Hierarchical View-Frustum Culling for Z-buffer Rendering

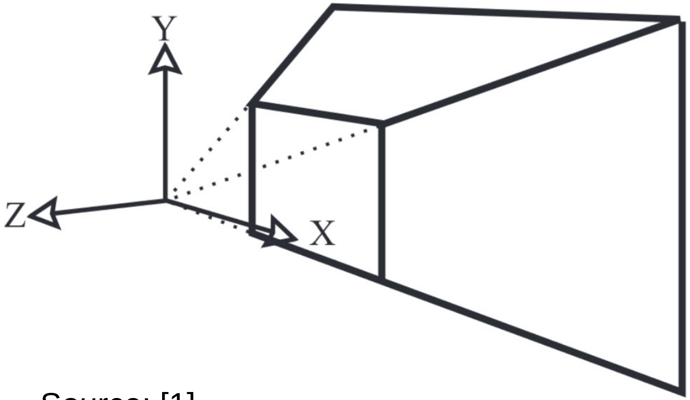
Tomáš Bubeníček

Contents

- What is view frustum culling
- Algorithms used
- How to speed it up

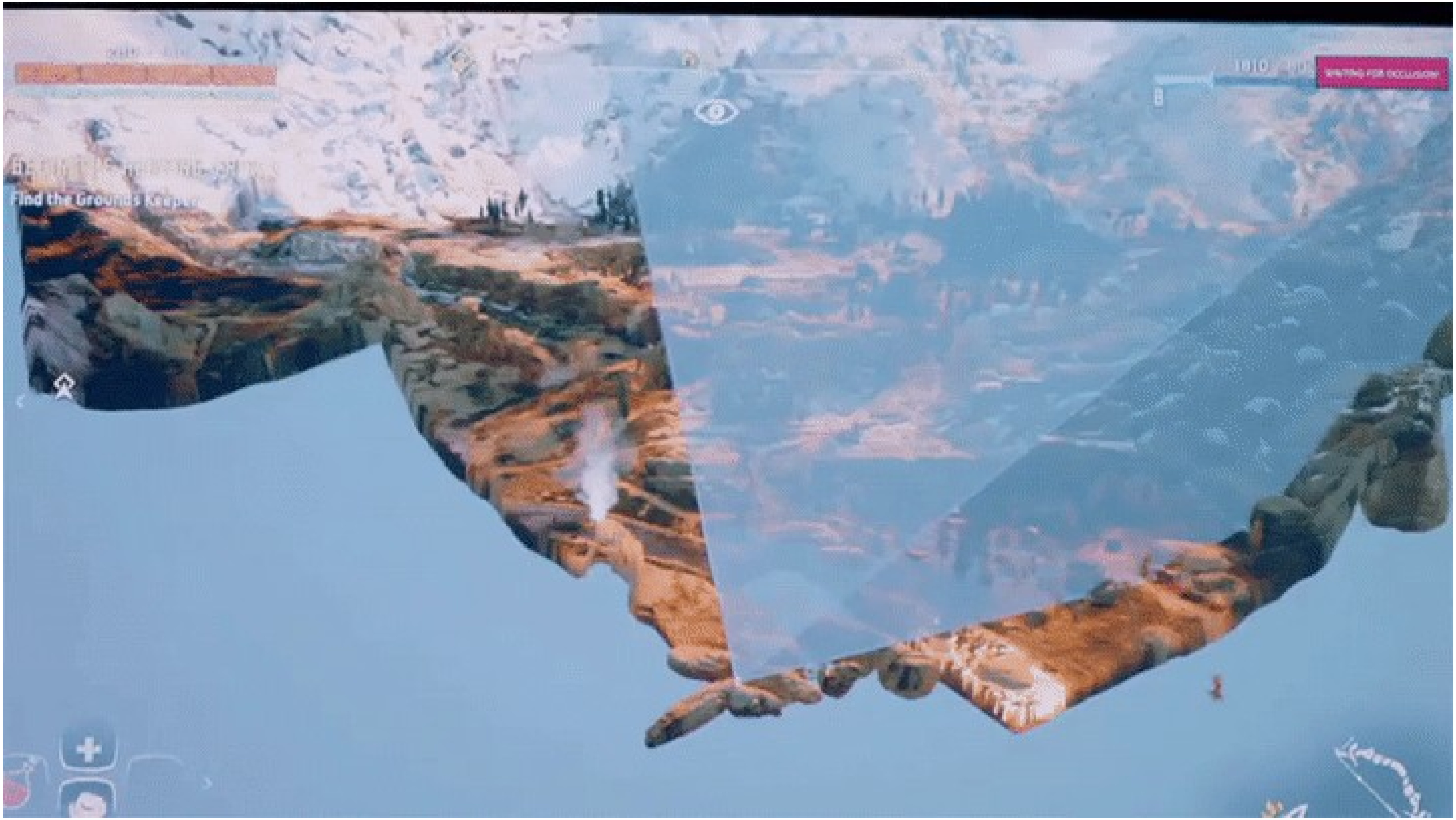
What is frustum culling

- Used with rasterization
- Removing objects that are outside the visible volume from the rendering process
- Saves vertex shader work on the GPU
- Visible volume = View frustum



Source: [1]

Example



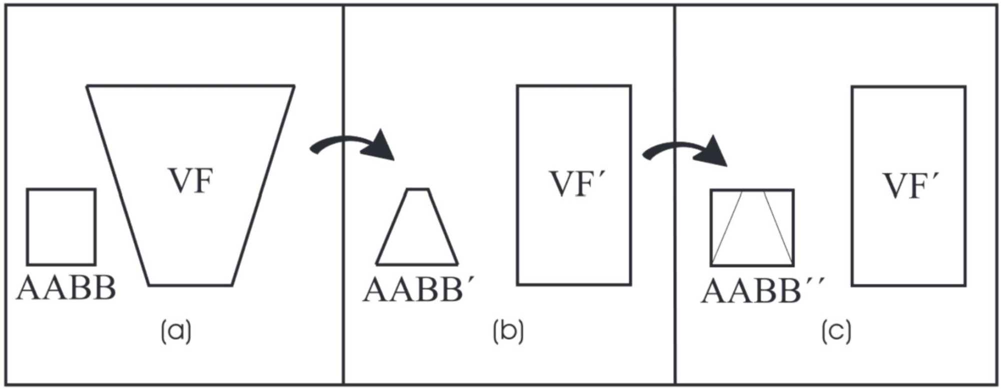
Source: Horizon Zero Dawn – The making of the game (2017)

Hierarchical approach

- Using bounding volume hierarchy
- 3 options when checking a BV vs frustum:
  - Inside – render the entire subtree
  - Outside – discard the entire subtree
  - Intersects – traverse the subtree, render if leaf
- We require a fast BV-frustum intersection predicate

Node-Frustum intersection

- One approach:
  - Transform all vertices of the node with the projection transform
  - Test AABB of the transformed envelope with AABB of the transformed frustum (tested in perspective coordinate system)
  - Relatively expensive
    - Transform requires 72 multiplications (sped up with SIMD)
    - Test requires only 6 comparisons



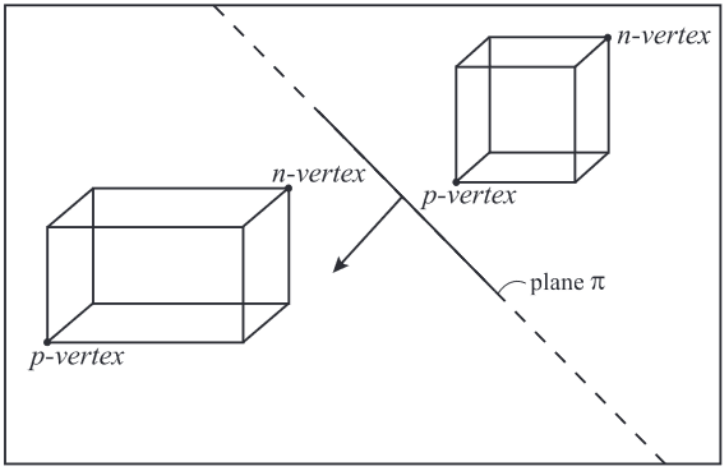
Source: [1]

Node-Frustum intersection

- Different approach:
  - Consider frustum as six separate planes
  - Requires fast box-plane intersection predicate
  - Calculated in world space
  - If object is inside the frustum, it is inside all six planes
  - Test can correctly end if object is outside just one plane

Box-plane intersection

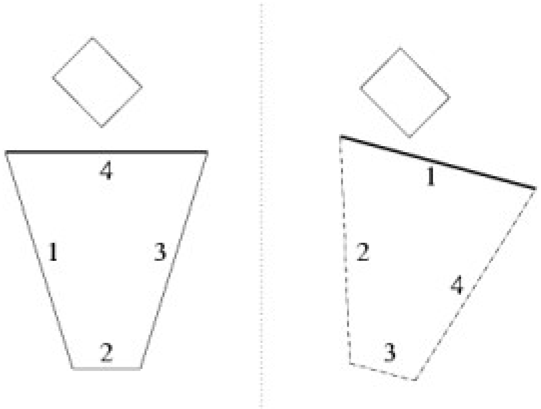
- Conservative approximation – Sphere-plane intersection
- Fast box plane intersection
  - Compare only 2 points instead of 8
    - n-vertex (negative-far point)
    - p-vertex (positive-far point)
  - The points are selected as the points on the diagonal that is closest to the plane normal
    - When using AABB, the selection can be done with a lookup table
  - Comparing the distances of points from the plane
    - If n positive, then outside
    - If p negative, then inside
    - Else intersects



Source: [1]

Order of plane tests

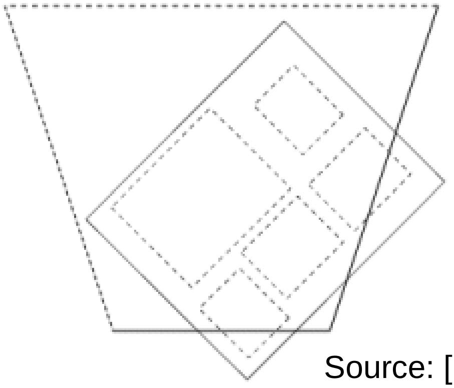
- The order of planes tested can affect performance
- We first want to test planes with high chance of the object being outside
- Temporal coherency
  - Object is outside a plane in one frame - chance of happening again
  - Start the next test with that plane



Source: [2]

Number of plane tests

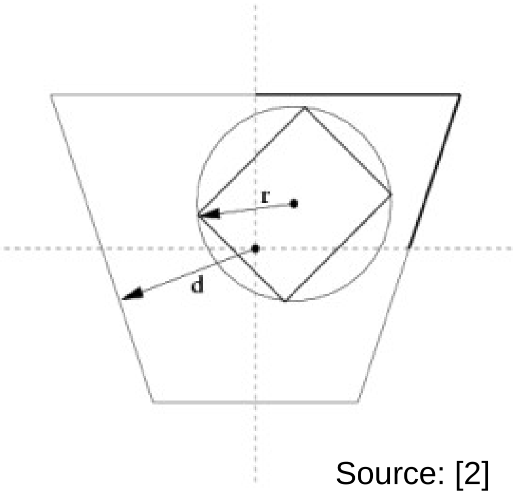
- When traversing the hierarchy, we don’t need to always check all planes
- Plane-masking
  - Only check the planes that the parent node intersects
  - When recursively traversing, keep a bit mask of which planes to check



Source: [2]

Other possible speedups

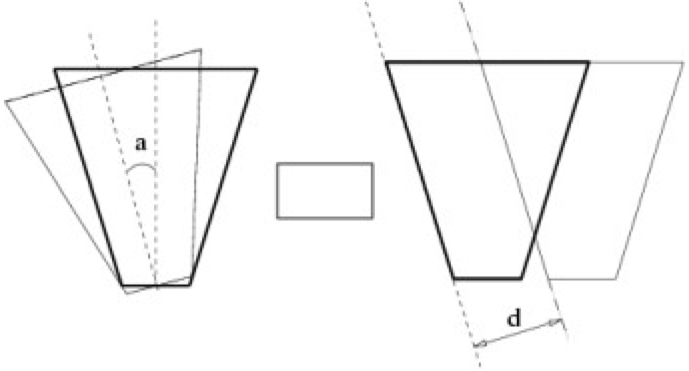
- Octant test
  - If radius of a bounding sphere around the AABB is smaller than smallest distance from frustum center to plane, we can discard three plane tests



Source: [2]

Other possible speedups

- Rotation coherency
  - Frustum only rotates by less than 180° on x/y/z axis between frames
  - For each BV, we can return outside if outside in last frame and their distance to plane increased
- Translation coherency
  - Frustum only moves between frames
  - For each plane, the distance to each BV changes by d
  - We compare the distance in last frame with d, if d > dist, then intersection has to be computed



Source: [2]

Recap

- View frustum culling is a way of speeding up rasterization
- Can be sped up by using hierarchical data structures
- Frustum-volume intersection can be computed both in world coordinates and projection coordinates
- During the plane intersection tests
  - Order of planes matters
  - Some planes don’t need to be always tested

Sources

- [1] Ulf Assarsson, and Tomas Moller. "Optimized view frustum culling algorithms for bounding boxes." Journal of graphics tools 5.1 (2000): 9-22. Online: http://www.cse.chalmers.se/~uffe/vfc_bbox.pdf

- [2] Daniel Sýkora, and Josef Jelínek. "Efficient View Frustum Culling." Central European Seminar on Computer Graphics. 2002. Online: http://old.cescg.org/CESCG-2002/DSykoraJJelinek/index.html

- Horizon Zero Dawn – The making of the game (2017). vpro documentary. Online: https://www.youtube.com/watch?v=A0eaGRcdwpo