**Graphics Pipeline**

*What are the steps of the Graphics Pipeline? What does a Geometry Shader do?*

**Application**: The Application Stage is usually done once and is used to generate the Database. It repeats the main loop until the exit.

**Geometry**: The geometry stage is where the vertices of the 3D models are processed and transformed into 2D projections. This includes operations such as translation, rotation, scaling, and perspective projection.

**Rasterization**: The rasterization stage is where the 3D models are transformed into a 2D representation by determining which pixels of the display they cover. This stage is responsible for determining the visible portions of the models and the colour of each pixel in the final image.

**Frame Buffer**: The frame buffer is a memory area used to store the final image before it is displayed on the screen. The contents of the frame buffer are updated after each frame is rendered and represent the final image that will be displayed.

**Display**: The display stage is the final step in the graphics pipeline and is responsible for showing the final image on the screen. This stage typically involves a display controller that communicates with the GPU and the display hardware to transfer the image from the frame buffer to the screen.

**Geometry Buffer**: A geometry shader is a programmable stage in the graphics rendering pipeline that operates on individual primitives (e.g., points, lines, triangles) and has the ability to generate new primitives. It is run between the vertex shader and the pixel shader.

*What is double buffering? What is V-Sync?*

At the display stage the GPU needs to be synchronized with the display, or you can get artifacts. Therefore, proper buffering is needed. The display swaps between scan out and display.

The GPU swaps between **clear + draw** and **Idling**. If the frame buffer is changed during the displays scan out artifacts like flickering will occur. This is called **single buffering**.

A better solution to this problem is to use more than one frame buffer -> **Double buffering**. The frame buffer is divided into **front and back buffers**. While the front buffer is scanned out and displayed on the screen, the GPU calculates the next frame in the back buffer. After this calculation is complete, the execution of the swap command is done. There is no flickering with this method, but tearing can appear when swapping happens during scan-out.

VSync, popularly known as Vertical Sync, is used to avoid swapping the buffers during scan-out, thus the display is done with the frame scan-out from the GPU. This avoids Screen Tearing. If the rendering is slow, frame rates can only be integer fractions of the display rate which results in a lot of additional latency and lost frames. Adaptive Sync can be used to turn of V-Sync when rendering is slow.

**ShaderProgramming**

*What is a command buffer and a queue (in Vulkan or DirectX12)? What is the main difference to a traditional rendering context (in OpenGL/DirectX11)?*

Commands in Vulkan, like drawing operations and memory transfers, are not executed directly using function calls. Before performing an operation, they need to be recorded in the **command buffer**. The advantage of this is that when we are ready to tell the Vulkan what we want to do, all of the commands are submitted together and Vulkan can more efficiently process the commands since all of them are available together. In other terms, the optimization and validation happen during building and not during the submission. Furthermore, it is an immutable and reusable pipeline state configuration and the selected pipeline state variables can be declared mutable.

In addition, this allows command recording to happen in multiple threads if so desired. Many command buffers can be built from many threads simultaneously and each buffer switches to its favourite pipeline state. Synchronization primitives can be used across command buffers (barrier, semaphore, fence, event,..).

After the command buffers are submitted to the Queue. The command Queue is used to schedule the command buffers.

In a traditional rendering context, the pipeline execution is largely asynchronous. The configuration of state can only be done incrementally. The pipeline state is not made explicit in the rendering context which can lead to bottlenecks. The pipeline abstraction is single-threaded on the CPU.

*What is a vertex shader? Name two examples?*

It is a shader which processes each vertex. The **input vertex stream** is transformed into a **stream of vertices mapped onto screen** by the vertex shader. It is a programmable part of the graphics pipeline. The vertex coordinates get transformed via **the Model-View-Projection matrix.**

Example: Blinn-Phong Shading,

*What is a pixel shader? What does it do?*

Pixel shaders are used to modify the fragments to be rendered, for example to achieve a more realistic representation of surface and material properties or to change the texture representation. The pixels of the final image may result from multiple fragments, for example, when multiple objects can be seen simultaneously due to transparency. The shaders process these objects independently, so the term fragment shader is actually the correct term, however the term pixel shader has become common in the Direct3D environment, in OpenGL it is more correctly called fragment shaders.

**Shading Models**

*What are the three components of the Cook-Torrance shading model (called D, F, and G in the lecture materials)? Explain in words what physical characteristic of a surface each component describes (you do not have to give any formulae)*

The Cook-Torrance Shading Model is a Model which puts Microfacet distribution, Fresnel reflectance and Geometric attenuation together.

**D Microfacet distribution**: A microfacet is a tiny facet of the surface of an object being rendered, used in approximating reflections. In other words, the surface is assumed to be made of tiny mirrors. The distribution for those mirrors needs to be modelled. The microfacet distribution accounts for the mirrors which reflect light back at the viewer.

**F Fresnel reflectance**: Fresnel reflectance is the amount of light reflected by a surface at an interface between two materials with different refractive indices, based on the angle of incidence and the polarization of the light. It's a key factor in determining the overall reflectance of the surface, especially for transparent materials like glass or water.

**G Geometric attenuation**: Accounts for shadowing and masking of facets by one another. Light can be blocked from reaching microfacets by other microfacets or reflected light can be blocked from reaching the camera by other microfacets.

*What does Blinn-Phong shading do?*

Blinn-Phong is a shading model which approximates energy conversation. It is an extension to the Phong shading model and is used for surface shading. The Blinn-Phong model is largely similar to the Phong model, but approaches the specular model slightly different which as a result overcomes the problem of the circular highlights. Instead of relying on a reflection vector we're using a so-called halfway vector that is a unit vector exactly halfway between the view direction and the light direction. The closer this halfway vector aligns with the surface's normal vector, the higher the specular contribution.

**Textures**

*What is Mip Mapping?*

Mip mapping is a down sampling technique which is heavily used in real-time graphics. The name comes from the Latin phrase “multum in parvo” (mip) which means much in a small area. The texture is precalculated for different sizes and is reduced by factors of two. This is a simple and memory efficient technique to dynamically adjust the resolution of the texture. The last image is only one texel big.

What is texture mapping?

Texture mapping is a technique that involves mapping a 2D image onto a 3D model's surface. This provides a way to add detail and realism to the surface of 3D models by simulating the appearance of materials.

*What are texture coordinates used for? Which methods for creating texture coordinates do you know?*

Texture coordinates are coordinates of a 2D texture (uv space) which are used to map portions of textures to the given fragment of an object. Planar-, Cylindrical-, Box- and Polar Parametrization.

*Multipass rendering. Methods and applications?*

Multiple-pass rendering is a process in which an application traverses its scene graph multiple times in order to produce an output to render to the display. Multiple-pass rendering improves performance because it breaks up complex scenes into tasks that can run concurrently.

The two main methods are: Render to the auxiliary buffers and use the result as texture and redraw the scene using fragment operations. Both of these methods can be mixed.

There are many possible effects with multiple passes such as Environment maps, Shadow maps, Reflections, Transparency

*What is the alpha buffer? How does blending work? What do you have to consider if transparent and non-transparent objects are shown together?*

The alpha buffer is used to measure opacity of a texture. It is used to simulate translucent objects like glass or water.

Blending combines new pixels with what is already in the framebuffer using a blend equation. Linear Blending is the most common technique. Depth sorting and back to front blending is required if transparent and non-transparent object are shown in the same scene.

*What is Light Mapping?*

Light mapping is a technique that precomputes the lighting of a 3D scene and stores it in a texture map, called a light map, which is then applied to the surfaces of the 3D models during the rendering process. It is most often used in first-person shooters.

In light mapping, the lighting calculations for a scene are performed once and stored in a 2D texture, allowing the scene to be rendered more quickly and with less computational overhead. This is also more accurate than vertex lighting. Global effects like shadow can also be taken into account.

*What is environment mapping? Name some methods of generating environment maps?*

Textures are used to create reflections on objects. It works by using a pre-computed texture map (known as an environment map) that contains the surrounding environment of an object.

The environment map is generated by rendering the surrounding environment from a single viewpoint, typically the position of a reflective object in the scene, into a cube map, a dual paraboloid map or a spherical map.

*What is Bump Mapping?*

Bump mapping is technique that simulates the appearance of surface detail, such as bumps, roughness, and textures, on a 3D model without actually changing its geometry. This allows for the creation of more detailed and realistic 3D models with less computational overhead than would be required to model the additional geometry. It simulates rough surfaces by calculating per-pixel illumination. Per-pixel normals are a prerequisite for Bump Mapping. Since bumpmaps only change the texture and not the geometry, it leads to problems at the edges.

**Deferred Shading**

*What is Deferred Rendering?*

Deferred rendering is a technique that involves rendering scene information, such as the material properties and lighting information, into separate buffers or "g-buffers" before performing the final shading and rendering pass. This is in contrast to traditional forward rendering, which performs shading and lighting calculations for each pixel during the initial rendering pass.

The initial rendering pass writes information about the scene into a set of buffers, including a position buffer, a normal buffer, a material buffer, and a depth buffer. This information can then be used in subsequent passes to perform per-pixel shading and lighting calculations, allowing for more complex lighting and material models to be used.

*What’s a G-Buffer? What can be stored in a G-Buffer? Give 2 use cases.*

G-Buffers pass data from one stage of the deferred rendering pipeline to next. Color, Depth, Normal vectors, Position and Object identity can all be stored in a G-Buffer.

Examples for deferred rendering: Screen-space ambient occlusion, Non photo-realistic rendering, High-dynamic range rendering and Deferred shading

*Compare Deferred Shading to conventional Forward Shading? Applications? Pros/Cons?*

Conventional shading is done in fixed function pipeline or fragment shader. Deferred Shading is done at the end of the rendering pipeline as a post processing effect.

Examples for deferred rendering: Screen-space ambient occlusion, Non photo-realistic rendering, High-dynamic range rendering and Deferred shading

Pros:

* Render geometry only once
* Perform complex shading and post-processing per pixel
* Complexity O(Light sources + Objects) instead of O(Lights\*Objects)
* Independent of geometry and depth complexity
* Time for shading can be predicted well-> good for games

Cons:

* Requires more memory and frequent read/write operations
* Advanced effects (transparency, ghostings) require per-pixel sorting
* Cannot use hardware anti-aliasing
* Forward shading may be faster, if low number of light sources, low depth complexity and no need for post-processing effects

**Special Effects**

*What are SSAA and MSAA?*

SSAA (Super-Sample Anti-Aliasing) and MSAA (Multisample Anti-Aliasing) are techniques used in computer graphics to reduce the visual artifacts known as aliasing, which can occur when the edges of 3D models appear jagged or stair-stepped.

SSAA works by rendering the scene at a higher resolution than the final output, and then down-sampling the result to the final resolution. This effectively averages out the jagged edges and results in a smoother, more visually appealing image. It is the most accurate antialiasing method and helps with all forms of spatial aliasing. However, SSAA can be computationally expensive and may not be practical for use in real-time applications, such as video games, due to the high processing overhead required.

MSAA, on the other hand, works by taking multiple samples at each pixel, and then averaging them to produce the final color. This is done during the rasterization stage of the graphics pipeline and results in smoother, less jagged edges without the performance overhead of SSAA. MSAA is often used in video games and other real-time graphics applications as a more efficient alternative to SSAA.

*What is Temporal Anti-Aliasing? What is jittering? What is reprojection?*

TAA is an Anti- Aliasing technique which used multiple frames of the same scene and combines them together to get an anti-aliasing effect. The past frames are stored in a buffer.

Jittering is a technique that can be used to improve TAA. It jitters the projection matrix and generates a slight offset to improve the result.

Reprojection: The history for the current pixel may be at a different location, or might not exist at all. The geometry is transformed twice, using the current frame’s transformation matrices and the previous frame’s transformation matrices. The resulting offsets are stored into a motion vector texture. Using this texture, we can look up the previous location in the history buffer

*What are some methods for achieving Motion Blur?*

Fast moving objects appear blurry to the eye and camera and makes them look more real.

Discrete Methods:

Simplest method: Render the object at past positions with varying transparency. The object needs to be rendered multiple times.

Image Space Motion Blur: Render the object to buffer then copy the buffer with varying transparency. This is more efficient.

Continuous Motion Blur:

For each pixel compute how, the pixel moves over time. The current and previous model-view projection matrix form a velocity buffer. Sample the line along that direction. Blend the colour values along that line.

*What is a particle system? Give pseudocode for a generic particle system algorithm. Give three examples for use cases of particle systems.*

Particle systems model objects that change over time. Typically, many small objects are used. State-less vs. state-full particles are used. All particles of a system use the same update method and share the same properties.

Use cases: Rain, snow, clouds, Explosions, fireworks, smoke, fire, Sprays, waterfalls

**Semiglobal Illumination**

*How does a stencil buffer work? What are its applications?*

A stencil buffer allows to mask parts of the framebuffer. The applications are Reflections, CSG (constructive solid geometry). Shadow volumes

*What are Screen Space Reflections?*

Screen Space Reflections is a real-time rendering technique used in computer graphics to simulate the reflections of the environment on glossy or reflective surfaces. SSR works by tracing rays from the pixels on the screen into the scene to find the first visible reflection. The resulting reflection is then blended with the original image to create the final, more visually appealing result. SSR uses the depth buffer and normal buffer generated during the rendering process to approximate the reflections. This allows for real-time reflections in complex scenes, with the trade-off being that the reflections may not be as accurate as those generated with traditional ray tracing methods.

*How does Depth Peeling work? What is it used for? What is Dual Depth Peeling?*

It is used for interactive order independent transparency. Multiple passes are used for it. First pass: find the front-most depth/color and with each successive pass finds the depth/color for the next nearest fragment on a per-pixel basis. After that, compare the previous layer and current layer. In Dual Depth Peeling we Peel from the front and back in one pass.

*What is a per Pixel Linked List? How does it work?*

A per-pixel linked list is a data structure used to store information about the objects that contribute to each pixel in a scene. It works by creating a linked list for each pixel in the screen space, where each node in the list represents an object that contributes to the color of the pixel. In the rendering process the objects are sorted in back-to-front order, based on their depth. Then, for each pixel in the screen space, the linked list is traversed and the colors of the objects are combined to produce the final color of the pixel. The advantage of using a per-pixel linked list is that it can help to reduce the number of redundant shading calculations. With a per-pixel linked list, only the objects that contribute to the final color of a pixel are shaded, reducing the number of redundant calculations and improving performance.

*Explain the Shadow Mapping algorithm?*

Render a depth-map from the point of view of the light source. Use the depth-map when rendering the scene from the point of view of camera For each pixel: Compute the depth as seen from the light source, Compare to the depth-map, If the depth is larger then the pixel is in a shadow. If the distance is approximately equal the pixel is in light

Explain the Shadow Volume Stencil algorithm?

Intersect the view rays with the shadow volume and count the number of intersections, until receiver is hit. If the ray enters a new volume add a one. If it leaves a volume subtract a one. If the number is greater than zero when the receiver is hit, then the receiver is in a shadow.

Ambient occlusion? Pros/Cons?

Ambient occlusion is shadowing by ambient light. It is independent of light and object orientation. It darkens surfaces which are partially visible in the environment and adds depth and contrast but raycasting from every point is expensive.

**Global Illumination**

*What are virtual point lights? Describe an algorithm using virtual point lights for computating illumination? Applications?*

Are secondary light sources which are created when rays from the primary light source are shot through the scene. Each point hit is then defined as a VLP and a shadow map from this new light source is created. It is also possible that the ray bounces multiple times, creating more light sources.

Algorithm:

Pass 1: VPL generation

* Shoot photons from light source
* Follow them through scene
* At each hit point: create VPL
* Russian roulette to end path
* One shadow map for each VPL

Pass 2: Deferred rendering

* Render considering all the VPL and their shadow maps

*What is precomputed radiance transfer? What are its applications?*

It precomputes the light transport for a 3d model before the rendering. It is used for environment lighting, shadows and indirect illumination.

What are Spehrical harmonics?

They provide a compact and efficient way to describe the distribution of light and color in a scene and can be used to approximate a wide range of lighting effects, including ambient lighting, diffuse shading, and specular reflections. Spherical harmonics are defined over a sphere and are based on the spherical coordinate system. They are a combination of Legendre polynomials and trigonometric functions, and can be thought of as a series of terms, each with a unique direction and intensity. A set of coefficients can be computed for each light source in the scene and the environment map, which define the intensity and direction of light in the scene. These coefficients can then be used to calculate the final shading of an object, taking into account the contributions of all the light sources in the scene.

**GPU Raytracing**

*What is the DirectX Ray Tracing Pipeline?*

* The pipeline is split into five new shaders:
* The ray generation shader defines how to start ray tracing.
* The intersection shader defines how rays intersect geometry.
* The miss shader defines behavior when rays miss geometry.
* The closest-hit shader runs once per ray (e.g., to shade final hit)
* The any-hit shader runs once per hit (e.g., to determine transparency)

**Levels of Detail**

*What is Level of Detail Rendering? How does Predictive or Reactive Level of Detail Selection work? What types of LOD Switching are there?*

Render objects with less detail if they are at a great distance to the viewer.

Reactive LOD: Multiply the object size with a factor c. If the frame rate is too low decrease c. If the frame rate is too high increase c. This results in a roughly constant frame rate. Problems can occur, if complex objects suddenly become visible. This Requires hysteresis.

Predictive LOD: COST is the time for drawing object with a given LOD. The Goalis to get the best possible image quality. BENEFIT is the contribution of object to image quality. The Most important is the screen-size of object • Optimization problem – Sum(BENEFITS) ◊ max, BUT – Sum(COSTS) ≤ FRAMETIME

Hard Switching, Blending, Geomorphing

*Name a method of geometry simplification?*

Edge collapse, Vertex-pair collapse, Triangle collapse, Cell collapse and Vertex removal

*What is the quadric error metric used for?*

Quadric Error Metrics are a measurement of error that determines how far a vertex is from an ideal spot. It is used for surface simplification.

**Visibility**

Occlusion culling? Methods? Difference to Frustrum Culling and Backface Culling?

Occlusion culling increases rendering performance simply by not rendering geometry that is outside the view frustum or hidden by objects closer to the camera.

For Occlusion culling use two spatial data structures

* Scene data structure (SDS): Stores the objects in the scene
* Shadow volume data structure (SVDS): Generated by occluders (selected from scene), or generated by virtual occluders (synthesized)
* Cull SDS using SVDS

Frustum culling is a basic technique that every serious 3d engine is doing. In its simplest form, all objects of the scene are tested for intersection with the view frustum pyramid. This test can be conservative which means that it is acceptable to have some objects being reported visible even if they are not. Objects outside the pyramid do not get rendered.

Backface Culling is used to hide the backface of objects.

*What is the Bounding Volume Hierarchy?*

The Bounding Volume Hierarchy (BVH) is a data structure used for efficient acceleration of ray tracing and other geometric queries. The BVH is a tree structure, where each node represents a bounding volume that encloses a set of objects in the scene. It is constructed by recursively dividing the scene into smaller and smaller sub-volumes, creating a hierarchy of bounding volumes that encloses the objects in the scene. The leaves of the tree represent individual objects in the scene, while the internal nodes represent the bounding volumes that enclose a set of objects. When performing a ray tracing calculation or other geometric query, the BVH can be used to quickly find the objects that are intersected by the query ray. By traversing the BVH, the number of objects that need to be tested for intersection can be reduced, improving the performance of the query.

*What is a Potentially Visible Set?*

A Potentially Visible Set (PVS) is a data structure used to keep track of the objects in a scene that are potentially visible from a given viewpoint. The PVS is used in real-time rendering to determine which objects need to be drawn from a given viewpoint, and which objects can be skipped or culled. The PVS is constructed by dividing the scene into a set of view-independent visibility regions. For each region, the objects that are potentially visible from the viewpoint are determined, and a data structure is created that represents the set of objects that are potentially visible. When rendering a scene, the PVS can be used to quickly determine which objects need to be drawn and which objects can be skipped. By using the PVS, the number of objects that need to be processed and rendered can be reduced, improving performance and reducing the amount of processing required.

*What is the Cells and Portals Technique?*

The Cells and Portals technique is a method used for fast and efficient visibility determination in large, complex environments, such as in virtual reality or first-person shooter games. The technique works by dividing the environment into a set of interconnected cells, with each cell representing a distinct, enclosed space. The boundaries between cells are represented by portals, which are objects that connect one cell to another and allow visibility between cells. To determine the visibility of objects in the environment, the cells and portals technique uses a recursive process that starts from the player's current cell and visits each cell that is potentially visible from the current cell. As the process moves from one cell to another, it updates a list of visible objects, and keeps track of the objects that are potentially visible but have not yet been processed.