CONSTRUCTING SURFACE AREA BINARY SPACE PARTITIONING WITH ROPE TREES FOR RAY TRACING

Vlastimil Havran
Czech Technical University, Faculty of Electrical Engineering,
Department of Computer Science and Engineering,
Karlovo nám. 13, 121 35 Prague 2, Czech Republic
E-mail: havran@fel.cvut.cz

Keywords: computer graphics, ray tracing, BSP tree, surface area heuristics.

Ray tracing is well-known rendering technique for producing realistic images with reflective surfaces. The main drawback of this technique is rather big computational complexity, that disallows its interactive use. The main principle of the technique is simulating the light behaviour by means of geometrical optics: light interaction in space is modelled by rays given a point and direction. The principal expense of ray tracing is the determination of the closest ray-object intersection point. This problem is often referred to as ray-casting. It can be solved in time complexity \( O(N) \) using naive algorithm, where \( N \) is number of objects. Naive algorithm checks a given ray with all the objects and selects the nearest one. Unfortunately, complexity \( O(N) \) makes it unusable for practical applications.

There are techniques aimed to decrease the computational complexity of ray-casting. Their survey is given in [1], but it does not contain novel approaches invented recently. The algorithmic approaches that use spatial data structures can be classified into two groups: bounding volume hierarchies (bottom-up approach) and spatial subdivisions (top-down approach). The latter approach has been shown to attain better performance. Let us describe one of spatial subdivisions schemes in detail.

A Binary Space Partitioning (abbr. BSP) for a set \( S \) of objects in \( \mathbb{R}^d \) is a tree defined as follows: Each node \( v \) in BSP represents a box (rectangular parallelepiped) \( R_v \) and set of objects \( S_v \) that intersects \( R_v \). Leaf is such a node for which the number of objects \( |S_v| \) belonging to \( v \) is smaller than a specific constant or the depth of \( v \) in BSP is equal to maximal depth allowed. The box associated with the root of a BSP is \( R_\text{root} = \mathbb{R}^d \) itself. Each interior node of BSP is assigned cutting plane \( H_v \) that intersects \( R_v \) into two boxes. If we let \( H^+ \) be the positive halfspace and \( H^- \) the negative halfspace bounded by \( H_v \), the boxes associated with the left and right children of \( v \) are \( R_v \cap H^+ \) and \( R_v \cap H^- \), respectively. The left subtree of \( v \) is a BSP for the set of objects \( S^-_v = \{ s \in S_v | s \in R_v \} \). Right subtree is defined similarly. Let size of BSP be the number of interior nodes and the number of references to objects in all its leaves.

BSP is constructed hierarchically until termination criteria for leaf are reached. The cutting plane \( H \) is for ease of computing the intersection with a ray perpendicular to one of coordinate axes (orthogonal cutting). There is an efficient algorithm for traversing the BSP with orthogonal cutting planes by a ray [3].

The goal of construction algorithm is to construct BSP of small size assuming the termination criteria given are fulfilled. Usually, the cutting plane is positioned in a mid-point of the current axis; the axis is changed regularly in order \( z, y, \) and \( z \). Unfortunately, the placing of cutting plane in the spatial median of \( R_v \) is not convenient. Such choice does not consider geometrical positions of objects in \( R_v \) and therefore it can create large trees with poor performance in ray-tracing application. Note that, the objects cutting plane must be referenced in both child nodes. This results in the \( BSP \) of big size and in poor performance consequently.

To alleviate the problem \( BSP \) performance the inventive approach has been designed by MacDonald and Booth [2]. It takes into account the number of objects in left child node \( |S^-_v| \) and in right child node \( |S^+_v| \) using surface area heuristics. The heuristics expresses probability that a ray of arbitrary origin and direction pierces an convex object \( B \) if it passes through convex object \( A \) (see Fig. 1) assuming \( A \in B \).

Let \( A_\text{left}, A_\text{right}, \) and \( A_\text{root} \) be the surface area of \( R^-_v, R^+_v, \) and \( R_v \), respectively. The selection of cutting plane for BSP proposed in [2] is proceeded by maximizing the measure \( f_M = A_\text{left} / A_\text{root} + |S^-_v| / |S_v| + A_\text{right} / A_\text{root} + |S^+_v| / |S_v| \) for each constructed cutting plane \( H \). It supposes the cost \( f_c(S^-_v) \) of left (right) node corresponds the number of objects in child nodes \( f_c(S^-_v) = |S^-_v| \) and the passing of a ray through the whole \( BSP \) without piercing an object. We propose to consider the probability that a ray terminates in \( R^-_v \) or \( R^+_v \). Let us denote these probabilities \( p^-_v \) and \( p^+_v \). Let \( A^+_v \) be the surface area of cutting plane \( (A^+_v = A_\text{right} + A^+_v - A_\text{root}) \). Then we can define measure to be minimized as

\[
f_M = f_c(S^-_v) + (1 - p^-_v) f_c(S^+_v) + \frac{p^-_v}{|S^-_v|} f_c(S^-_v) + (1 - p^-_v) f_c(S^+_v) + \frac{p^-_v}{|S^-_v|} f_c(S^-_v)
\]

The measure \( f_M \) takes into account the probabilities, that a ray terminates in left or right node. The probabilities can be estimated for each cutting plane by incremental algorithm, whose description is out of the scope of this paper.

The further enhancement of construction \( BSP \) is to create empty space when it is advantageous. The leaf nodes containing no objects may be created using surface area heuristics, however, empty space is also present at leaf nodes containing objects. We separate such empty space in these non-empty leaf nodes only if it is advantageous using similar surface area based measure. The culling of these empty spaces results in decreasing the number of intersections computed on average. The situation is depicted in Fig. 2.

In order to improve the performance of \( BSP \) in ray-tracing further we adopted the approach of rope outlined in [2]. We use our own modification of a called rope tree, that creates from the \( BSP \) more general spatial structure. \( BSP \) with rope trees contains for each face of leaf node rope tree, that is a two-dimensional variant of \( BSP \). The leaves of a rope tree are assigned the links to neighbouring node \( w \), whose face intersects with given face of \( v \). The rope trees eliminates the hierarchical traversal steps by skipping the long traversal paths from nodes close to root to leaves.

The combination of acceleration techniques presented here decreases computational complexity by 20-38% in comparison with [2]. The size of \( BSP \) is also reduced in dependence of input scene. It remains open problem how to estimate the total cost \( f_c(S^-_v) \) of node \( R_v \) containing \( N \) objects assuming \( R_v \) is to be refined by constructing its \( BSP \). Solving the problem should further improve performance of \( BSP \) for ray-tracing.

References