Assignment 9

Basic Techniques in Computer Graphics
WS 2022/2023
January 13, 2023

Frederik Muthers	412831
Tobias Broeckmann	378764
Debabrata Ghosh	441275
Martin Gäbele	380434

Exercise 1 Ray-Quadric Intersection

(a) Surface Description We consider the surface in $\mathbb{R}^3$ defined by the quadric

$$\mathbf{Q} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Using the implicit form of the quadric we get

$$\begin{aligned} \begin{pmatrix} x & y & z & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} &= 0 \\ \implies \begin{pmatrix} x & y & z & 1 \end{pmatrix} \begin{pmatrix} 0 \\ y \\ z \\ -1 \end{pmatrix} &= 0 \\ \implies y^2 + z^2 - 1 &= 0 \\ \implies y^2 + z^2 &= 1 \end{aligned}$$

So, we have an infinite cylinder through the origin $(0,0,0)^\top$ defined by its axis $(1,0,0)^\top$ i.e. the x-axis with radius 1 defined as $y^2 + z^2 = 1, x \in \mathbb{R}$.

(b) Ray Intersection We intersect the above surface with the ray $\mathbf{c} + \lambda\mathbf{r}$, where $\mathbf{c}$ is given as $(0,0,\sqrt{2})^\top$ and $\mathbf{r} = (r_x, r_y, r_z)^\top$ is a general direction.

The point of intersection lies on the quadric whose implicit form gives us the following equation

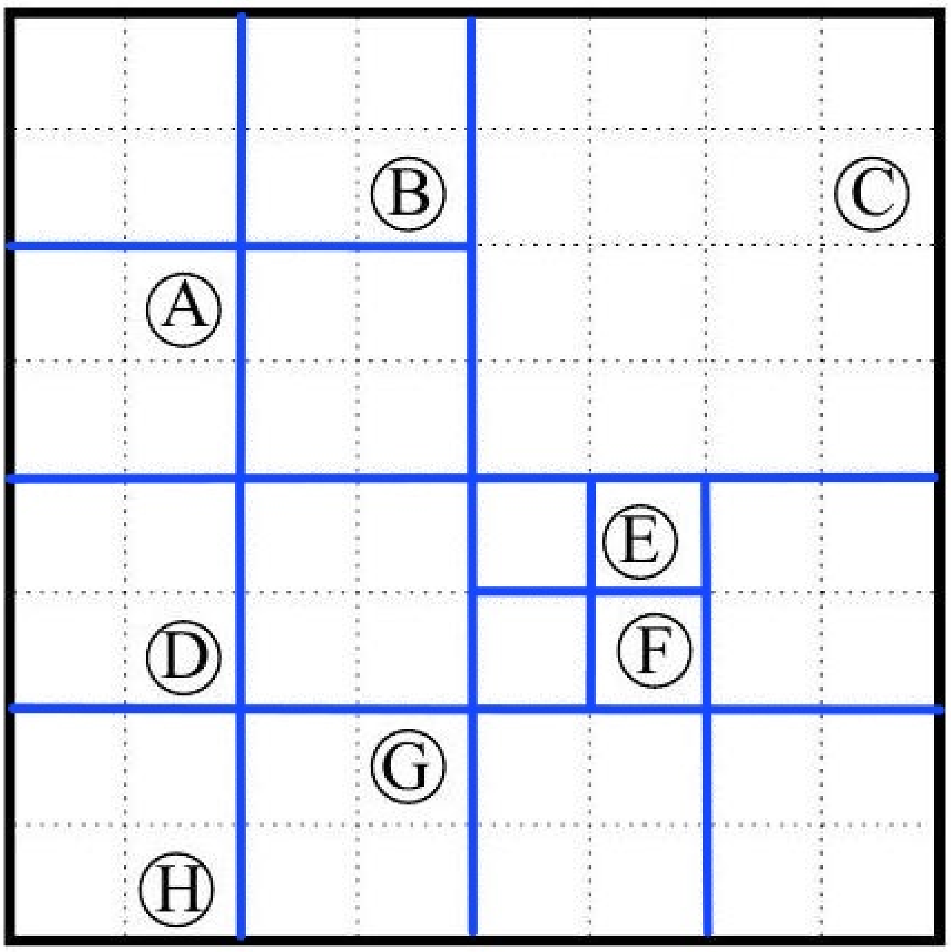
$$\begin{aligned} (\mathbf{c} + \lambda\mathbf{r})^\top \mathbf{Q} (\mathbf{c} + \lambda\mathbf{r}) &= 0 \\ \implies \begin{pmatrix} \lambda r_x & \lambda r_y & \sqrt{2} + \lambda r_z & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} \lambda r_x \\ \lambda r_y \\ \sqrt{2} + \lambda r_z \\ 1 \end{pmatrix} &= 0 \\ \implies \begin{pmatrix} \lambda r_x & \lambda r_y & \sqrt{2} + \lambda r_z & 1 \end{pmatrix} \begin{pmatrix} 0 \\ \lambda r_y \\ \sqrt{2} + \lambda r_z \\ -1 \end{pmatrix} &= 0 \\ \implies \lambda^2 r_y^2 + 2 + 2\sqrt{2}\lambda r_z + \lambda^2 r_z^2 - 1 &= 0 \\ \implies \lambda^2 (r_y^2 + r_z^2) + 2\sqrt{2}\lambda r_z + 1 &= 0 \\ \implies \lambda &= \tfrac{-2\sqrt{2} r_z \pm \sqrt{8 r_z^2 - 4(r_y^2 + r_z^2)}}{2(r_y^2 + r_z^2)} \\ \implies \lambda &= \tfrac{-2\sqrt{2} r_z \pm \sqrt{4 r_z^2 - 4 r_y^2}}{2(r_y^2 + r_z^2)} \end{aligned}$$

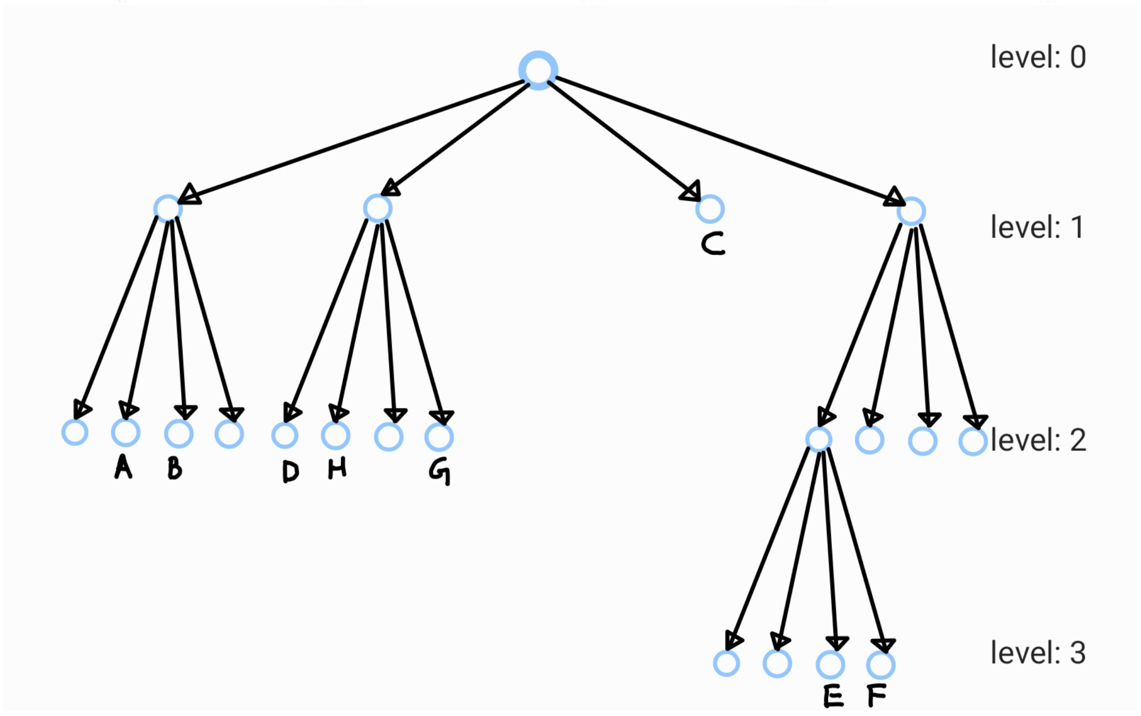
$$\implies \lambda = \frac{-2\sqrt{2}r_z \pm 2\sqrt{r_z^2 - r_y^2}}{2(r_y^2 + r_z^2)}$$
$$\implies \lambda = \frac{-\sqrt{2}r_z \pm \sqrt{r_z^2 - r_y^2}}{r_y^2 + r_z^2}$$

(c) Example Now if we consider $\mathbf{r}$ to be given as $(0, 1, -1)^\top$, then we have $r_y = 1$ and $r_z = -1$, which gives us the the values of λ as: $\lambda = \frac{-\sqrt{2}(-1) \pm \sqrt{(-1)^2 - 1^2}}{1^2 + (-1)^2} = \frac{\sqrt{2} \pm \sqrt{1-1}}{1+1} = \frac{\sqrt{2} \pm 0}{2} = \frac{\sqrt{2}}{2}$. Since, we get only one value of λ, the intersection point is also tangential to the quadric surface (infinite cylinder). Also, the point of intersection is $(0, 0, \sqrt{2})^\top + \frac{\sqrt{2}}{2}(0, 1, -1)^\top = \left(0, \frac{\sqrt{2}}{2}, \sqrt{2} - \frac{\sqrt{2}}{2}\right)^\top = \left(0, \frac{\sqrt{2}}{2}, \frac{2\sqrt{2} - \sqrt{2}}{2}\right)^\top = \left(0, \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right)^\top$.

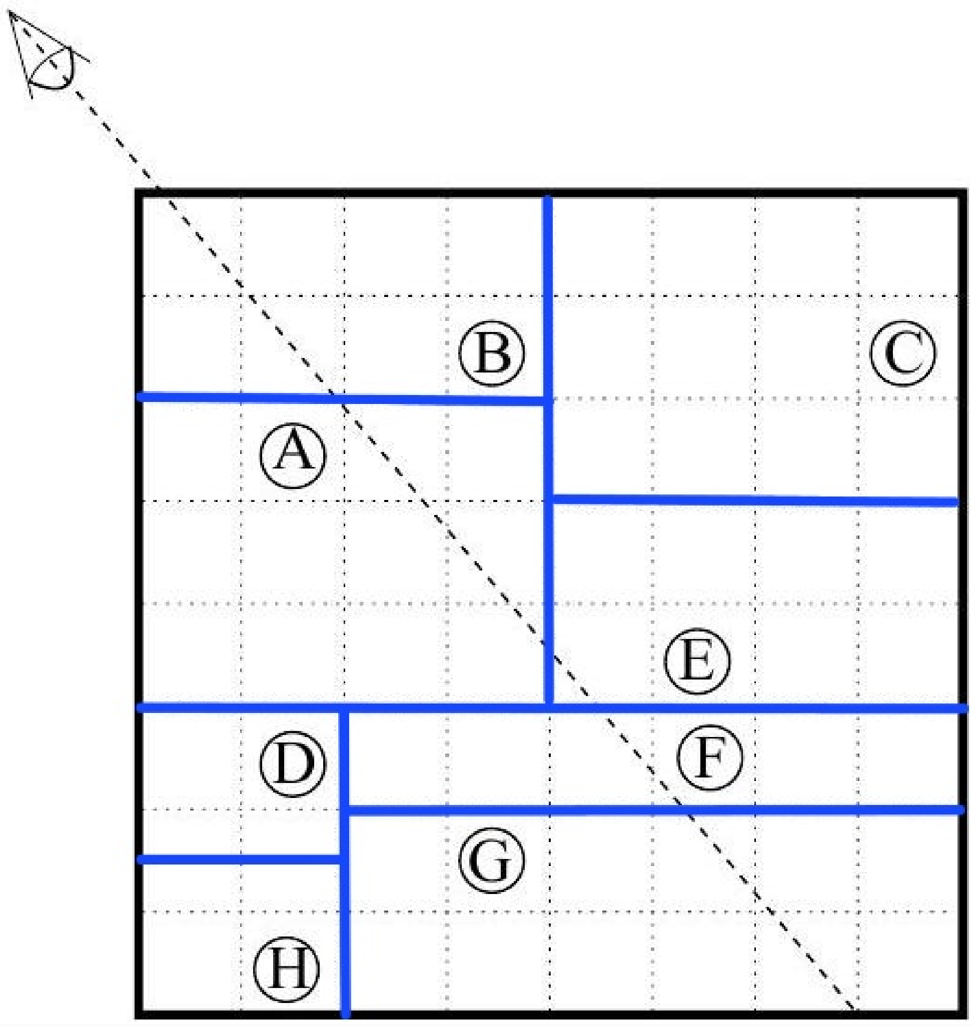
Exercise 2 Spatial Data Structures

(a) Quad-Tree

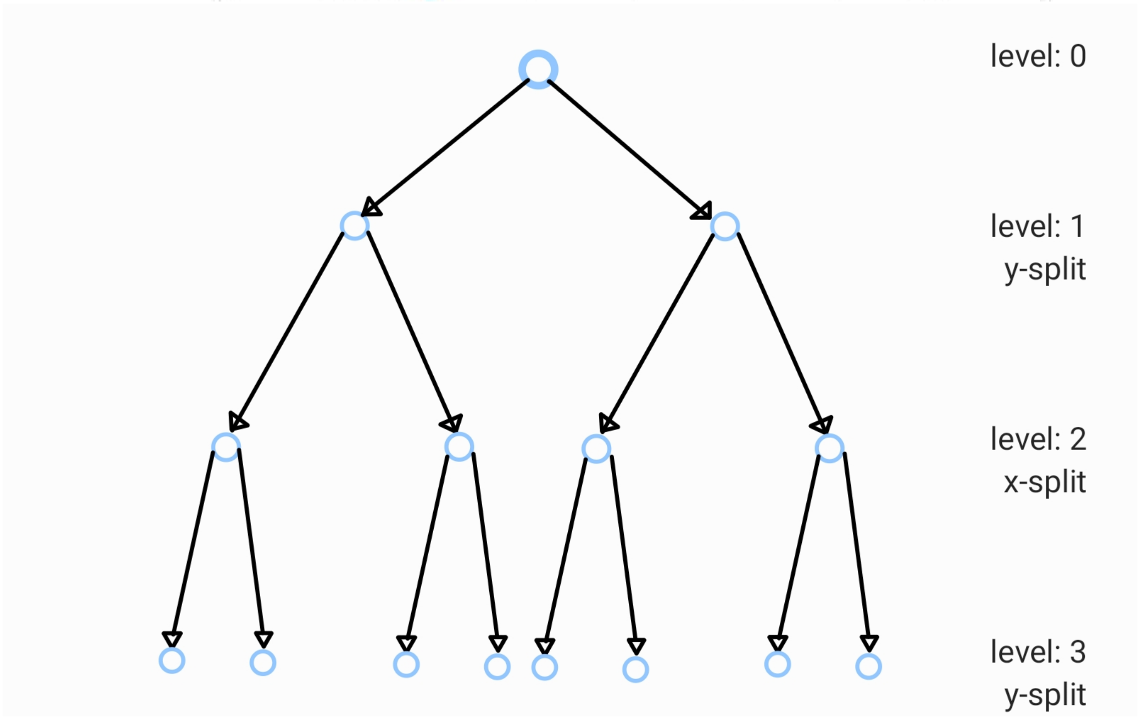




(b) kD-Tree

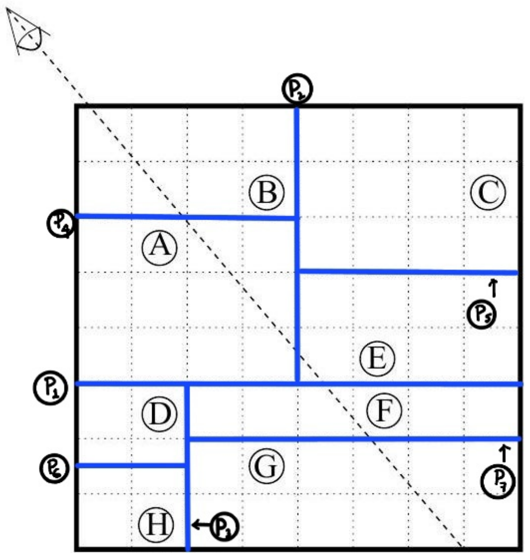


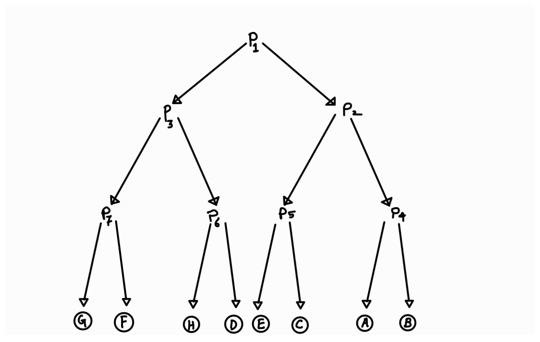
B
C
A
E
D
F
G
H



level: 0
level: 1
y-split
level: 2
x-split
level: 3
y-split

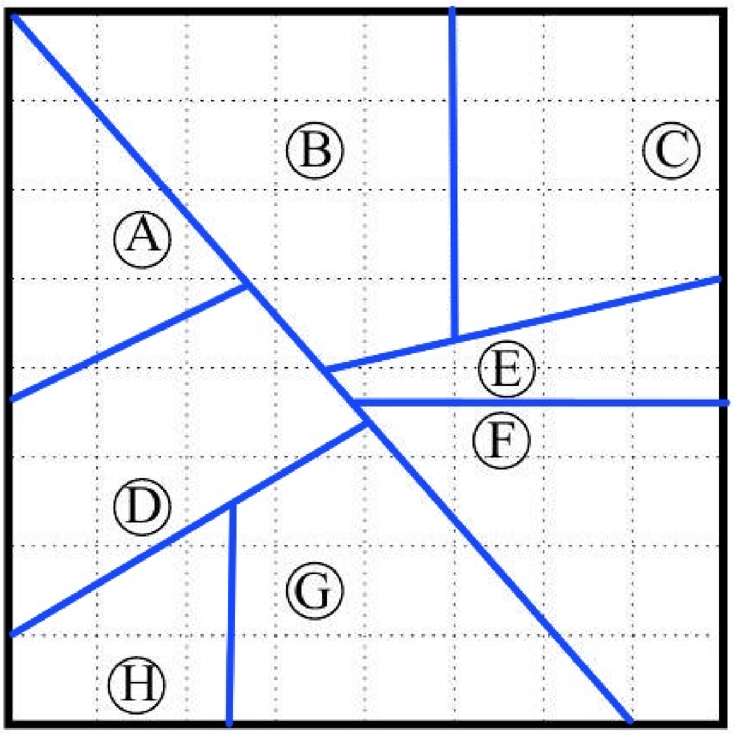
(c) Painter's Algorithm

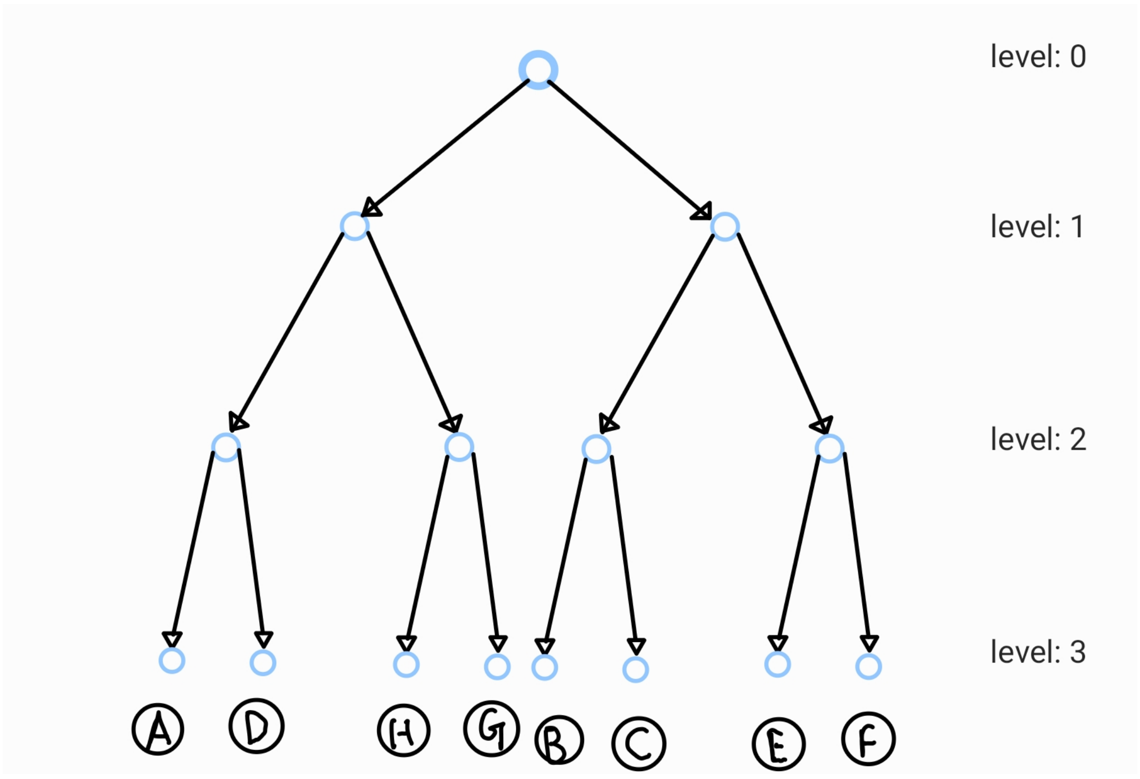




From the above hierarchical structure of partitioning made using the kD-tree constructed in 2c), we can determine the order in which we can render or draw the objects to implement Painter's Algorithm as the object farthest from the camera is situated leftmost in the hierarchical tree and then we follow the hierarchy to the right to get the following order: G, F, H, D, E, C, A, B.

(d) Binary Space Partition



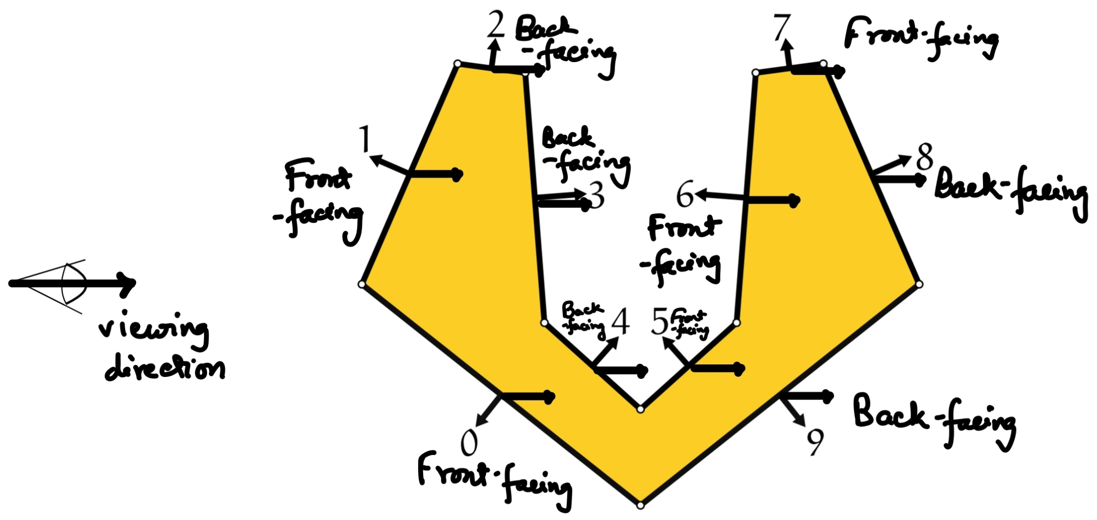


(e) Comparison The main difference between BSP-trees and kD-trees is in the way we choose the partitioning place. While we select splitting planes alternatively along the dimensions with kD-trees, with BSP-trees we can choose them arbitrarily given that the resulting tree is balanced.

Although kD-trees have better balancing than something like octrees or voxel grids, BSP-trees have the best balancing among them. But as a disadvantage, the tree construction is usually more complex for BSP-trees.

Exercise 3 Culling

(a) Backface Culling Example



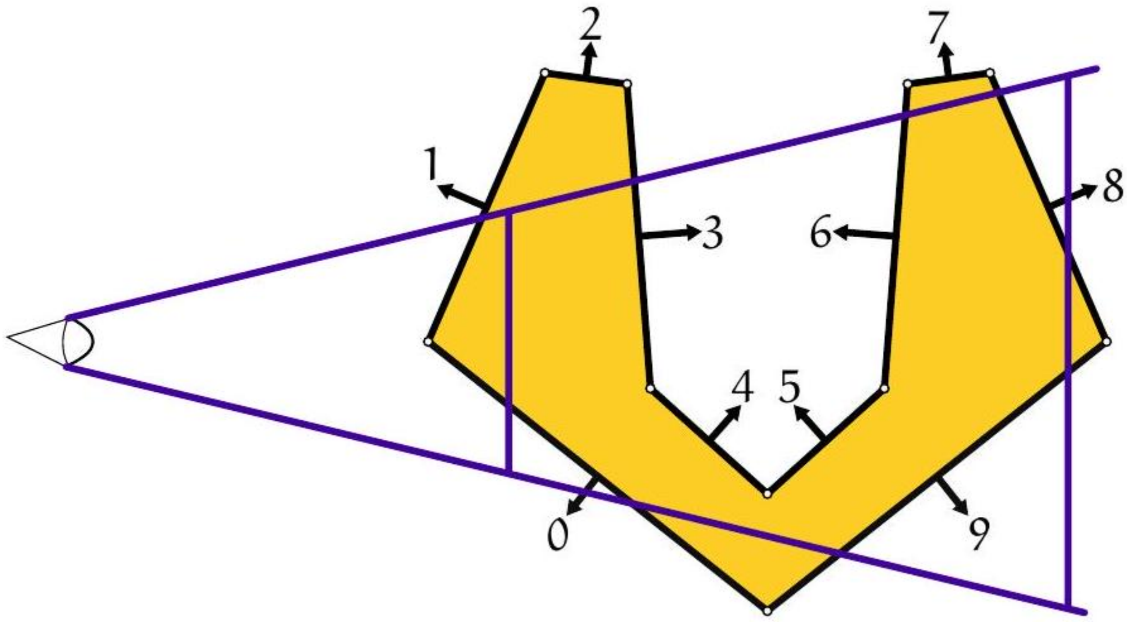
Triangle	BF Culled
0	×
1	×
2	✓
3	✓
4	✓
5	×
6	×
7	×
8	✓
9	✓

(b) Backface Culling If we perform backface culling after applying frustum transformation instead of performing it in camera space, we could have perspective shortening of the object and corresponding triangles. In such scenario, normal vectors which were aligned at an angle close to 90 degrees might fall in the other direction of the spectrum after the transformation and the triangle in that case can become backfacing from frontfacing or vice versa. Our strategy of backface culling in our example might see some changes, particularly with triangle number 7 which could become backfacing.

(c) View Frustum Culling and Clipping In view frustum culling we cull away anything outside the viewing frustum as it will not be visible from the current viewing position and hence can be discarded from the rendering process. We create a bounding volume hierarchy to perform view frustum culling efficiently. Each node with a bounding volume is tested against the frustum. and if the bounding volume of the node is outside the frustum, then that node is not processed further.

Clipping usually refers to polygon clipping. Given an arbitrary polygon, we have to determine the parts of this polygon that are inside the viewing frustum which reduces to clipping the polygon's edges. Such methods also perform view frustum culling for polygons but the idea of view frustum culling is for whole objects at once.

(d) View Frustum Culling and Clipping Example



Triangle	VF Culled	Clipped
0	×	✓
1	×	✓
2	✓	-
3	×	✓
4	×	×
5	×	×
6	×	✓
7	✓	-
8	×	✓
9	×	✓

(e) Occlusion Culling Occlusion culling tries to cull away objects that are occluded i.e. hidden by other objects in the scene. The optimal occlusion culling algorithm would select only the objects that are visible without sending all objects through the rendering pipeline.

One method of occlusion culling is using point-based visibility to keep what can be seen from a single viewing location which is usual for rendering. In any standard occlusion culling algorithm for every object if it's not occluded (depending on the method such as point-based visibility) we render the object and add it to the list of objects which can occlude other objects else we cull away. Also, the rendering order is front-to-back.

Exercise 4 Color Models

(a) Metamerism is the effect in which physically different colors are perceived as the same colors by the human eye.

(b) The primary motivation is the analogy to the way the human eye works. The human eye also perceives three base colors via red, green and blue cones.

(c) One hardware oriented color model is the RGB model. This model is most commonly used in computer screens. There it essentially arises out of the base colors in the screen which are taken as the base red, green and blue and every color that can be displayed now is a linear combination out of the three within a unit-cube.
A user oriented color model is the HSV model, which models color by the hue, saturation and value. This model is more intuitive for users which for example makes it more fitting for an artist choosing or describing their colors.
A scientifically founded color model is the CIE-Model which is similar to an RGB-Model in which the red, green and blue (called x,y and z) are chosen in order to be able to represent all the colors the human eye can perceive via positive linear combinations. These base colors are more artificial than the usual RGB colors however normally the colors used in the hardware-oriented RGB model cannot represent every color the human eye can perceive unless one would be able to take negative values of the base colors which in practice is usually impossible (for example when the base colors are the base colors of a computer screen).

(d) An additive color model is the RGB model, a subtractive color model is the CMY color model. In an additive color model we can create the white color by adding all colors and the black color by adding no color. In a subtractive color model however we create black by adding all colors and white by adding none.

(e) A (linear) RGB color is not gamma corrected. The human eye perceives changes in brightness in a non-linear way, perceiving ratios instead of differences in brightness. This means that for example a picture taken by a camera has to get gamma corrected in order for the brightness of the picture to appear realistic to the human eye. This is done by applying an exponential scale to the brightness which is done in sRGB pictures but not in normal RGB pictures. In order to convert an RGB image to an sRGB image we map the brightness levels to an exponential scale.

(f) The DSLR camera takes a sequence of pictures with increasing exposure time to then compose this sequence into a single HDR image.