Errata

- **Page 121:** The paragraph "PVS-based arbitrary geometry occlusion culling. [...]" was sourced from "P. Laurila. Geometry Culling in 3D engines. *Graphics and GPU Programming*. `gamedev.net`, 2000." and is changed as follows:

  PVS-based geometry occlusion culling. PVS-based occlusion culling algorithms designed to handle complex and arbitrary geometry in 3D scenes typically extend traditional PVS-based occlusion culling techniques to accommodate complex scenes with intricate and diverse geometric shapes. These algorithms determine the visibility of arbitrary shapes accurately from a given point of view by approximating the PVS for each cell or region in the scene. Proper handling of complex geometry (i.e., non-convex shapes and irregular surfaces) and occlusion relationships (i.e., involving transparency and translucency) is typically needed. Appropriate support for dynamic updates to the PVS determination is important for real-time or interactive applications where the perspective and the viewport may change rapidly. Schaufler et al. [Sch+00] presented a conservative approach to determine volume visibility that works on a discrete representation of the scene using an octree. The opaque interiors of objects are used as occluders to mark opaque octree nodes, and visibility is determined from a particular viewpoint by computing the occluded nodes in a conservative fashion. Durand et al. [Dur+00] introduce an alternative algorithm that works on a pre-processing stage to compute the PVS using "extended projection operators", which project occluder objects onto a group of projection planes. This enables the efficient occlusion culling computation from all locations within a region.