Graphics Pipeline

Rendering approaches

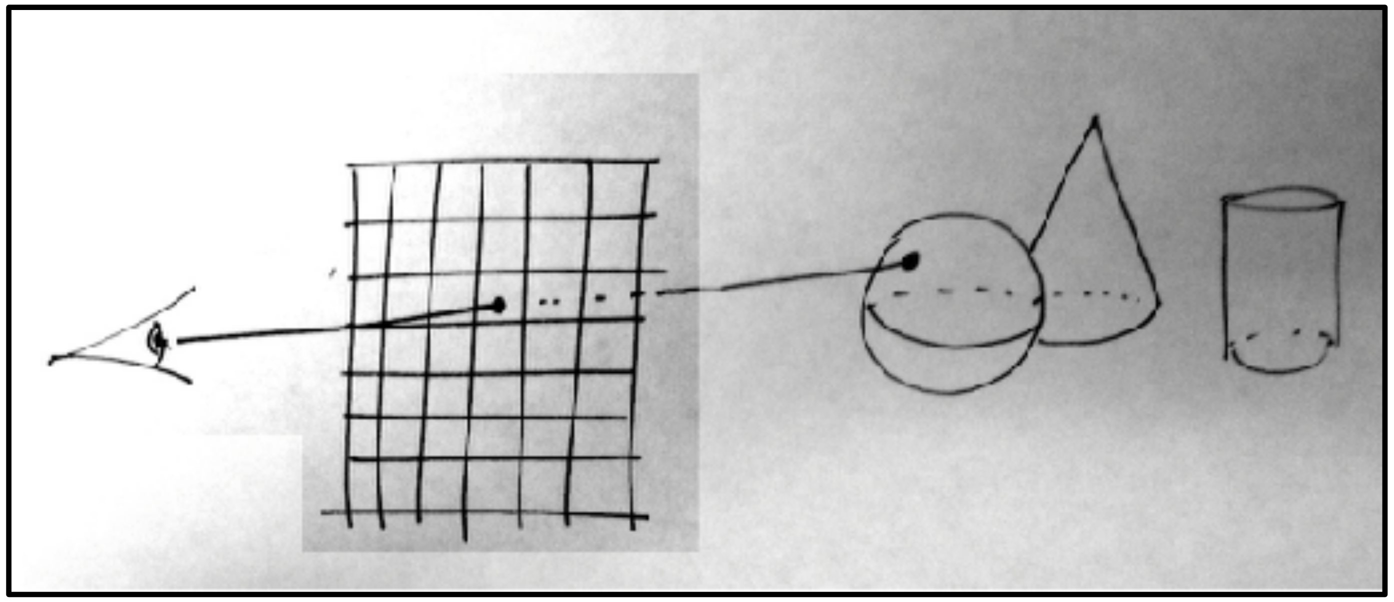
1. **object-oriented**

foreach object ...

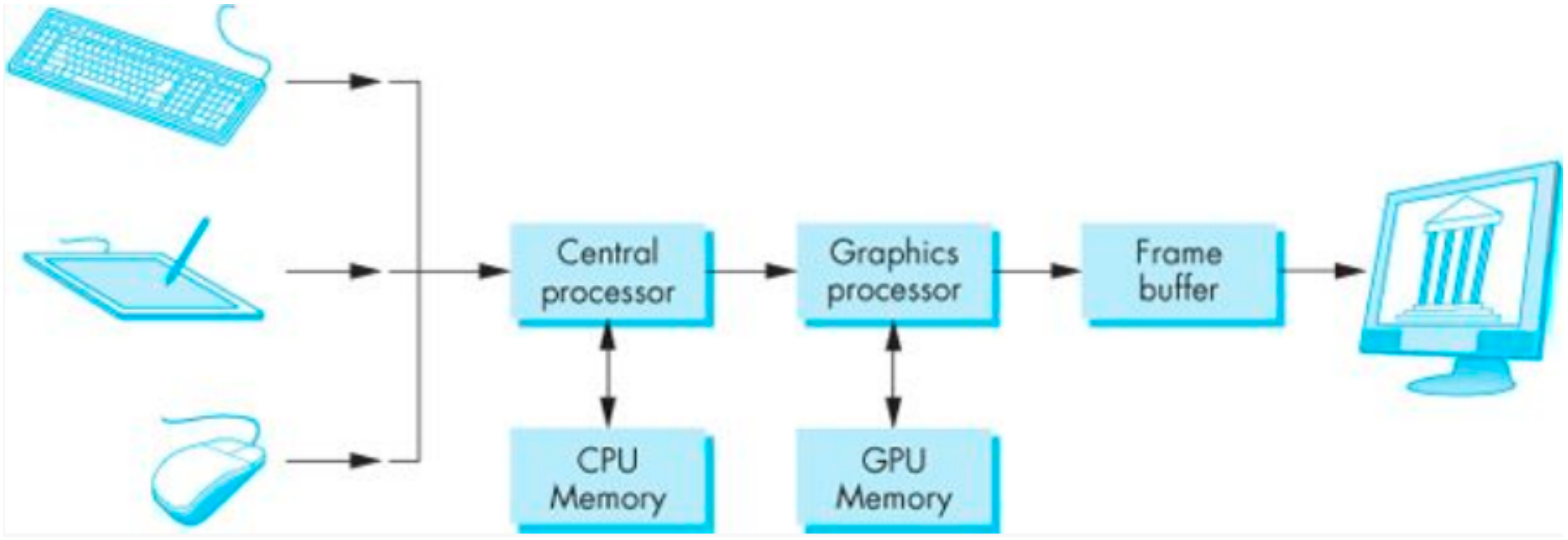
$$\begin{tikzcd}\text{vertices} \arrow[r] & \boxed{\text{3D rendering pipeline}} \arrow[r] & \text{image}\end{tikzcd}$$

2. **image-oriented**

foreach pixel ...



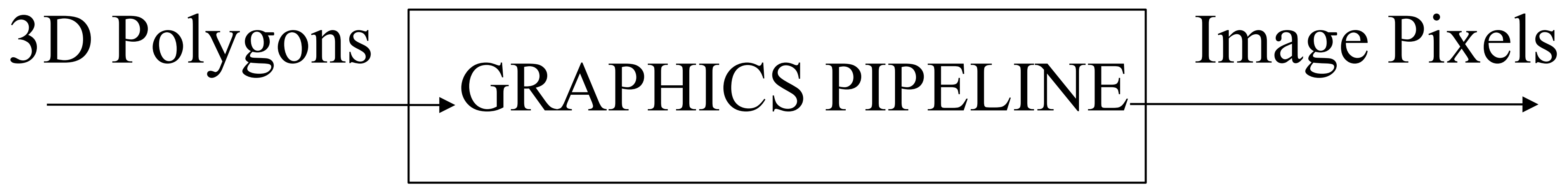
Modern graphics system



[Angel and Shreiner]

Z-buffer Rendering

- Z-buffering is very common approach, also often accelerated with hardware
- OpenGL is based on this approach



Choice of primitives

- Which primitives should an API contain?
  - small set - supported by hardware, *or*
  - lots of primitives - convenient for user

Choice of primitives

- Which primitives should an API contain?

➡ **small set - supported by hardware**

  - lots of primitives - convenient for user

Choice of primitives

- Which primitives should an API contain?

➡ **small set - supported by hardware**

  - lots of primitives - convenient for user

Performance is in **10s millions polygons/sec**
portability, hardware support key

Choice of primitives

- Which primitives should an API contain?

➡ **small set - supported by hardware**

- lots of primitives - convenient for user

GPUs are optimized for
points, **lines**, and **triangles**

Choice of primitives

- Which primitives should an API contain?

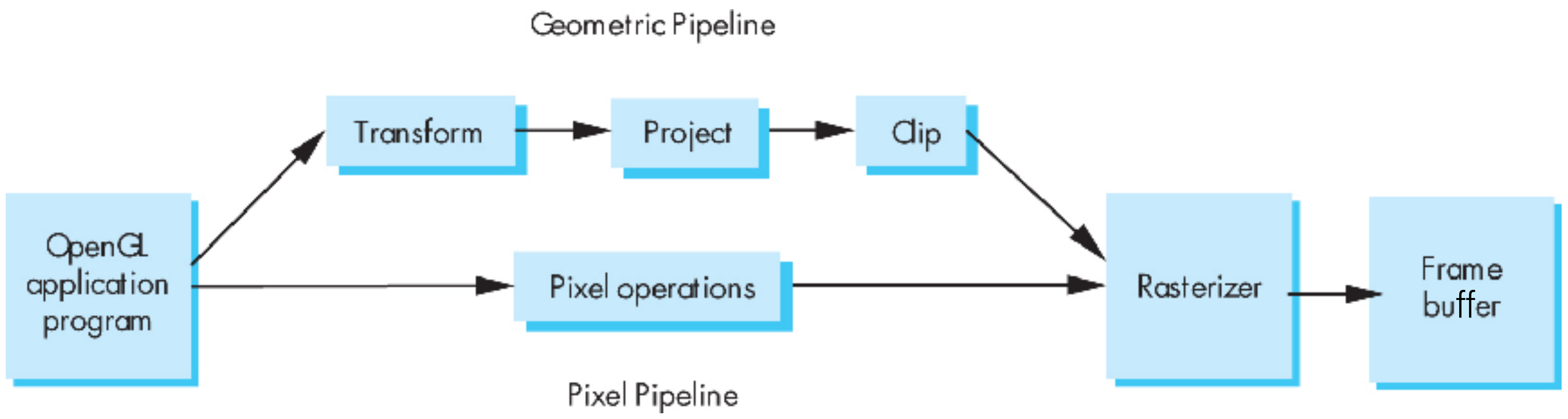
➡ **small set - supported by hardware**

- lots of primitives - convenient for user

GPUs are optimized for
points, **lines**, and **triangles**

Other geometric shapes will be built out of these

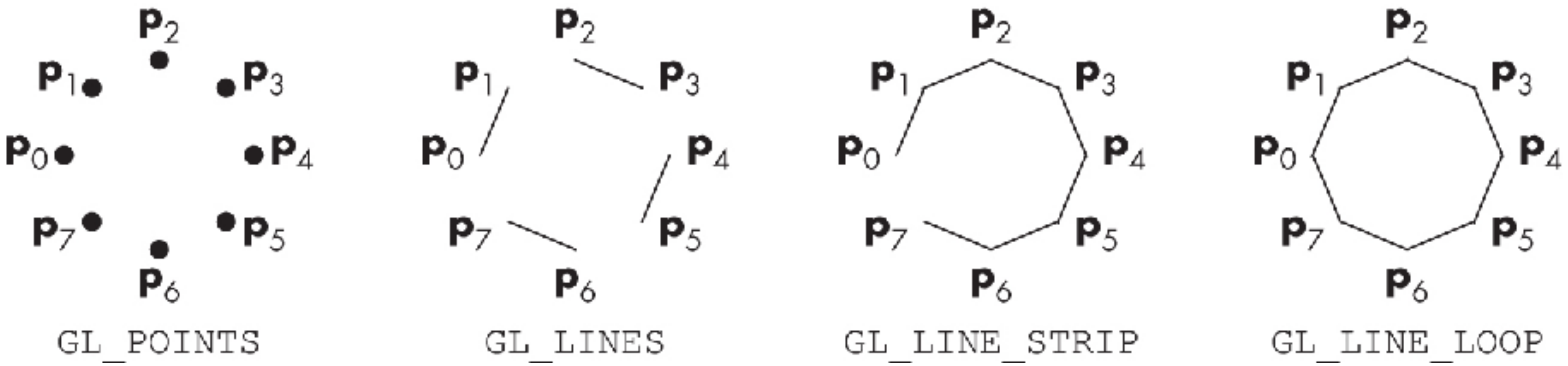
Two classes of primitives



[Angel and Shreiner]

Geometric : points, lines, polygons
Image : arrays of pixels

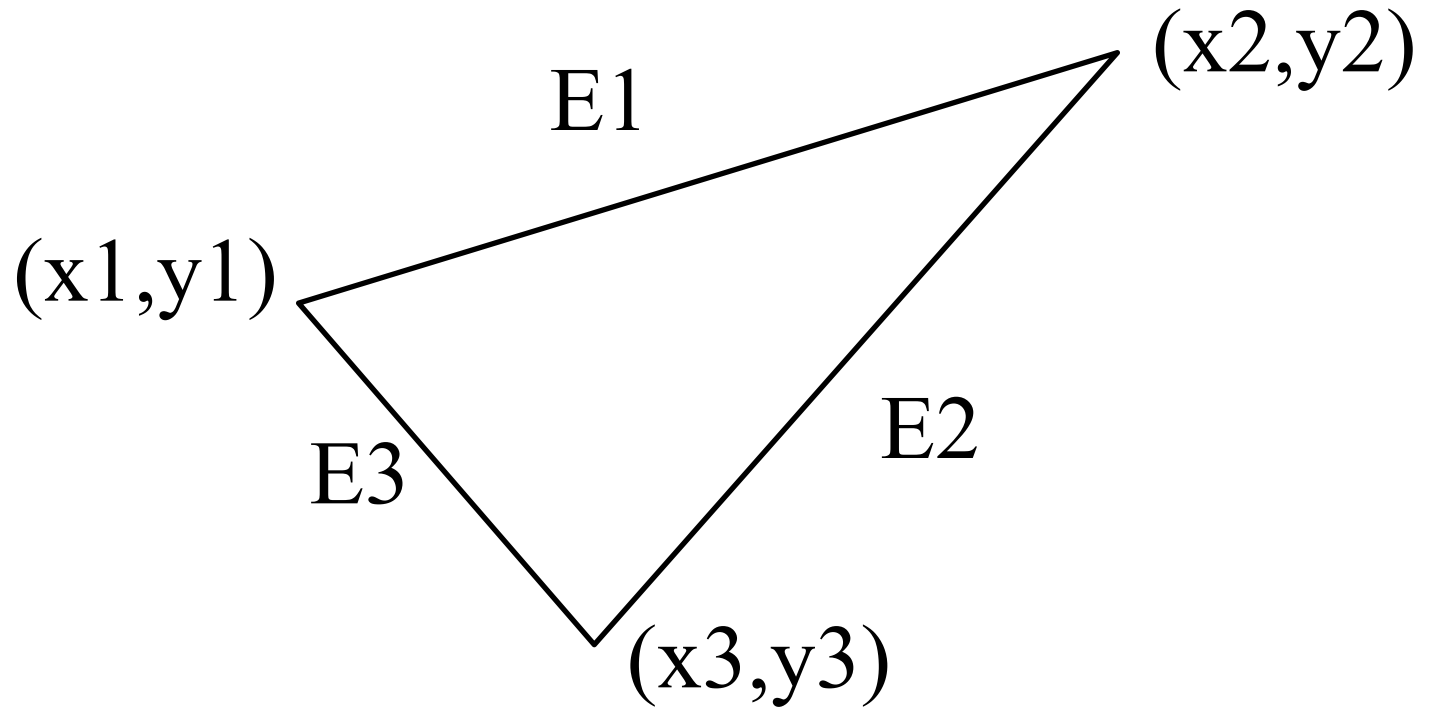
Point and line segment types



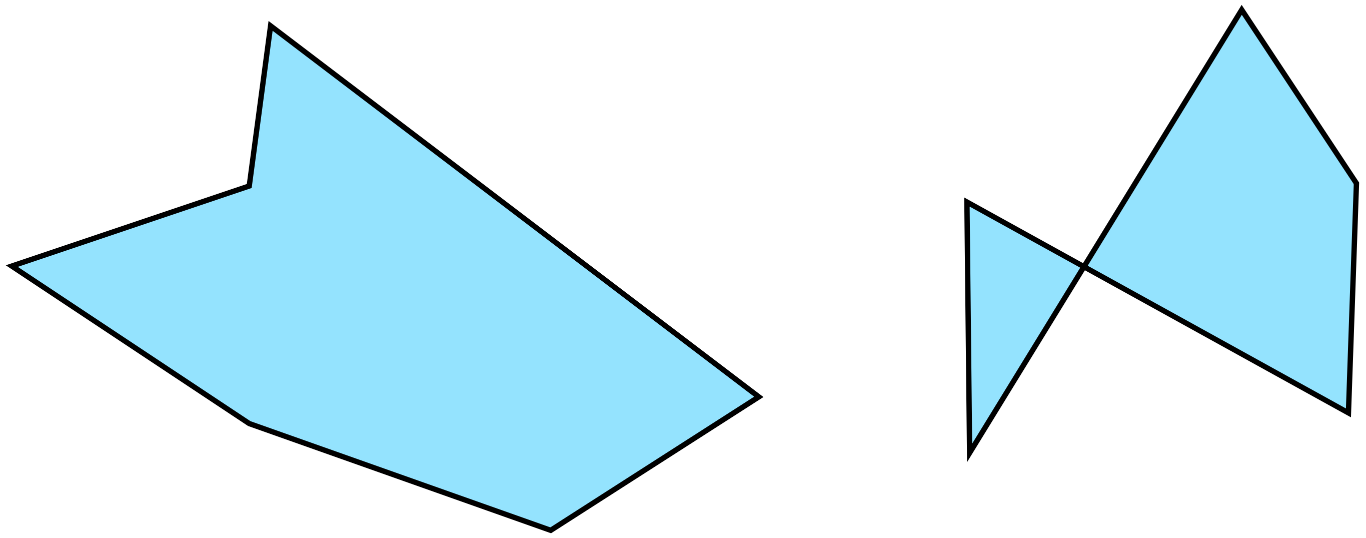
[Angel and Shreiner]

Polygons

- Multi-sided planar element composed of edges and vertices.
- Vertices (singular: vertex) are represented by points
- Edges connect vertices as line segments

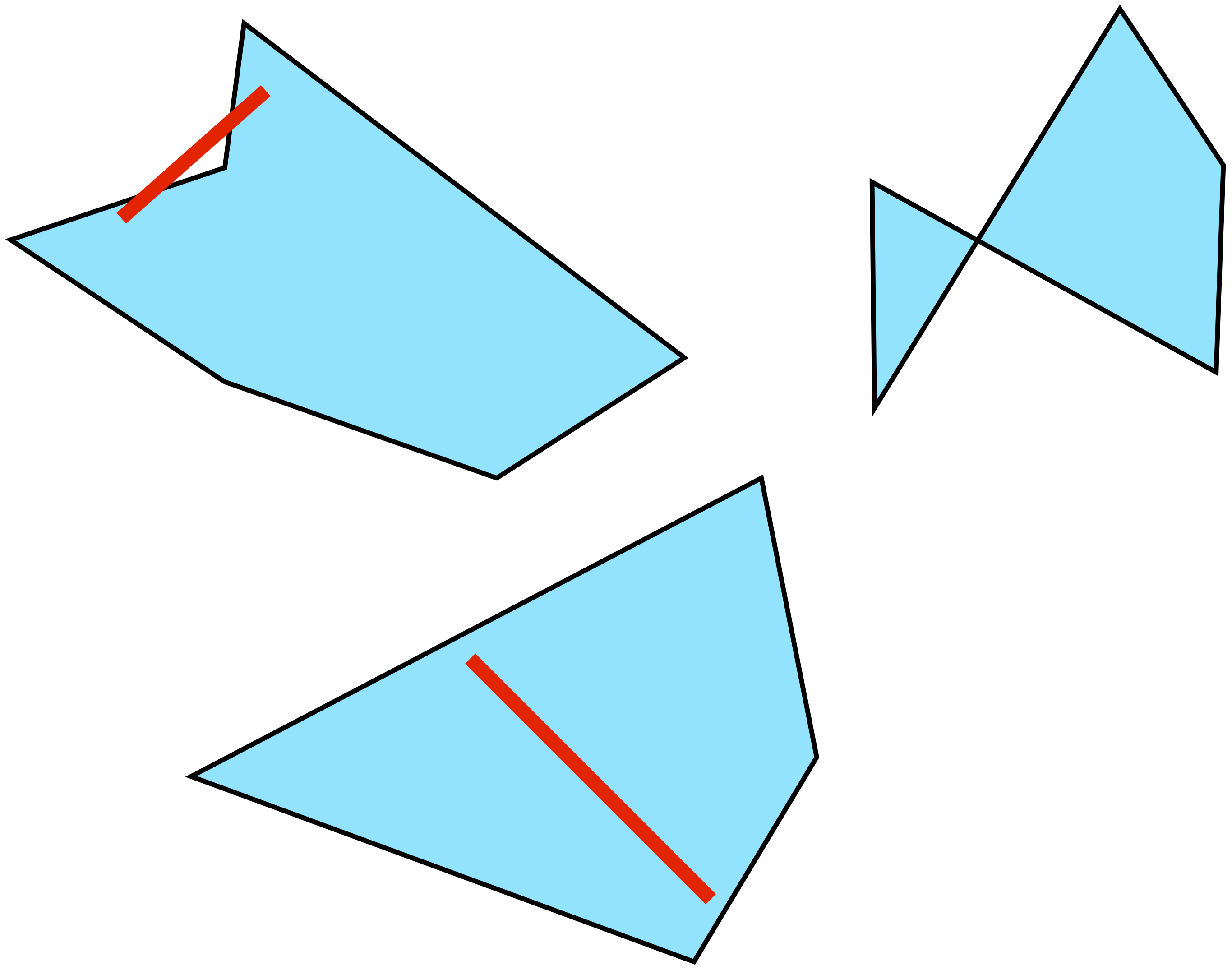


Valid polygons



- Simple
- Convex
- Flat

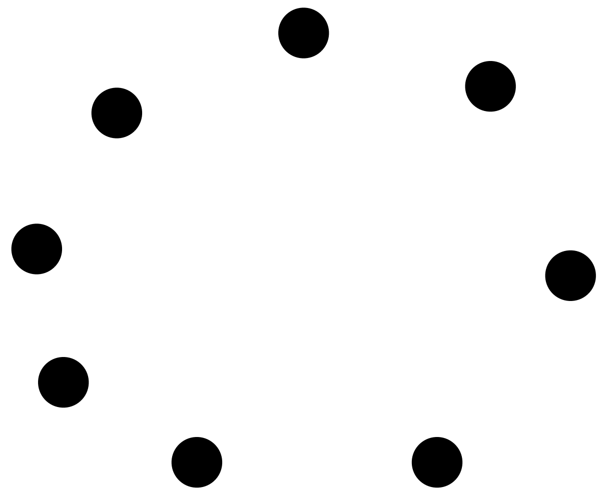
Valid polygons

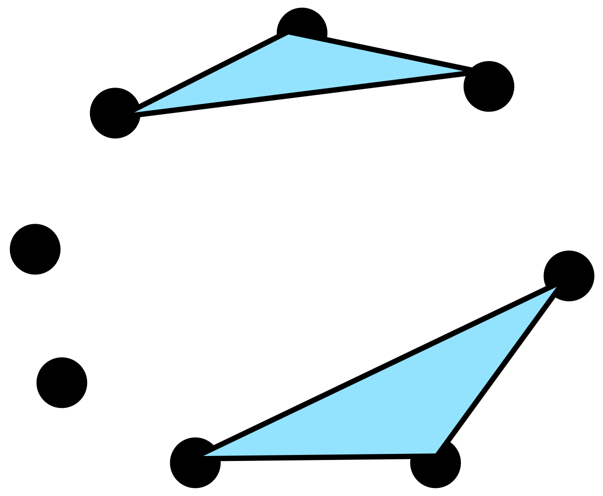


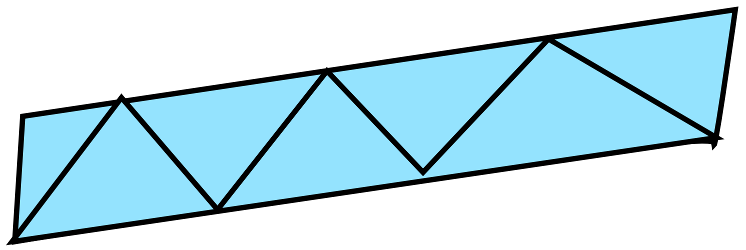
- Simple
- Convex
- Flat

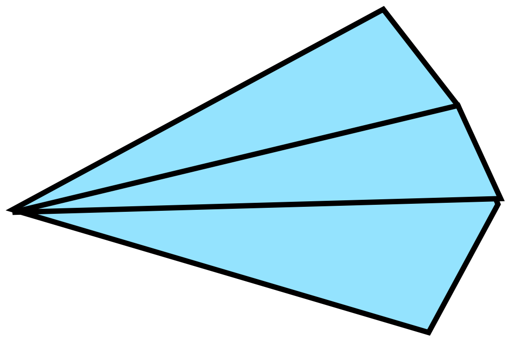
OpenGL polygons

- Only triangles are supported (in latest versions)

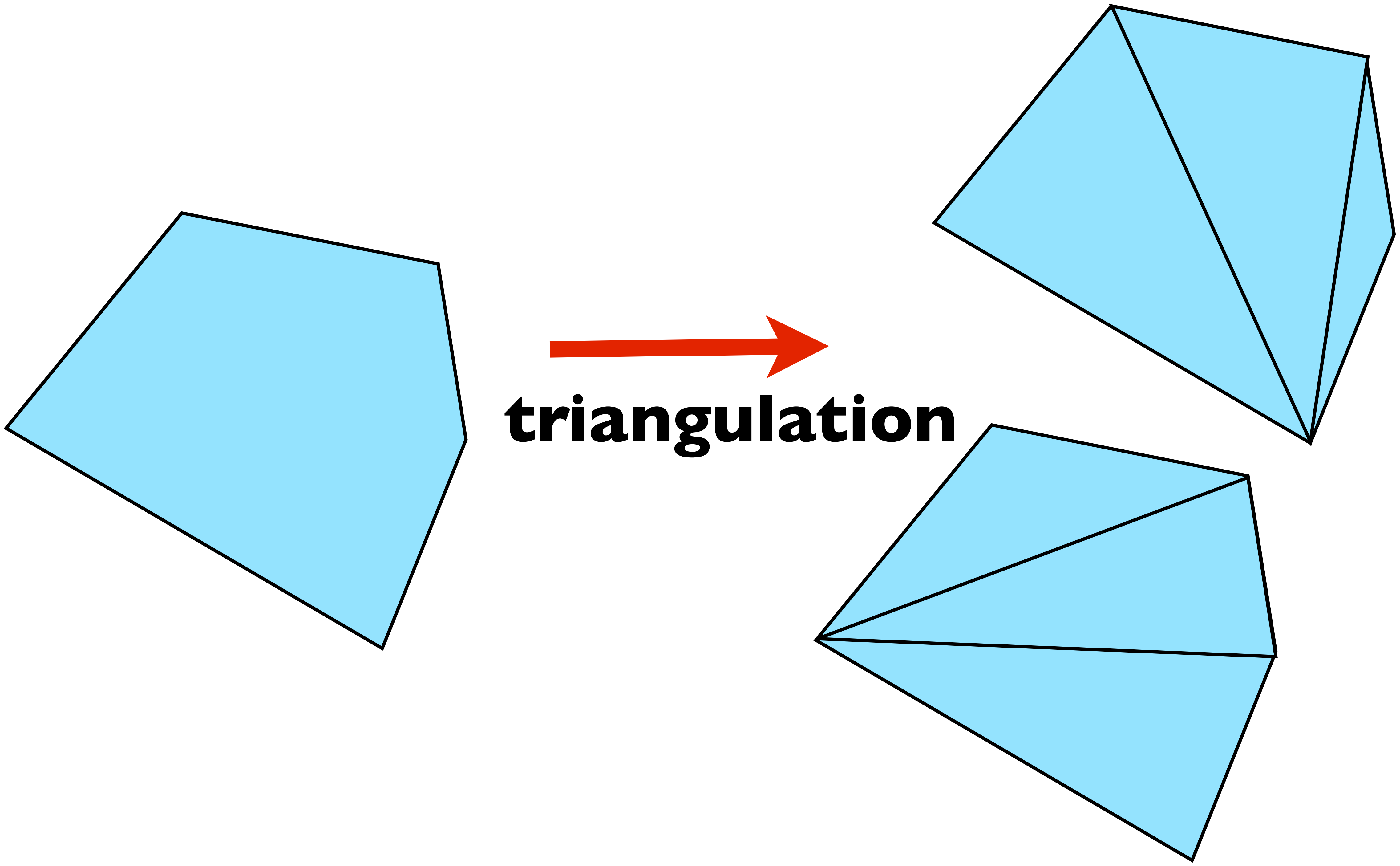

GL_POINTS


GL_TRIANGLES

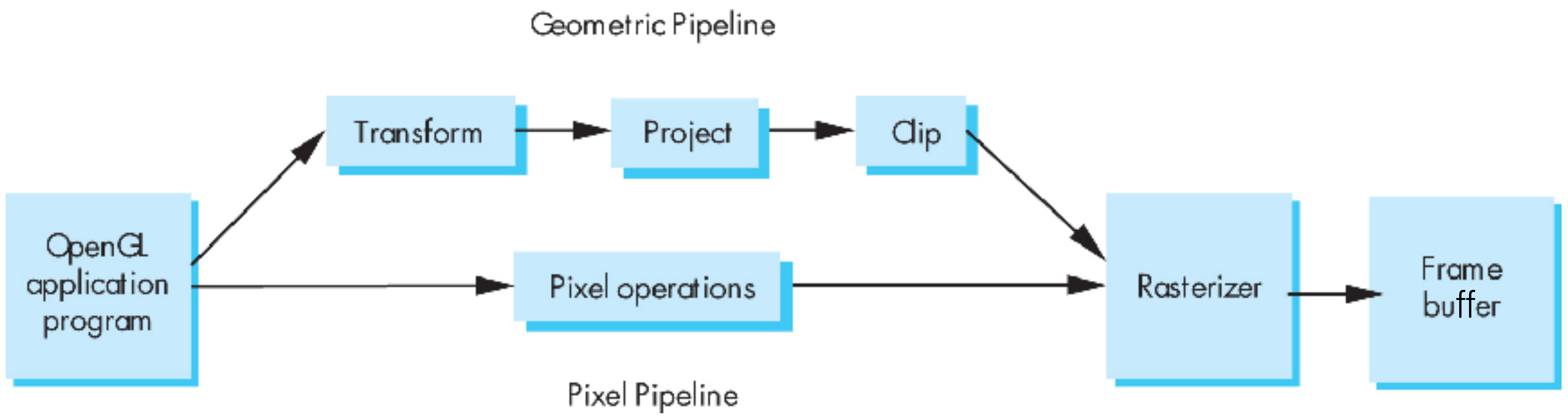

GL_TRIANGLE_STRIP


GL_TRIANGLE_FAN

Other polygons



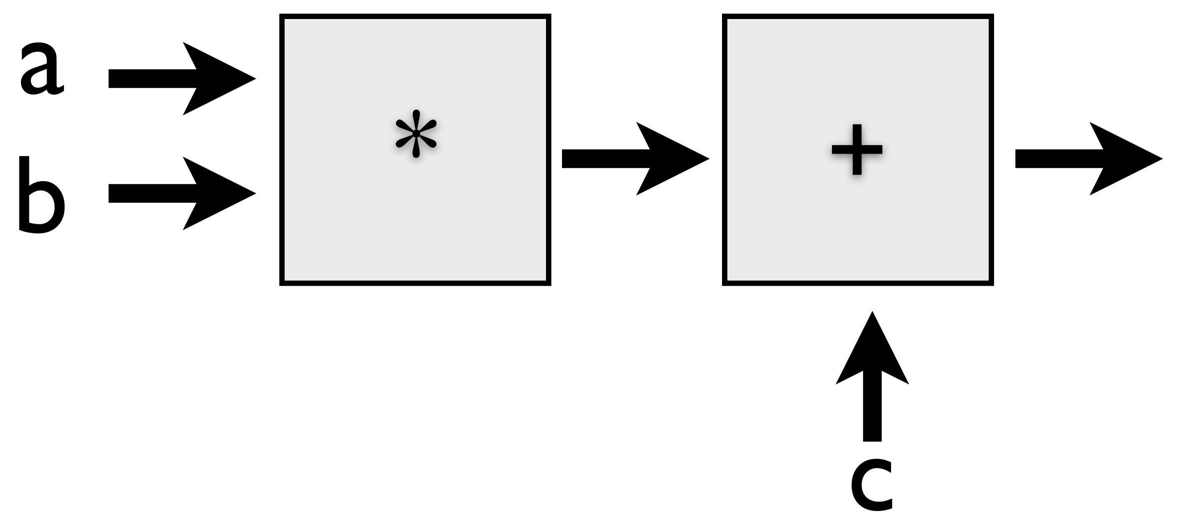
Graphics Pipeline

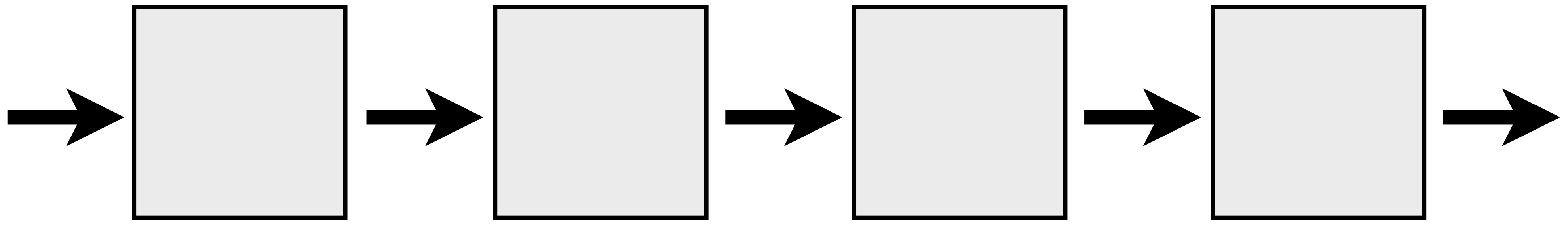


[Angel and Shreiner]

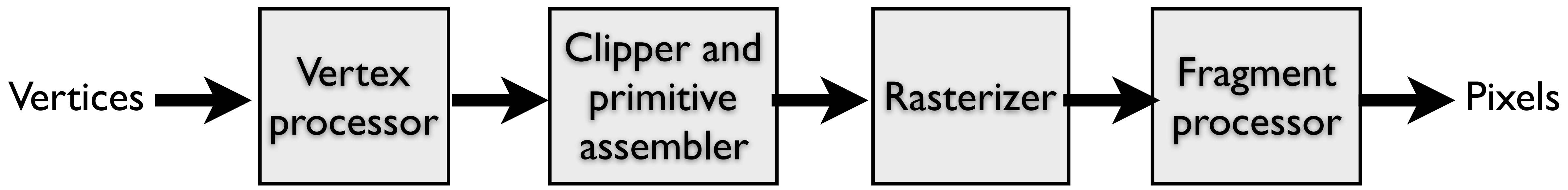
Pipelining operations

An arithmetic pipeline that computes c+(a*b)





3D graphics pipeline



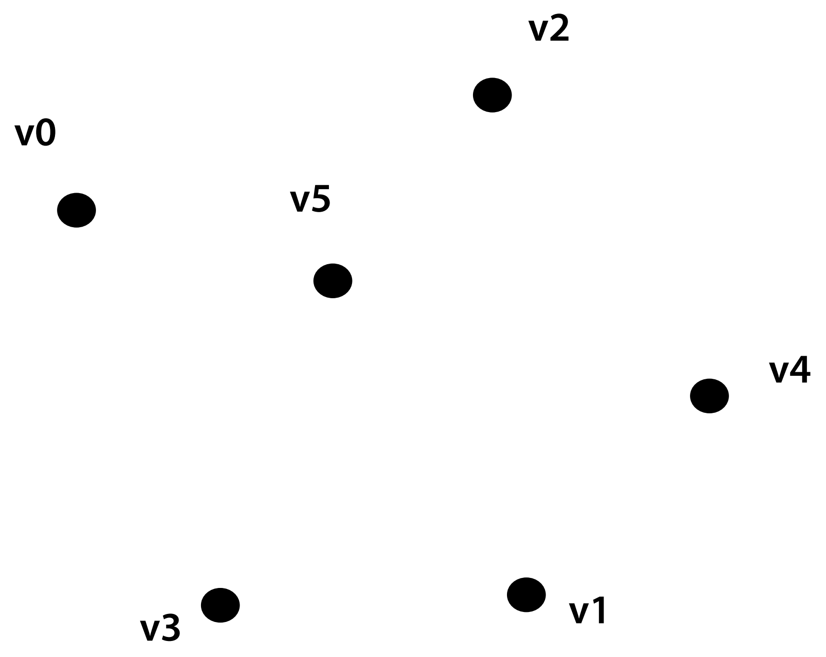
Geometry: primitives – made of vertices
Vertex processing: coordinate transformations and color
Clipping and primitive assembly: output is a set of primitives
Rasterization: output is a set of fragments for each primitive
Fragment processing: update pixels in the frame buffer

Graphics Pipeline

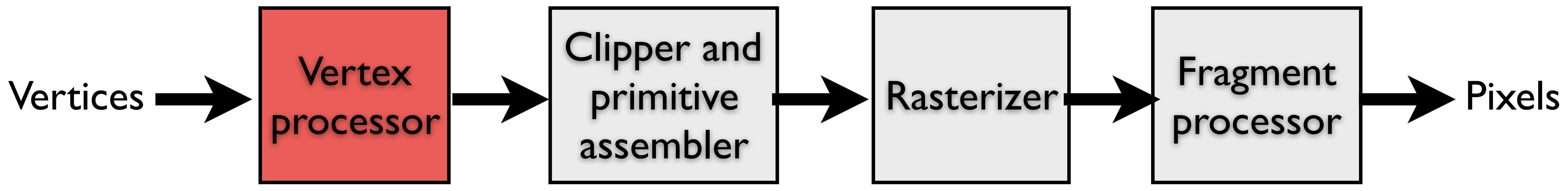
(slides courtesy K. Fatahalian)

Vertex processing

Vertices are transformed into "screen space"

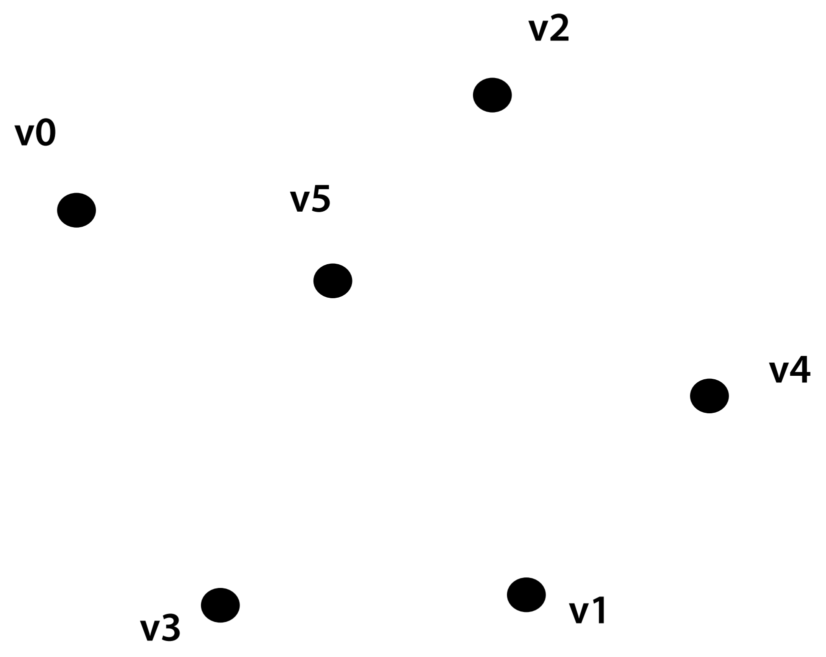


Vertices



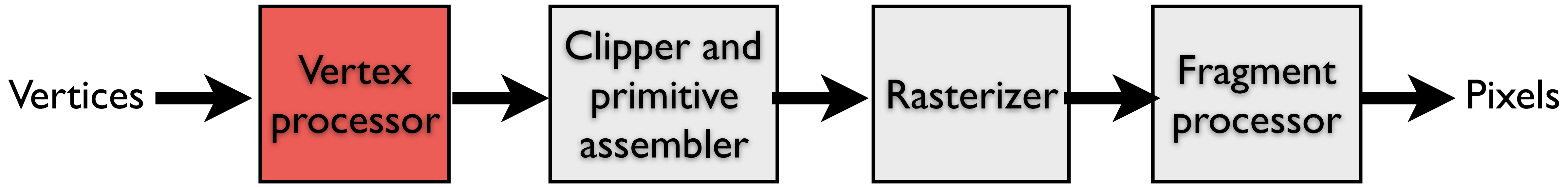
Vertex processing

Vertices are transformed into "screen space"



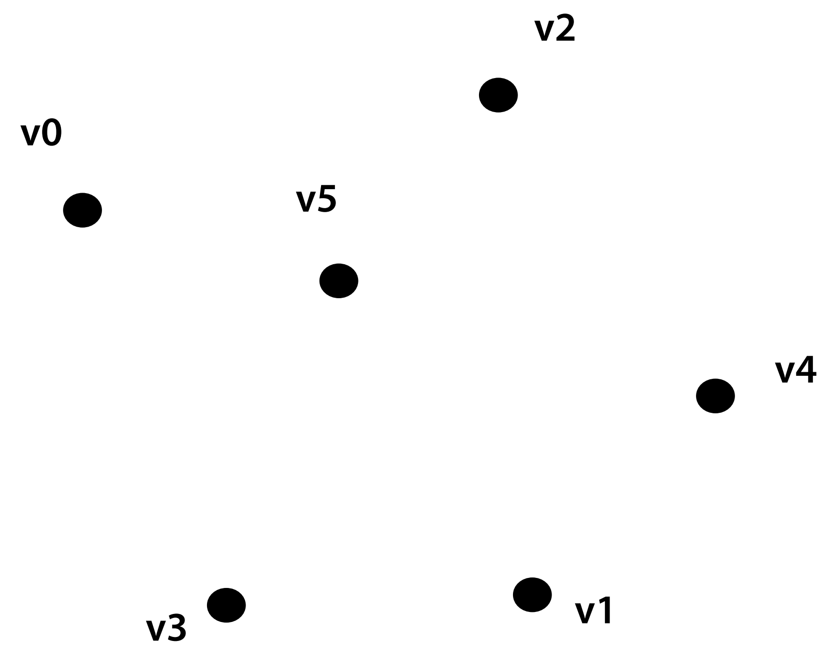
EACH VERTEX IS TRANSFORMED INDEPENDENTLY

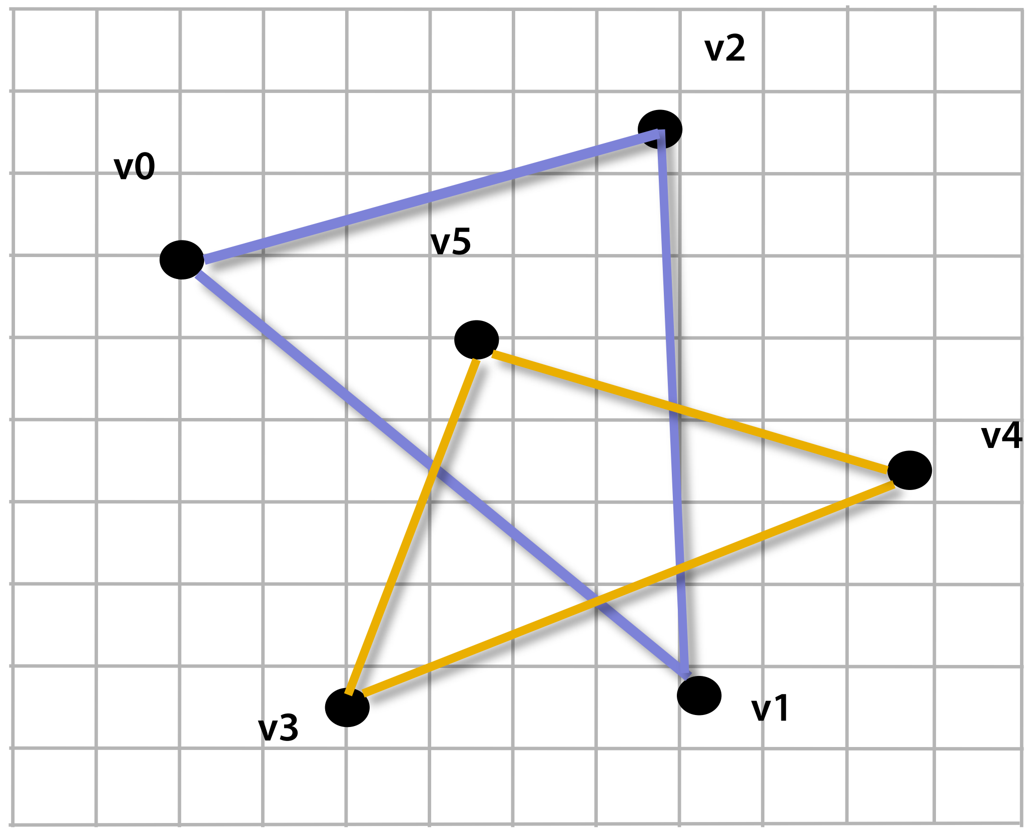
Vertices



Primitive processing

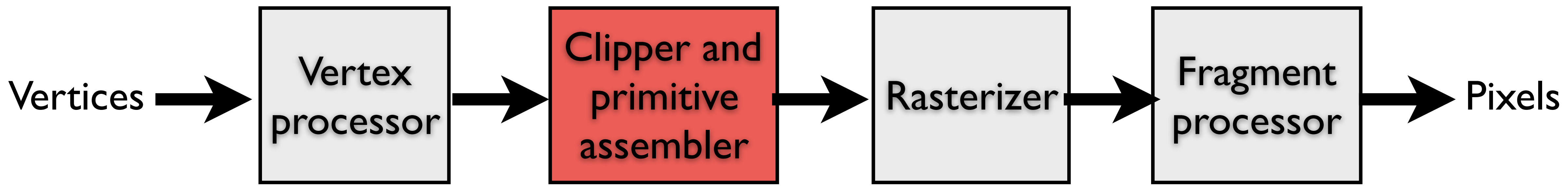
Then organized into primitives that are clipped and culled…





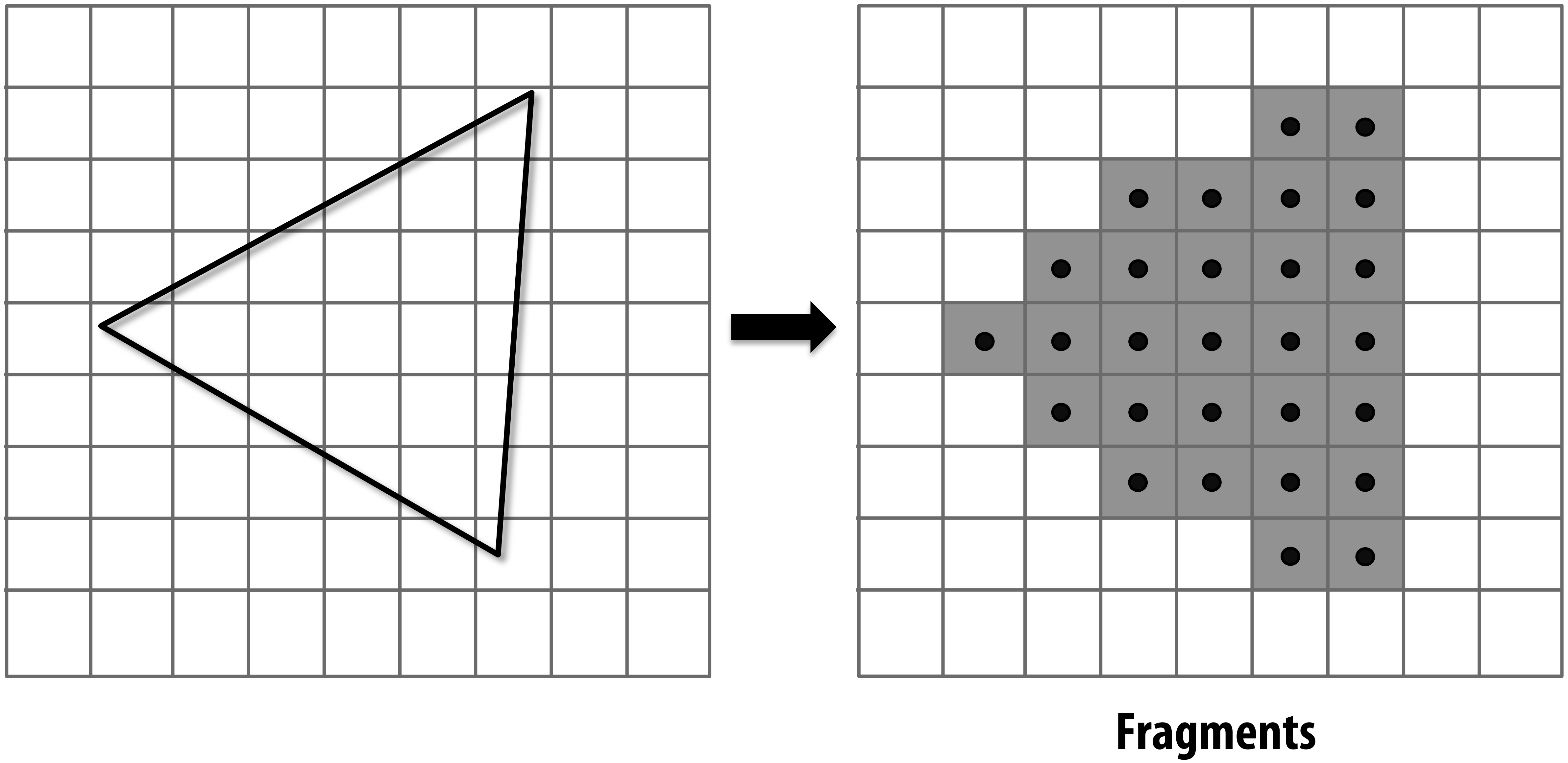
Vertices

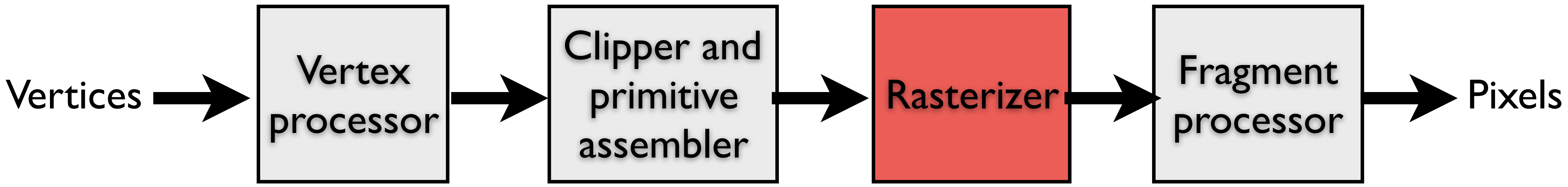
Primitives (triangles)



Rasterization

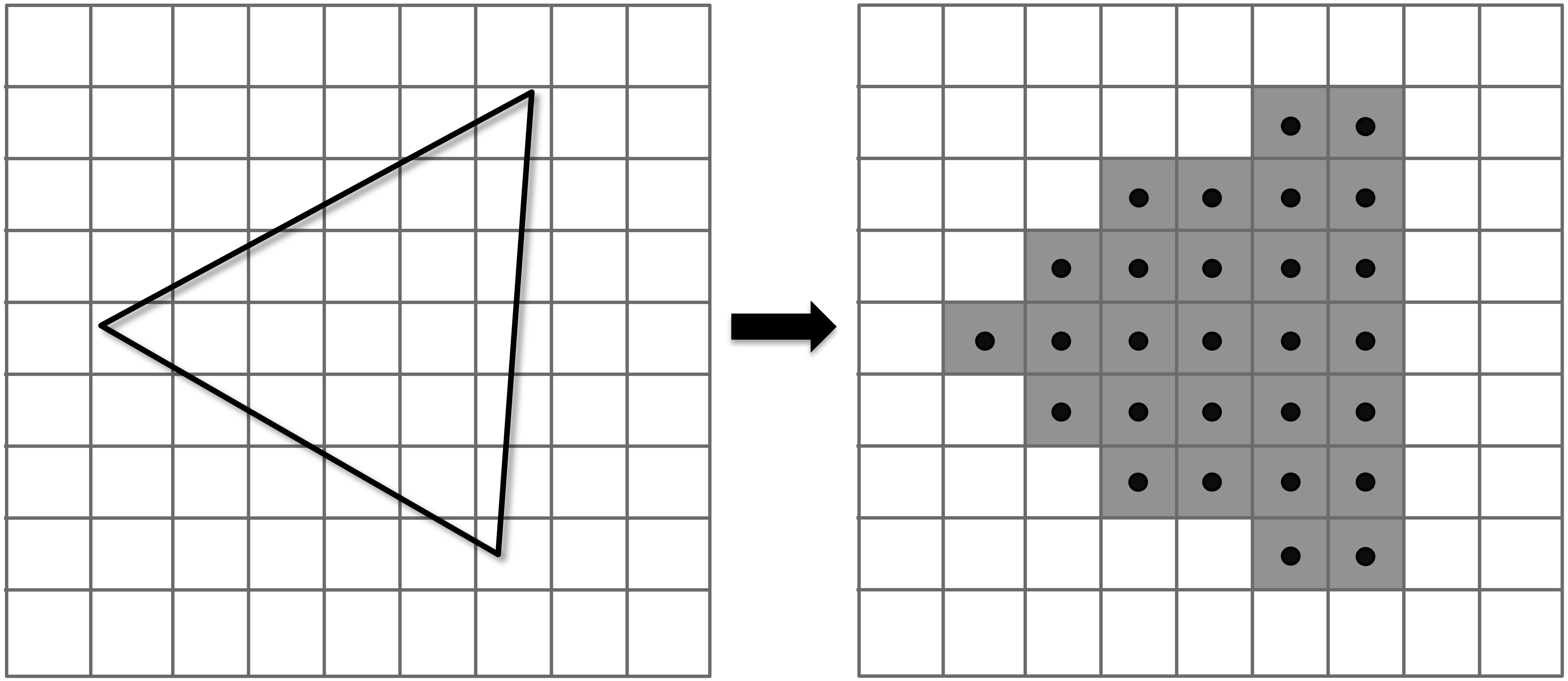
Primitives are rasterized into "pixel fragments"



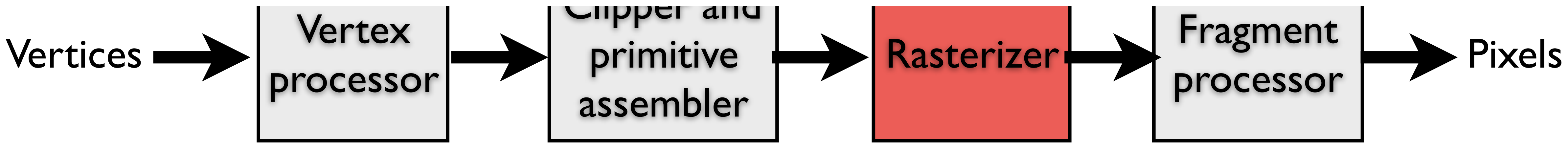


Rasterization

Primitives are rasterized into “pixel fragments”

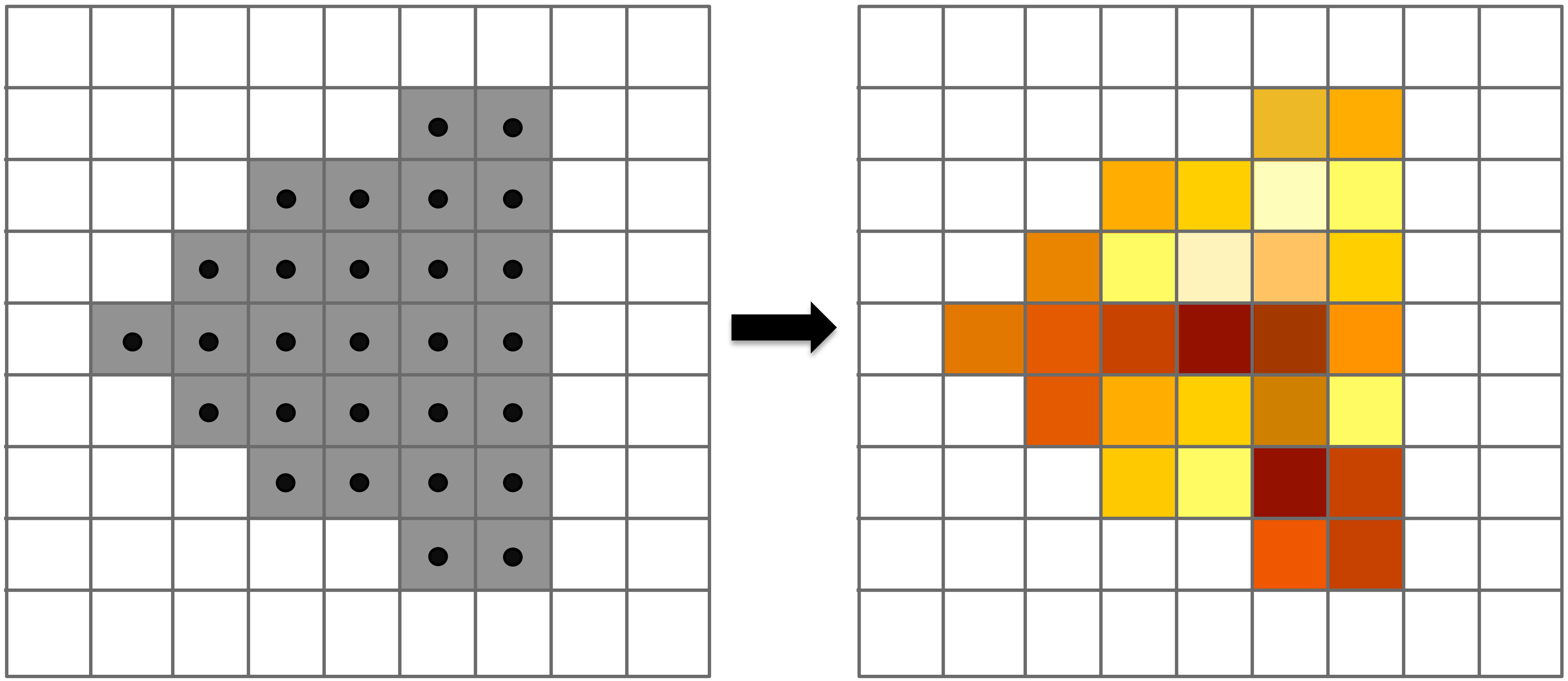


EACH PRIMITIVE IS RASTERIZED INDEPENDENTLY

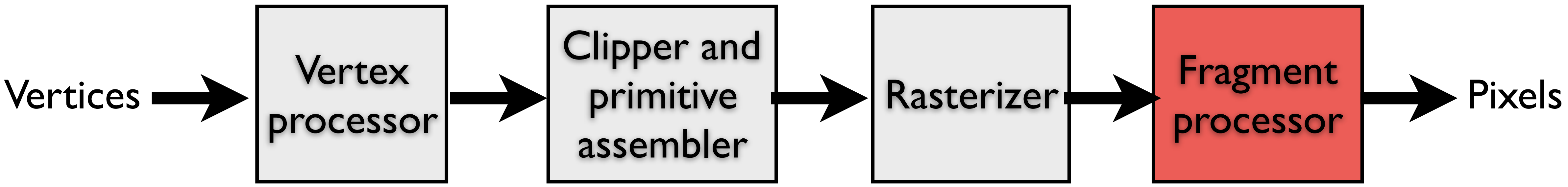


Fragment processing

Fragments are shaded to compute a color at each pixel

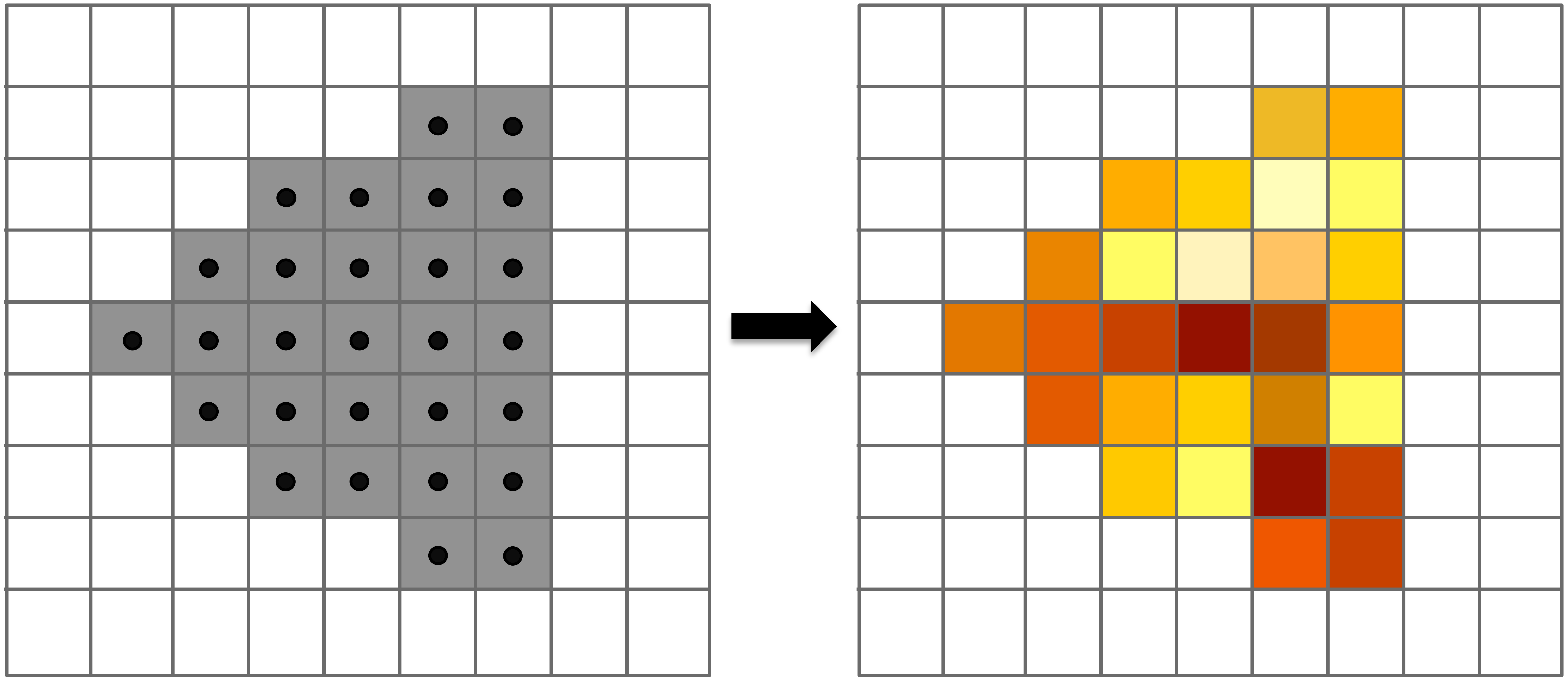


Shaded fragments

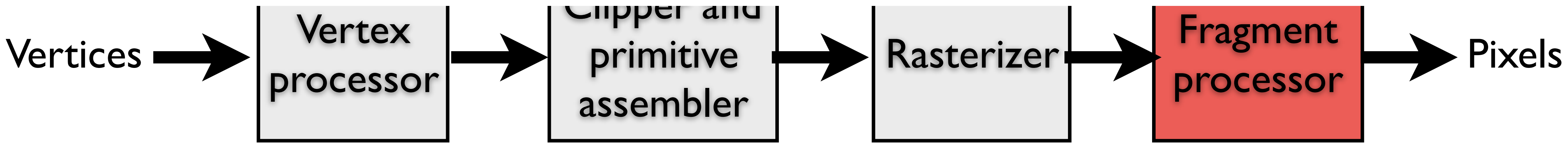


Fragment processing

Fragments are shaded to compute a color at each pixel

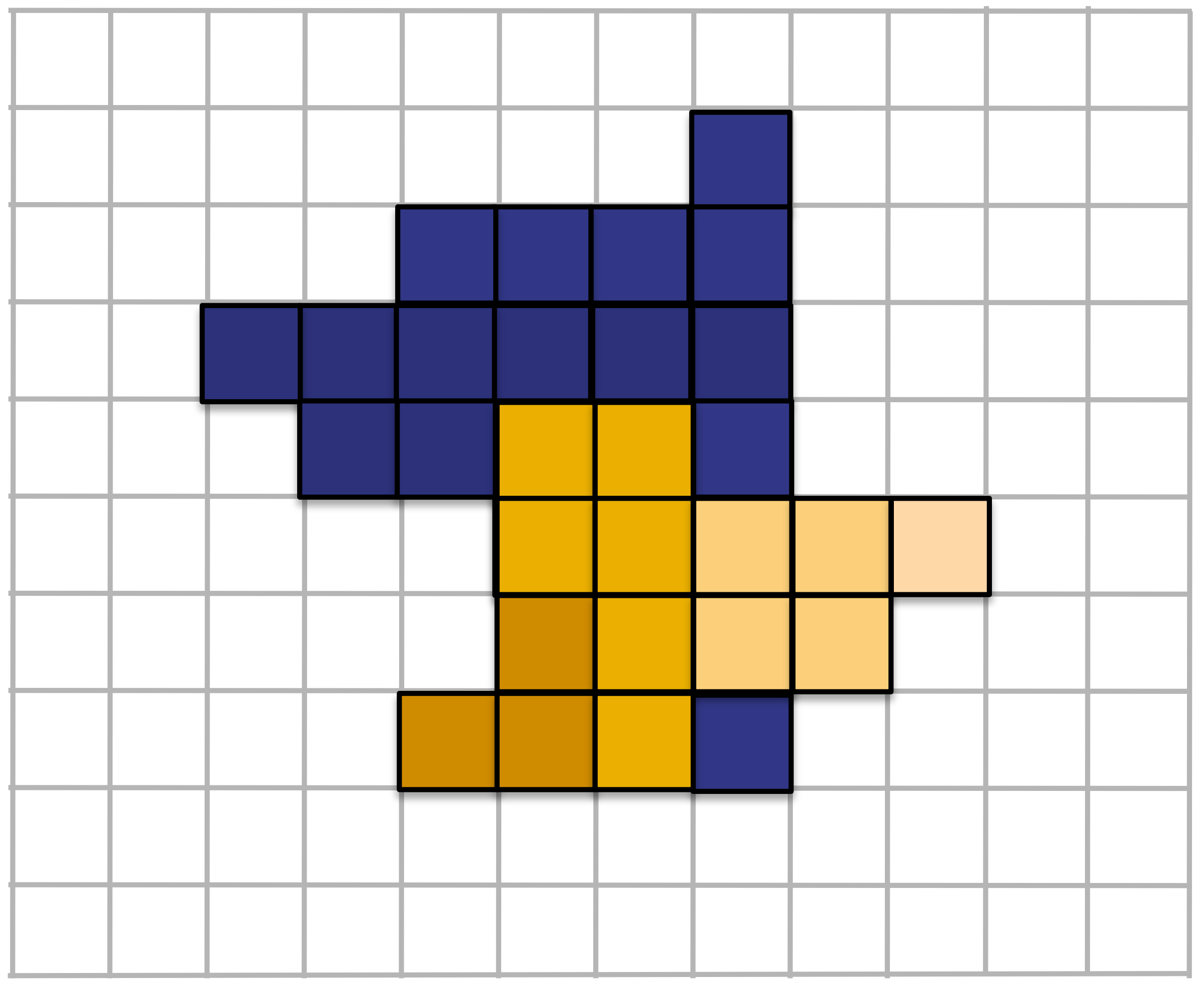


EACH FRAGMENT IS PROCESSED INDEPENDENTLY



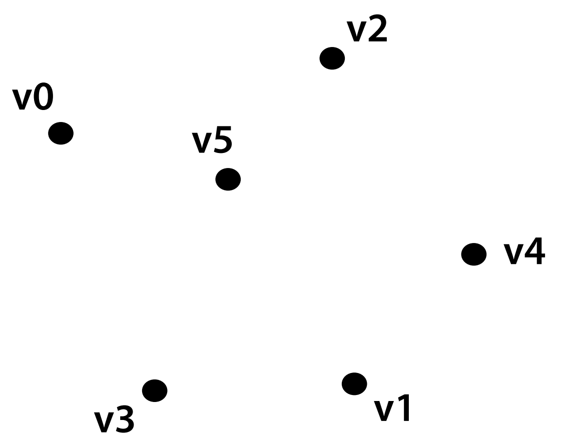
Pixel operations

Fragments are blended into the frame buffer at their pixel locations (z-buffer determines visibility)

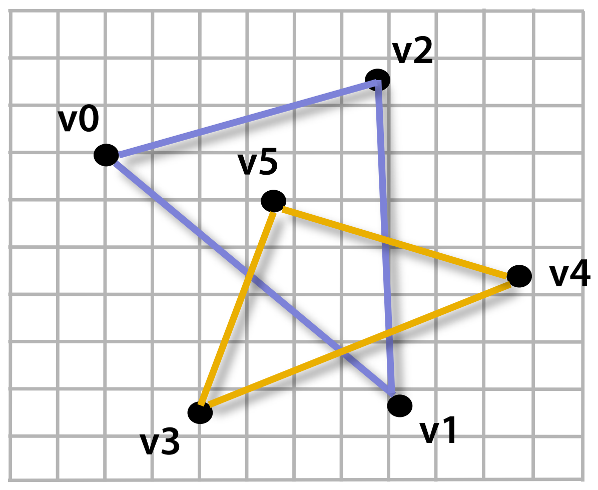


Pixels

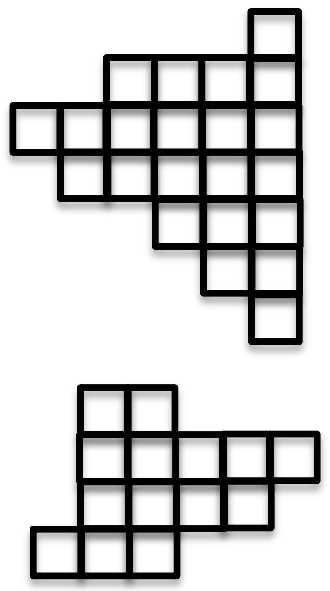
Pipeline entities



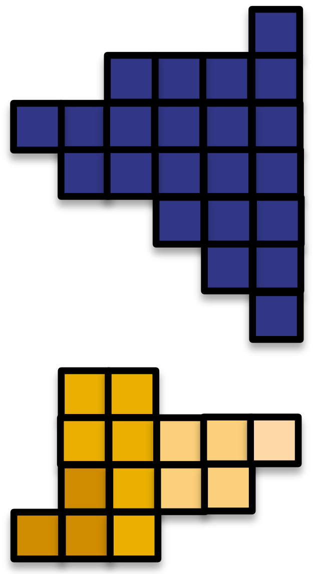
Vertices



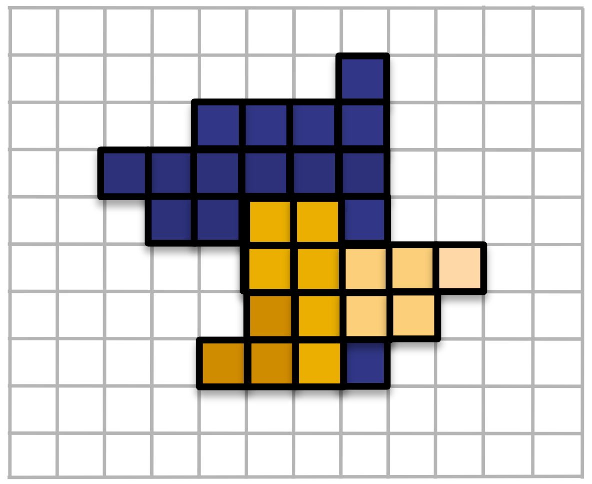
Primitives



Fragments

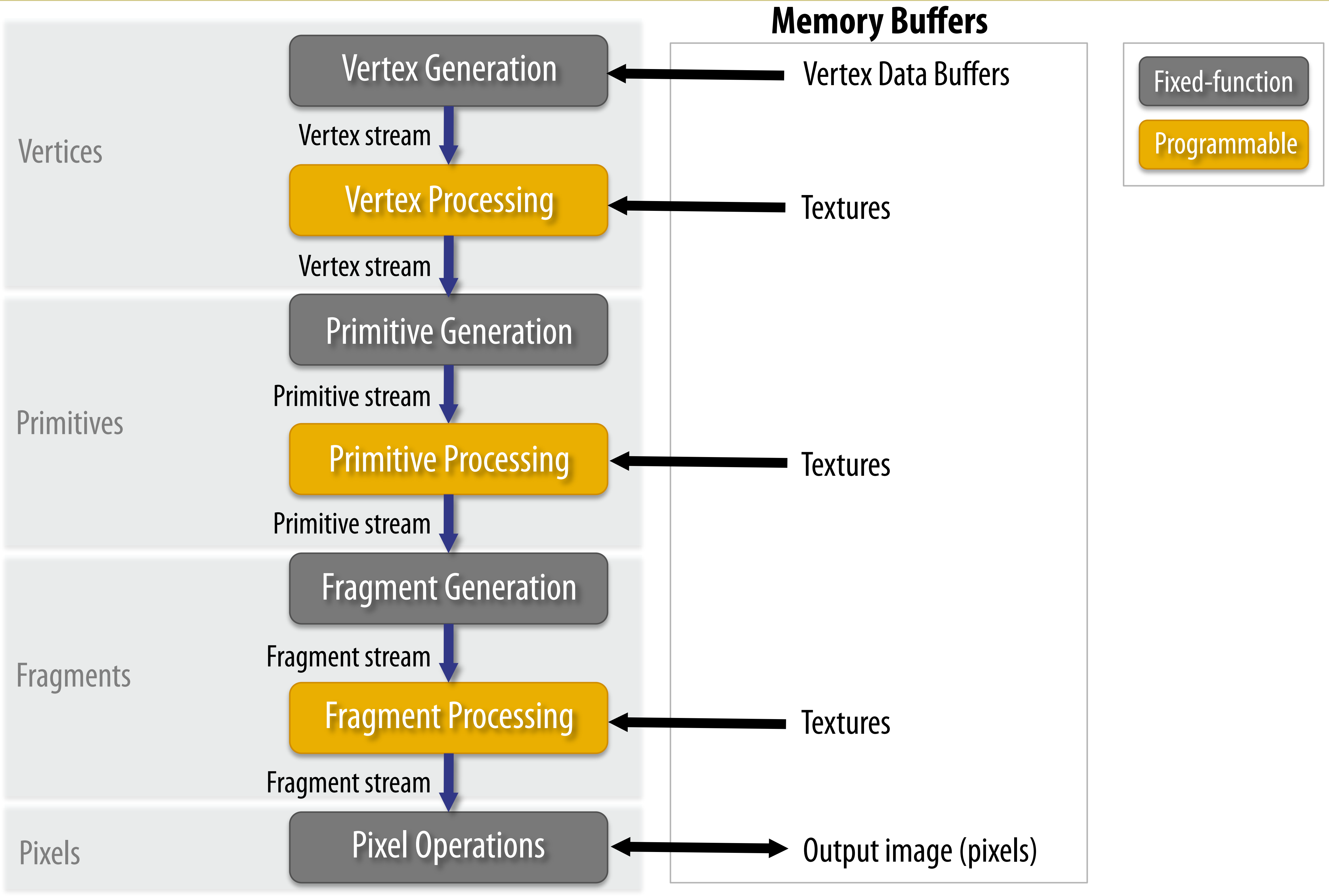


Fragments (shaded)



Pixels

Graphics pipeline


Memory Buffers
Vertices
Primitives
Fragments
Pixels
Vertex Generation
Vertex stream
Vertex Processing
Vertex stream
Primitive Generation
Primitive stream
Primitive Processing
Primitive stream
Fragment Generation
Fragment stream
Fragment Processing
Fragment stream
Pixel Operations
Vertex Data Buffers
Textures
Textures
Textures
Output image (pixels)
Fixed-function
Programmable