Computing the Set of All Distant Horizons of a Terrain

Daniel Archambault* William Evans* David Kirkpatrick*

Abstract

We study the problem of computing the set of all distant horizons of a terrain, represented as either: the set of all edges that appear in the set of all distant horizons; the connected sets in the union of all points that appear in the set of all distant horizons (the set of *edge fragments*); or a search structure to efficiently calculate the edge fragments or edges on a distant horizon from a particular viewing direction. We describe a randomized algorithm that can be used to solve all three forms of the problem with an expected run time of $O(n^{2+\epsilon})$ for any $\epsilon > 0$ where n is the number of edges in the piecewise linear terrain. We show that solving either of the first two versions of the problem is 3SUM hard, and we also construct a terrain with a single local maxima and a quadratic number of edge fragments in the set of all distant horizons, showing that our solution to the second version of the problem is essentially optimal.

1 Introduction

A *terrain* is a piecewise linear, two-dimensional function in three dimensions. It is represented using a polyhedral mesh composed of faces, edges, and vertices. The function is defined over the xy-plane and the function value or z value of the terrain is the point's *height* or *altitude*. Intuitively, a *distant horizon* of the terrain is a horizon from some horizontal viewing direction. Formally, a point on an edge of the terrain appears on the distant horizon if and only if it supports a horizontal line V (parallel to the xy-plane) that does not properly intersect the terrain. Such a line V is called a horizontal *visual line* in the visual hull literature [11] [12].

Our goal is to compute the set of all distant horizons of a terrain. One motivation for this goal is to obtain a representation of the terrain data that is less detailed and thus faster to render, but that closely approximates the view of the true terrain when rendered from any distant view point. We could query for this distant horizon given a horizontal viewing direction or we could pre-calculate the set of edges or edge fragments that contribute to the set of all distant horizons. By rendering this pre-calculated set of edges, we render an accurate distant horizon independent of the horizontal viewing direction. We therefore consider three versions of the distant horizon problem:

1. The problem of computing all *distant horizon edges* (the set of terrain edges that contribute to at least one distant horizon).

2. The problem of computing the set of all *distant horizon edge fragments* (the set of terrain edge fragments that contribute to at least one distant horizon). These edge fragments are the connected sets in the union of all points on the terrain that appear in at least one distant horizon.

3. The problem of computing a query data structure that returns the set of edge fragments or the set of edges from a given horizontal viewing direction.

The paper places our results in the context of previous work, describes an algorithm to compute the set of all distant horizons of a terrain, and then discusses the complexity of each of the problems described above.

2 Previous and Related Work

The problem of computing the horizon of a terrain of n edges from a single viewing direction can be reduced to computing the upper envelope of n line segments in the plane. A single horizon has near linear complexity $\Theta(n\alpha(n))$ [4] where $\alpha(n)$ is the inverse Ackermann function, a very slow growing function that can be considered constant for most practical values of n. Many algorithms exist to compute the horizon from a single view point. Atallah [2] describes a divide-and-conquer scheme to compute the upper envelope of n line segments in $O(n\alpha(n)\lg n)$ time by recursively dividing the line segments into halves and pairwise merging the resultant upper envelopes via a sweep line technique. Hershberger [10] improves the running time of the algorithm to $O(n \lg n)$ by carefully selecting the halves in Atallah's scheme to ensure that the upper envelopes have linear complexity at each level. De Floriani and Magillo [7] describe a randomized algorithm that computes the horizon of a terrain in $O(n\alpha(n)\lg n)$ expected time. The algorithm uses a data structure that can insert or remove line segments from the horizon as the viewpoint changes or if the terrain changes its level of detail. Stewart [14] develops an algorithm that can compute the approximate horizon of a terrain at all n vertices of the terrain in parallel. He divides the view around each vertex into s sectors and assumes that the elevation within a sector is constant, leading to an algorithm with a runtime of $O(n\lg^2 n + sn)$.

*Department of Computer Science, The University of British Columbia, 201-2366 Main Mall, Vancouver, B.C. V6T 1Z4, [archam, will, kirk]@cs.ubc.ca

There has been some work done on bounding the number of combinatorially distinct orthographic views of a terrain. Two such views are distinct if the two sets of edges, faces, vertices, and intersections visible in the two projections from the two viewing directions are distinct. De Berg et al. show a lower bound of $\Omega(n^5\alpha(n))$ for any $\epsilon > 0$ [5], while Agarwal and Sharir show an upper bound of $O(n^{5+\epsilon})$ for any $\epsilon > 0$ ([1] and [9]). A critical part of the proof in [9] is dualizing the n edges of the terrain into n sets of $n-1$ bivariate functions of bounded, constant degree and taking the upper envelope of each. Our algorithm uses a similar dualization transform to compute the set of all distant horizons and the resulting functions can be shown to be of bounded, constant degree. We describe this algorithm in the next section.

3 Computing the Set of All Distant Horizons

An edge appears on the distant horizon if and only if it supports a horizontal visual line. If we intersect a terrain with an arbitrary vertical plane π then only the edge whose intersection point is of highest altitude supports a horizontal line in π that satisfies this definition (figure 1 right). All other horizontal tangents in π to the edges that intersect π must properly intersect the terrain. Thus, we could compute the set of distant horizons by transforming each terrain edge e into a surface patch $(m, b, f_e(m, b))$ in a dual (m, b, h)-space, where the partial function $f_e(m, b)$ is the height of the edge e in the vertical plane π whose vertical projection has slope m and y-intercept b. Let the edge e have vertices (x_1, y_1, h_1) and (x_2, y_2, h_2). For a fixed slope m, $b_1(m) = y_1 - x_1 m$ gives the y-intercept of the unique horizontal line with slope m passing through the vertex (x_1, y_1, h_1). Similarly, $b_2(m) = y_2 - x_2 m$ for (x_2, y_2, h_2). The function $f_e(m, b)$ for this edge is given by[1]:

$$f_e(m,b) = \begin{cases} (h_2 - h_1)\dfrac{b - b_1(m)}{b_2(m) - b_1(m)} + h_1, & \\ \qquad b \in [b_1(m), b_2(m)], & \\ \qquad b_1(m) \neq b_2(m) & \\ \max(h_1, h_2) & b = b_1(m) = b_2(m) \\ \text{undefined} & \text{otherwise} \end{cases}$$

We can use an upper envelope calculation [9], [13], or [3] to compute the upper envelope of the surface patches associated with the terrain edges in dual (m, b, h)-space. This upper envelope can be computed in $O(n^{2+\epsilon})$ for any $\epsilon > 0$ time [9] and solves versions (1) and (2) of the problem as stated in the introduction.

[1] Potentially, problems could arise when $m = \infty$, but this case can be handled as a special case.

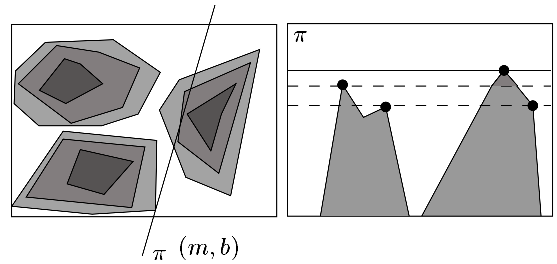


Figure 1: The diagram shows two views of the terrain. Left: Vertical projection of π and the terrain. The concentric polygons of different colours represent the contour lines of the terrain at different altitudes. Right: The vertical slice π of the terrain.

4 Query Data Structure for a Fixed Distant Horizon

We can use the output of the above described algorithm to query for a distant horizon given a horizontal viewing direction. In particular, once the upper envelope in (m, b, h)-space is known, a query data structure can be constructed in $O(n^{2+\epsilon} \lg n)$ time for any $\epsilon > 0$, to determine the edge fragments or the edges that lie on the distant horizon from a horizontal viewing direction. We construct this data structure by projecting the upper envelope down onto the mb-plane from above. The mb-plane is then divided into a number of regions, which are called *patches*, where a bivariate function $f_e(m, b)$ realizes the upper envelope. By determining the extent of a patch in the m direction, we create intervals of horizontal viewing directions for which an edge fragment appears on the distant horizon. We load these intervals into an interval tree [6] so that they can be efficiently searched. If we only want to know the edges that appear on the distant horizon, we merge overlapping intervals resulting from the same edge into a single interval. If we want to know the precise distant horizon, we store the equations defining the boundary of the patches with the interval.

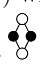
Given a horizontal viewing direction m, the s edges on the distant horizon can be determined in $O(\lg n + s)$ time since each of the s edges is reported exactly once. The precise distant horizon or edge fragments on it can be determined in $O(\lg n + k)$ time where k is the number of edge fragments on this distant horizon. The value of s is at most n and the value of k is at most $O(n\alpha(n))$.

5 3SUM Hardness of Computing All Distant Horizon Edges

In this section, we show that the problem of computing all distant horizon edges, or even determining the cardinality of this set, is 3SUM hard. This leads one to believe that algorithms with sub-quadratic run times to compute the solution to this problem are unlikely to exist. The 3SUM hardness reduction is from the *GeomBase* problem [8] and is similar to Gajentaan and Overmars' [8] reduction from

GeomBase to *Separator1*. The *Geombase* problem has the following definition:

GeomBase: Given a set of n points with integer coordinates on three horizontal lines $y = 0$, $y = 1$, and $y = 2$, determine whether there exists a non-horizontal line containing three points.

A terrain equivalent to an instance of the *GeomBase* problem is shown in figure 2. The $O(n \lg n)$ construction of this terrain is as follows. Sort the set of points on each of the lines $y = 0$, $y = 1$, and $y = 2$ by x-coordinate. Let $\max$ and $\min$ be the maximum and minimum x-coordinate. Between any two horizontally adjacent points (x_1, y) and (x_2, y) with $x_1 < x_2$, create a horizontal line segment with endpoints $(x_1 + \frac{1}{8}, y, 1)$ and $(x_2 - \frac{1}{8}, y, 1)$. For all points (x, y), place two vertices at $(x, y \pm \delta, 0)$ with $0 < \delta < \frac{1}{2}$. Connect these four points into structures  called *slits*. The rest of the construction is shown in figure 2.

If we intersect a horizontal plane with this terrain at any altitude between 0 and 1, we have a number of horizontal segments in the plane with a horizontal distance less than $\frac{1}{4}$ between any adjacent pair. It has been shown in [8] that this horizontal distance is sufficiently small to ensure that a line passes through one pair of segments on each of $y = 0$, $y = 1$, and $y = 2$, if and only if three points were collinear in the original instance of the *GeomBase* problem. Thus, a line passes through three slits if and only if three points in the original *GeomBase* problem were collinear. Once we have computed all distant horizon edges, we answer "yes" to the *GeomBase* problem if and only if an edge adjacent to a slit appears on the distant horizon. This can be done in $O(n)$ time and concludes our reduction.

Although this demonstrates the problem of determining all distant horizon edges is 3SUM hard, the reduction requires $\Omega(n)$ local maxima. It is still unknown if the problem is 3SUM hard when the terrain has a constant number of local maxima. It is relatively straight forward to show that there exists a terrain of m local maxima with $\Omega(mn)$ distant horizon edge fragments. However, in the next section, we demonstrate the problem of computing all distant horizon edge fragments takes $\Omega(n^2)$ time even for terrains with a single local maximum.

6 An $\Omega(n^2)$ Lower Bound for Computing all Distant Horizon Edge Fragments

In this section, we present a terrain T with a single local maximum and $\Omega(n^2)$ edge fragments. It follows immediately from this example that any algorithm to compute all distant horizon edge fragments would take at least $\Omega(n^2)$ time. The intersection T_h of terrain T with a horizontal plane $z = h$ gives a simply connected region. The points on the convex hull of this simply connected region are the points that support a horizontal visual line in this plane. We count the number of edge fragments by counting the number of times an intersection point between $z = h$ and an edge of T appears or disappears from the convex hull of T_h as the sweep plane $z = h$ is lowered through the terrain from highest to lowest altitude. These events correspond to the start or end of an edge fragment since the edge either begins or ceases to support a horizontal visual line at these points.

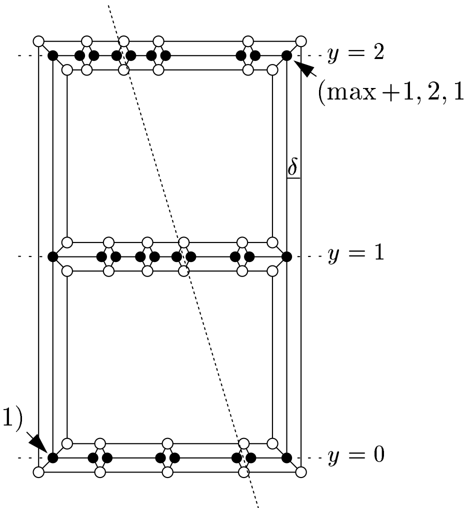


Figure 2: A terrain created from an instance of *GeomBase*. The solid vertices in the diagram are at height 1. The hollow vertices are at height 0.

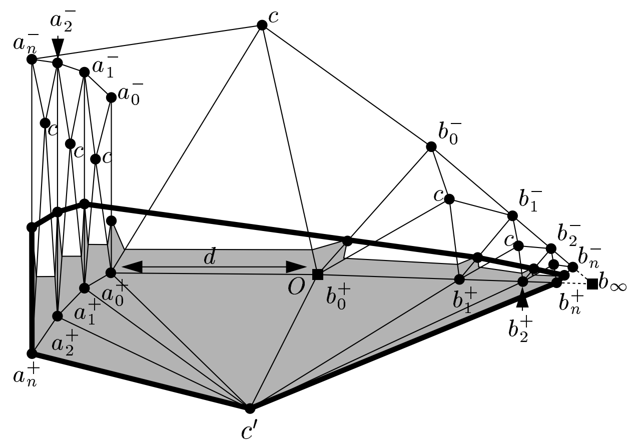


Figure 3: Top down view of the terrain T with $\Omega(n^2)$ edge fragments. Its intersection T_h with a horizontal plane at an altitude $z \in [h^+, h^-]$ is shown in grey and the convex hull of the intersection is shown as a thick black line.

In figure 3, the edges that cause the $\Omega(n^2)$ edge fragments have vertices labelled a_j^+, a_j^-, b_i^+, and b_i^- for all $i \in \{0, 1, 2 \ldots n\}$ and $j \in \{0, 1, 2 \ldots n\}$. The altitude of every vertex labelled a_j^+ or b_i^+ is h^+ and the altitude of every vertex labelled a_j^- or b_i^- is h^- with $h^- = h^+ - n$. All the vertices labelled c are at an altitude less than h^- and the