Real-Time 3D Rendering

As much as computer graphics have progress in the number of years on the number of billions of triangles drawn per second, real-time graphics rendering will always pose a problem because there can always be more. As our hardware reaches its limit, the frame rate suffers as a result. It then becomes clear that hardware is not the only solution. Therefore, we as programmers must produce higher performance software. We need to implement speed-up techniques and algorithms to perform this very task. My proposal is to discuss this very topic of the various techniques that have been developed over the last years. This will mainly include spatial data structures such as bounding volume hierarchies, BSP trees, octrees, and scene graphics. Spatial data structures can indirectly aid a programs rendering performance. The technique of culling utilizes the various spatial data structures to remove triangles not visible or do not contribute to the rendering. This in turn can then free up the video bus from sending useless data such as textures and vertices.

References

Thomas Akenine-Moller, Eric Haines, Real-Time Rendering, 2nd Edition, A K Peters, 2002.

David Eberly, 3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics, Morgan Kaufmann, 2000.

Gil Gribb, Klaus Hartmann, "Fast Extraction of Viewing Frustum Planes from the World-View-Projection Matrix," June 2001.
http://www2.ravensoft.com/users/ggribb/plane%20extraction.pdf

* Jaap Suter, Introduction to Octrees,
http://www.flipcode.com/tutorials/tut_octrees.shtml

* Jacco Bikker, Building a 3D Portal Engine,
http://www.flipcode.com/portal/