coverage

The algorithm of the previous section provides an effective way to determine the angles for which a rectangle is blue. We would also like the rectangles not to contain large regions void of blue points. In other

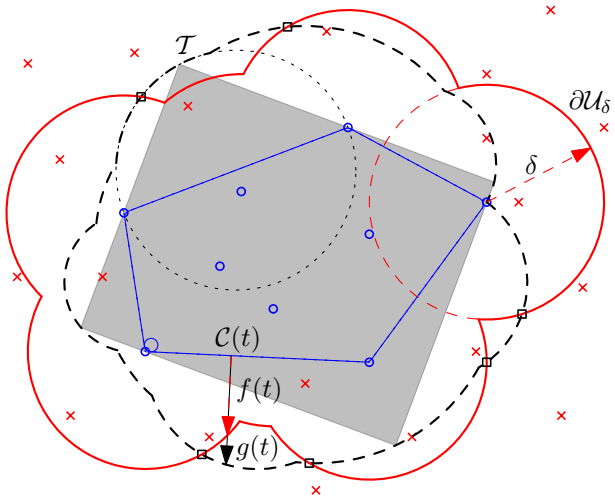

Figure 2: The stuctures used to find the rectangles that contain all blue points B (circles) and no red points R (crosses) when δ-coverage is required. The boundary $\partial\mathcal{U}_\delta$ of the permissible region is shown solid, while the trajectory of the corners $\mathcal{T}$ is shown dashed. The rounded convex hull $\mathcal{C}$ and the functions $f(t)$ and $g(t)$ are also shown.

words, $\mathcal{R}$ must be δ-covered by B for some δ, which we assume is given.

To solve this problem, two additional structures are used: for the permissible region, and for the trajectory of the corners of $\mathcal{R}$. These structures are shown in Figure 2.

The permissible region $\mathcal{U}_\delta$ is the maximal region that is δ-covered by B. It is the union of all δ-disks centered on a point in B. To satisfy the δ-coverage criterion, $\mathcal{R}$ must be completely inside $\mathcal{U}_\delta$. It is well-known that $\mathcal{U}_\delta$ is bounded by $O(n)$ arcs and can be computed in $O(n \log n)$ time. If part of $\mathcal{CH}_B$ is not in $\mathcal{U}_\delta$, then no rectangular separator exists that is δ-covered. In the rest of this paper we will assume $\mathcal{CH}_B \subseteq \mathcal{U}_\delta$ and in this case $\mathcal{U}_\delta$ will have only one connected component whose boundary we denote by $\partial\mathcal{U}_\delta$, see Figure 2.

The trajectory $\mathcal{T}$ is the cycle of circular arcs, shown in Figure 2, that a corner of $\mathcal{R}$ follows as $\mathcal{R}$ is rotated over a full 2π turn. This concept has been previously used by Hoffmann *et al.* [6].

Lemma 2 *$\mathcal{T}$ can be constructed from $\mathcal{CH}_B$ in $O(n)$ time and contains $O(n)$ vertices and circular arcs.*

Proof. Let c be a corner of $\mathcal{R}$ and let $b_1, b_2 \in B$ be the contact points on the two edges of $\mathcal{R}$ incident to c. It follows from Thales' theorem that, while the contact points are unchanged, c follows a circular arc on the circle whose diameter is $b_1 b_2$. A new arc starts only when a contact point changes, and this happens $O(n)$ times. The sequence of the $O(n)$ arcs forms $\mathcal{T}$. □

Because the center of each arc in $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ is on $\mathcal{CH}_B$ and no arc intersects this cycle, the following observation can be made.

Observation 1 *Any outward ray emanating perpendicular to an edge of $\mathcal{CH}_B$ intersects $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ exactly once.*

The algorithm that solves the problem is based on the algorithm given in Section 2. The only difference is the addition of events when $\mathcal{R}$ starts and stops being δ-covered by B. There are two ways in which this can happen. Either during rotation an edge of $\mathcal{R}$ passes over a vertex of $\partial\mathcal{U}_\delta$, or a corner of $\mathcal{R}$ passes over an arc of $\partial\mathcal{U}_\delta$.

The vertices of $\partial\mathcal{U}_\delta$ are handled exactly like extra red points. This will add $O(n)$ events.

The events caused by a corner of $\mathcal{R}$ passing over $\partial\mathcal{U}_\delta$ happen exactly at the intersections of $\partial\mathcal{U}_\delta$ and $\mathcal{T}$, shown as square markers in Figure 2.

Lemma 3 *If $\mathcal{U}_\delta$ covers $\mathcal{CH}_B$, there are $O(n)$ intersections between $\partial\mathcal{U}_\delta$ and $\mathcal{T}$.*

Proof. The proof builds on the property that two piecewise simple functions with n sections have $O(n)$ intersections that can be computed in $O(n)$ time. By simple we mean that two such curves can intersect each other only a constant number of times.

To use this property, we must define two functions that have the same value at some argument if and only if $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ intersect. To create these functions, an extra structure is introduced to parameterize $\partial\mathcal{U}_\delta$ and $\mathcal{T}$, and thereby give the argument to define the functions. We will show that the functions created using this structure "detect" all intersections of $\partial\mathcal{U}_\delta$ and $\mathcal{T}$, which completes the proof.

Let $\mathcal{C}$, shown in Figure 2, be the boundary of the morphological opening, the dilation of the erosion, of the region inside $\mathcal{CH}_B$ using an ϵ-disk. The radius $\epsilon > 0$ is chosen such that for all vertices of $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ that are not on a vertex of $\mathcal{CH}_B$, the closest point on $\mathcal{CH}_B$ is also on $\mathcal{C}$.

Let $\mathcal{C}(t)$ be the point reached by following $\mathcal{C}$ from its topmost point for the fraction t of its length, and let $r(\mathcal{C}(t))$ be the outward ray emanating perpendicularly from $\mathcal{C}(t)$. Now $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ define functions by taking $f(t) = dist(\mathcal{C}(t), r(\mathcal{C}(t)) \cap \partial\mathcal{U}_\delta)$ and $g(t) = dist(\mathcal{C}(t), r(\mathcal{C}(t)) \cap \mathcal{T})$. These functions $f(t)$ and $g(t)$, shown in Figure 2, are well-defined: for each t, $r(\mathcal{C}(t))$ intersects $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ exactly once. This is true for each straight part of $\mathcal{C}$ because of Observation 1. Furthermore, for every circular arc of $\mathcal{C}$, all rays starting in the center of the corresponding ϵ-disk and intersecting the arc, intersect the same arcs of $\partial\mathcal{U}_\delta$ and $\mathcal{T}$; the only exception to this is when a vertex of $\mathcal{T}$ is on a vertex of $\mathcal{CH}_B$. But in this case, such a ray will intersect $\mathcal{T}$ once as well.

The properties of $f(t)$ and $g(t)$ ensure that each point on $\partial\mathcal{U}_\delta$ and $\mathcal{T}$ occurs for some t as the intersection of $r(\mathcal{C}(t))$ and $\partial\mathcal{U}_\delta$, respectively $\mathcal{T}$, and this is the only intersection point for that ray. Hence, $f(t)$ and $g(t)$ have the property that a value of t for which $f(t) = g(t)$ corresponds one-to-one to an intersection point of $\partial\mathcal{U}_\delta$ and $\mathcal{T}$. Since $f(t)$ and $g(t)$ are also piecewise simple, the result follows. □

Lemma 3 implies that there are a linear number of events of the type where a corner of $\mathcal{R}$ is at distance exactly δ from the nearest blue point. These are additional to the ones of the algorithm presented in Section 2 and the vertices of $\partial\mathcal{U}_\delta$. Handling these events is trivial. This leads to the following result.

Theorem 4 *For point sets R and B with total size n, and scalar δ, all angles $\theta \in [0, \frac{\pi}{2})$ for which there is a blue rectangle $\mathcal{R}_\theta$ that is δ-covered by B can be computed in $O(n \log n)$ time.*

4 Discussion

We presented an algorithm that computes all rectangles that tightly cover the 'blue' points in one set but not the 'red' points in another set. An extension of this algorithm also guarantees that no part of the rectangle does not have a blue point nearby. The algorithm is efficient and relatively simple to implement. To make it more useful for the application of geometric model reconstruction from LiDAR data, a number of extensions may be considered.

Firstly, we may need to allow a small number of red points inside the rectangle, because errors may have been made in the classification into red and blue points. It is easy to extend our algorithm to allow up to some pre-specified number of red points inside.

Secondly, we may want to allow a small number of blue points to be outside the rectangle. This extension appears considerably harder, and we do not expect to achieve the same running time in this case.

Thirdly, we may want to allow a small part of the area inside the rectangle to be not δ-covered by the blue points. Due to minor occlusion, it may be that some blue points are missing in a region. This extension is also difficult to incorporate, although an approximation version seems possible.

Fourthly, it would be useful to extend the algorithm to finding different shapes than rectangles, like L-shapes, without complicating the algorithm too much.

A final question is whether our algorithm or some adaptation of it will perform well on data that it would be given in real applications. Only by experimentation can we discover whether the quality of the data (density of sampling, limited presence of noise, proper classification of inliers and outliers) is sufficient to make our approach work well.

Acknowledgments This research has been supported by the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie).

References

[1] P. K. Agarwal, B. Aronov, and V. Koltun. Efficient algorithms for bichromatic separability. *ACM Trans. on Alg.*, 2(2):209–227, 2006.

[2] P. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. of Comp. Vis.*, 59(2), 2004.

[3] M. A. Fischler and R. C. Bolles. RANdom SAmple Consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6):381–395, 1981.

[4] S. Funke, T. Malamatos, and R. Ray. Finding planar regions in a terrain: in practice and with a guarantree. In *SCG '04: Proc. of the 20th Annual Symp. on Comp. Geom.*, pages 96–105, New York, NY, USA, 2004. ACM.

[5] S. Har-Peled and V. Koltun. Separability with outliers. In *ISAAC*, pages 28–39, 2005.

[6] F. Hoffmann, C. Icking, R. Klein, and K. Klaus. The polygon exploration problem II: The angle hull. Technical Report 245, Fernuniversität Hagen, Praktische Informatik VI,, 1998.

[7] F. Hurtado, M. Mora, P. A. Ramos, and C. Seara. Separability by two lines and by nearly straight polygonal chains. *Discrete Applied Mathematics*, 144(1–2):110–122, 2004.

[8] F. Hurtado, C. Seara, and S. Sethia. Red-blue separability problems in 3d. *Int. J. Comp. Geom. Appl.*, 15(2):167–192, 2005.

[9] John Chance Land Surveys and Fugro. Fli-map specifications. http://www.flimap.com/site47.php.

[10] H.-G. Maas. Fast determination of parametric house models from dense airborne laserscanner data. In *ISPRS Workshop on Mobile Technology*, Bangkok, Thailand, April 1999.

[11] S. Pu and G. Vosselman. Automatic extraction of building features from terrestrial laser scanning. *Int. Arch. of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5), September 2006.

[12] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, June 2007.

[13] G. Toussaint. Solving geometric problems with the rotating calipers. In *Proc. of the IEEE MELECON '83*, pages A10.02/1–4, 1983.

[14] Y.-H. Tseng, K.-P. Tang, and F.-C. Chou. *Surface reconstruction from LiDAR data with extended snake theory*, volume 4679 of *Lecture Notes in Computer Science*, pages 479–492. Springer Berlin / Heidelberg, August 2007.