Modern OpenGL/Vulkan pipeline

University of California Riverside
Evolution of OpenGL

- 1992: Initially fixed functionality pipeline
- 2004: Added programmable shaders
- 2008: Fixed pipeline deprecated
- 2009: Fixed paths removed
  - Still available for compatibility
  - Fixed pipe emulated with shaders
Pipeline

- Input: geometry
- Output: image (on screen)
- Programmable stages

Diagram:
- vertex input
- vertex shader
- tessellation
- geometry shader
- post vertex processing
- primitive assembly
- rasterization
- fragment shader
- per-sample operations
**Vertex input**

- Supply input data to pipeline
- Stream of vertices
- Indices (for meshes)
Vertex shader

- Programmable (user-defined)
- For per-vertex operations
- Used to transform vertices
- Can do other things here
  - Eg, per-vertex lighting
  - Define colors at vertices
  - Interpolate within triangles

Diagram:

1. vertex input
2. vertex shader
3. tessellation
4. geometry shader
5. post vertex processing
6. primitive assembly
7. rasterization
8. fragment shader
9. per-sample operations
Tessellation

- Programmable (user-defined)
- Optional stage
- For subdividing primitives

Diagram:

- vertex input
- vertex shader
- tessellation
- geometry shader
- post vertex processing
- primitive assembly
- rasterization
- fragment shader
- per-sample operations

Image source: [?]
Geometry shader

- Programmable (user-defined)
- Optional stage
- Input: one primitive (at a time)
- Output: (many) primitives
- Possible uses:
  - Instancing
  - Turn lines into curves
  - Draw points as squares, diamonds, or stars (plots!)
  - **Bad use:** tessellation

Diagram:
- Vertex input
- Vertex shader
- Tessellation
- Geometry shader
- Post vertex processing
- Primitive assembly
- Rasterization
- Fragment shader
- Per-sample operations
Post vertex processing

- **Clipping**
  - removes (part of) primitive
  - if outside image
  - if too close/far

- **Perspective divide**
  - \((x, y, z, w) \rightarrow (\frac{x}{w}, \frac{y}{w}, \frac{z}{w})\)
  - We will see this later
Primitive assembly

- Turn primitives into base primitives
  - Triangle strip to triangles
  - Line loop to segments
- Back-face culling
  - do not render the backside
  - cannot see it anyway
Rasterization

- Input: primitive (eg, triangles)
- Output: pixels

Diagram:
- Vertex input
- Vertex shader
- Tessellation
- Geometry shader
- Post vertex processing
- Primitive assembly
- Rasterization
- Fragment shader
- Per-sample operations
Fragment shader

- Programmable (user-defined)
- Input: pixel data
  - interpolated vertex data
- Output: depth, color
- Compute color of pixel
  - Phong shading
  - texture mapping
  - bump mapping

![Shader pipeline diagram](image-url)
Per-sample operations

- Z-buffering (occlusion)
  - Discard hidden pixels
  - Optimization: before fragment shader if possible
- Masking, blending
- Storing results