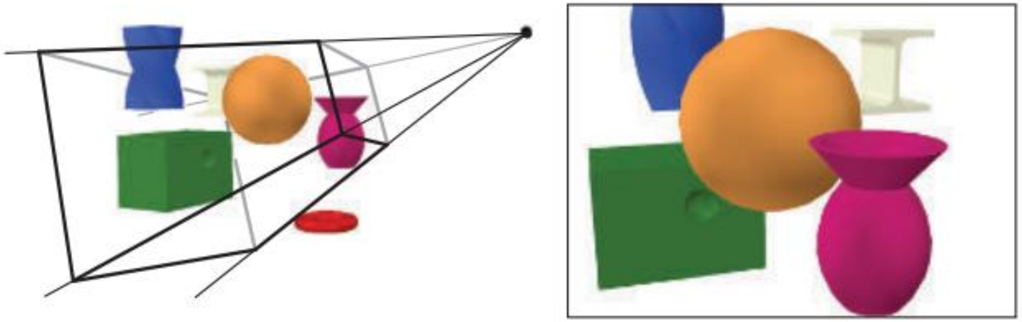
Rendering Pipeline in Computer Graphics

Sung Soo Hwang

Introduction

- Things that affect the locations and shapes of the objects
  - Geometry
  - Characteristics of the environment
  - Placement of the camera

- Things that affect the appearance of the objects
  - Material properties
  - Light sources
  - Textures
  - Shading equations

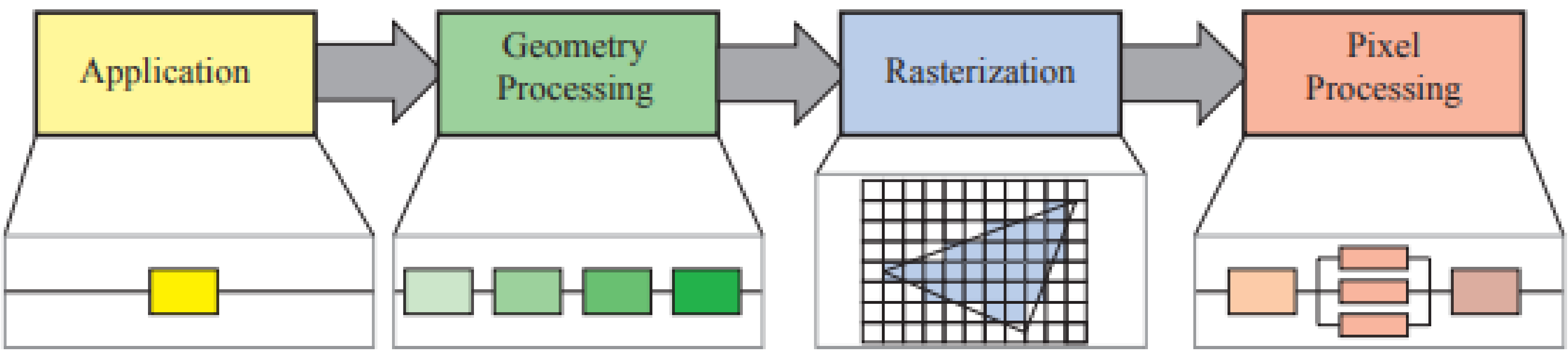


Rendering results through the graphics rendering pipeline

Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire,
「Real-Time Rendering, 4th Edition」, CRC Press

Architecture

- The graphics rendering pipeline (pipeline) is a key component of real-time graphics.
- The pipeline consists of multiple stages, each running in parallel based on the results of the previous stage. And it delays until the slowest step finishes its work.
- Several notations that express rendering speed
  - Frames Per Second(FPS), Hertz(Hz), Milliseconds(ms)



The basic construction of the rendering pipeline

Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire, 「Real-Time Rendering, 4th Edition」, CRC Press

Application Stage

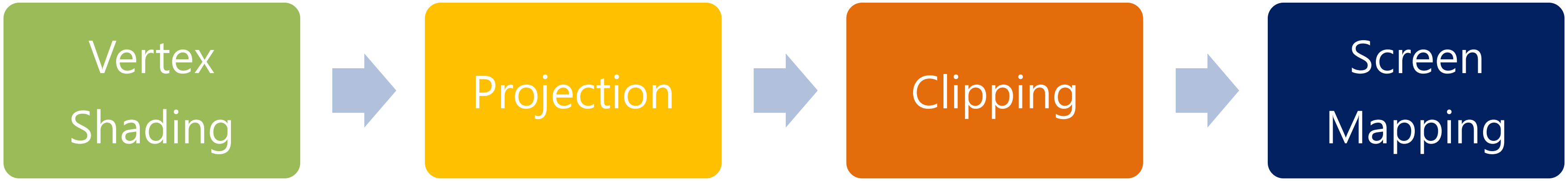
- The application phase is one that the programmer has full control.
- As the application develops, many changes can be made to its implementation to improve the performance.
- The output of the application phase is the default rendering.
  - Points
  - Lines
  - Triangles

Important jobs of the application stage

- Reading input
- Managing non-graphical output
- Texture animation
- Animation via transforms
- Collision detection
- Updating the state of the world in general

Geometry Processing

- The output of the Application Stage is polygons
- The Geometry Processing Stage processes these polygons using the following pipeline:



Substages of the Geometry Processing

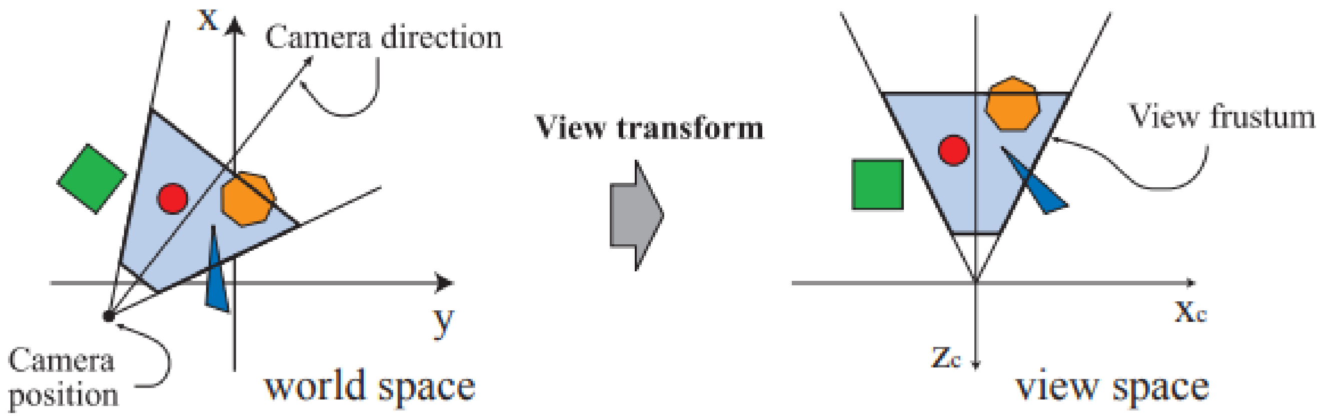
Vertex Shading

- Each 3D model has a coordinate system called model space.
- When all models in a scene are joined together, the models must be transformed from model space to world space.
- After that, you still need to consider the camera position.

Vertex Shading

- We transform the model into camera space or, more generally, into view space or eye space using a view transform.

- Then, the camera will sit at (0, 0, 0), looking into negative z



Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire,
「Real-Time Rendering, 4th Edition」, CRC Press

Vertex Shading

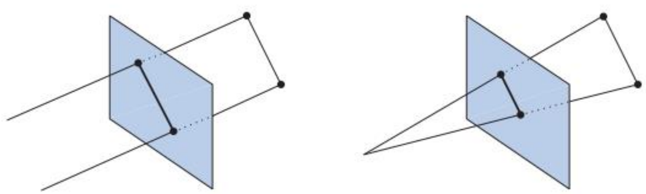
- Understanding the effect of light on the material is called shading.

- This involves calculating the shading equation at different points on the object.

- A variety of material data can be stored at each vertex:
  - Location
  - Normal
  - Color

Projection

- A projection transforms the view volume into a standardized unit cube.
- Then the vertices have 2D positions and z-values.
- There are two general forms of projection.
  - Orthographic: Parallel lines remain parallel and objects are not far apart
  - Perspective: The further away an object is, the smaller it appears



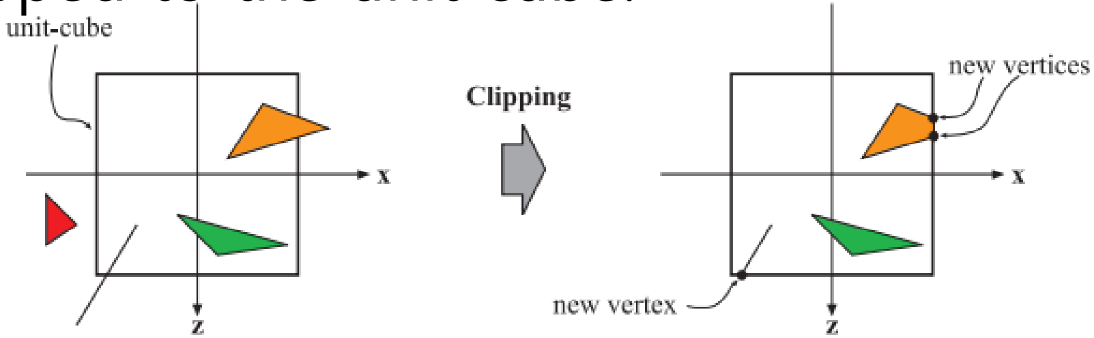
Orthographic

Perspective

Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire, 「Real-Time Rendering, 4th Edition」, CRC Press

Clipping

- Clipping handles polygons based on their position relative to the view volume.
- Cases of clipping:
  - Polygons inside the view volume do not change
  - Polygons that are completely outside the view volume are ignored (not rendered)
  - Polygons partially inside are clipped
    -> new vertices are created at the boundaries of the volume
- The benefit of doing the view transformation and projection before clipping is that it ensures that the base cube is always clipped to the unit cube.

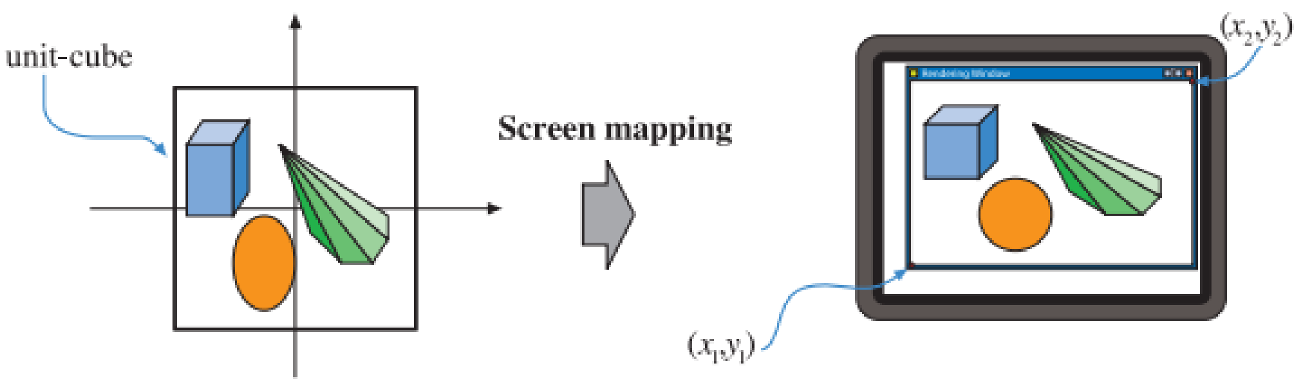


Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire, 「Real-Time Rendering, 4th Edition」, CRC Press

Screen mapping

- Screen mapping converts the x and y coordinates of each polygon from the unit cube to screen coordinates

- All APIs have pixel position values that increase from left to right, but positions of zero are not the same
  - Direct X defines the upper left of the screen as (0, 0)
  - OpenGL defines the lower left of the screen as (0, 0)



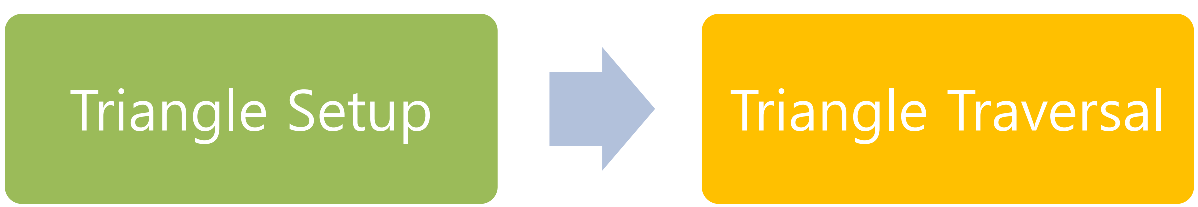
Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire, 「Real-Time Rendering, 4th Edition」, CRC Press

Rasterization

- The goal of the Rasterization stage is to take all the transformed geometric data and set a color for every pixel in the screen space

- Rasterization is also called:
  - Scan conversion
  - Synchronization

- The word Pixel is short for picture elements



Substages of Rasterization

Triangle Setup and Traversal

- Triangle Setup
  - Data is calculated for each triangle
  - This may include normal

- Triangle Traversal
  - Each pixel whose center is overlapped by a triangle must have a slice created for the portion of the triangle that overlaps the pixel
  - The properties of this fragment are created by interpolating the data from the three triangle vertices

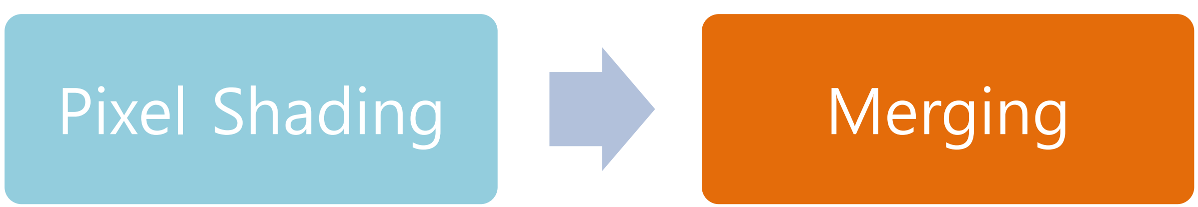
- This is done with fixed-acting (non-customizable) hardware

Pixel Processing - Pixel Shading

- Per-pixel shading (color) calculations are performed here using interpolated shading data as input

- This step is programmable, allowing you to apply different shading effects

- The most important effect is texturing or texture mapping



Substages of Pixel Processing

Pixel Shading - Texturing

- Texturing is gluing a (usually) 2D image onto a polygon
- To do this, we map texture coordinates to polygon coordinates
- Pixels in a texture are called texels
- This is fully supported in hardware(GPU)
- In some cases, you can apply multiple textures

Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michal Iwanicki, Sébastien Hillaire,
「Real-Time Rendering, 4th Edition」, CRC Press

Merging

- The final screen data containing the color of each pixel is stored in a color buffer
- The merge step is responsible for merging the color of each piece in the pixel shading step into the final color of the pixel
- Closely related to merging is visibility
- The final color of the pixel should be the color corresponding to the visible polygon (not the color behind it)
- Z-buffers are often used for this.