Lecture 18: Data Structures – Review Questions

- Why do we want to build a hierarchical data structure such a bounding box hierarchy?
- Describe the main difference between the following two approaches: (1) construct a bounding volume hierarchy (e.g. bounding boxes or bounding spheres), (2) construct a hierarchy of splitting planes (e.g. KD trees or BSP trees).
- Describe a technique for constructing a bounding box hierarchy. What data structure will you use to store this hierarchy?
- Is your technique top-down or bottom up? Give an algorithm for the alternative approach (i.e., if your first approach was top-down, give a bottom-up approach to constructing a bounding box hierarchy).
- How do we do intersection testing between a ray and a bounding box? Note that we do not need to find the point of intersection, but we only need to determine whether or not the ray intersects the bounding box.
- How do we do intersection testing between a ray and a bounding sphere? Again, we only need to know whether the ray intersects the sphere. We do not need to find the specific intersection point.
- Explain in detail how to use a bounding box hierarchy to identify the intersection between a ray and the closest object in the environment. Your algorithm should, of course, be more efficient (in the general case) than the brute force process of checking the ray for intersection with all objects.
- An alternative to the bounding box or bounding sphere hierarchy is to use splitting planes to divide space. Octrees, KD trees, and BSP trees are all splitting plane algorithms. Describe the differences between these approaches.
- Describe how to construct a KD tree. Describe how to perform ray-object intersection using this data structure. Your ray-object intersection algorithm should check regions from front to back order, so that the search for an intersection may be halted when an intersection point is found.
- In the KD tree scenario, what problem is introduced when the splitting planes intersect some of the objects? Assume that you do not want to use the plane to split the objects (e.g., splitting a sphere into two parts may not be desirable for ray tracing). How can you solve or work around this problem?
- Describe how to construct a BSP tree and how to do ray-object intersection testing in front to back order with this data structure.
- Constructive solid geometry (CSG) can be used to create more complex objects out of simple primitives using union, intersection, and difference operations. Sketch a CSG object that combines union, intersection, and difference operations. Draw the hierarchy of operations that must be performed to create that object.
- Given ray-object intersection points for each of the primitive objects used to create your CSG object, show how to find ray-object intersection points that result from each of the operators in the hierarchy (union, intersection, difference).