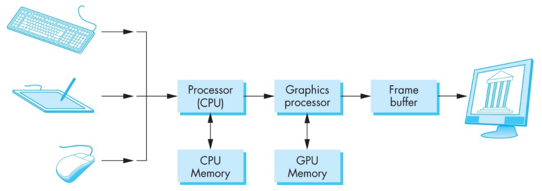
Graphics Systems

A Basic Graphics System



Input devices

Output device

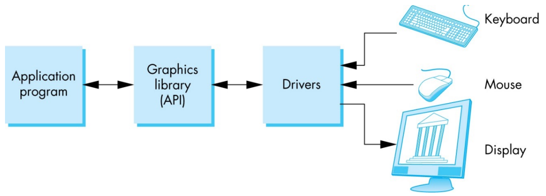
Image formed in frame buffer

Framebuffer

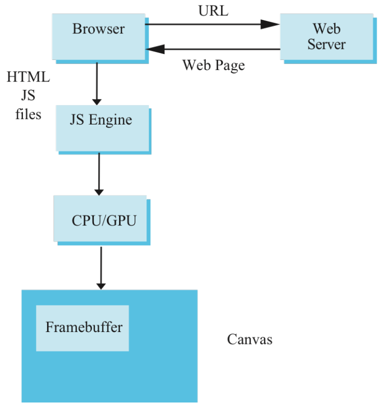
a collection of buffers that can be used as the destination for rendering. OpenGL has two kinds of framebuffers: the Default Framebuffer, which is provided by the OpenGL Context; and user-created framebuffers called Framebuffer Objects (FBOs). The buffers for default framebuffers are part of the context and usually represent a window or display device.

As defined by OpenGL.org

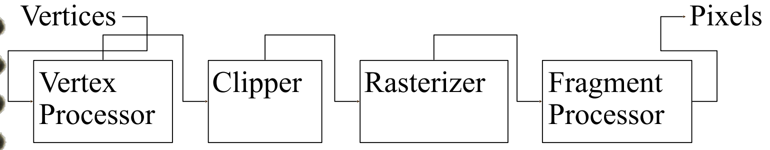
OpenGL System



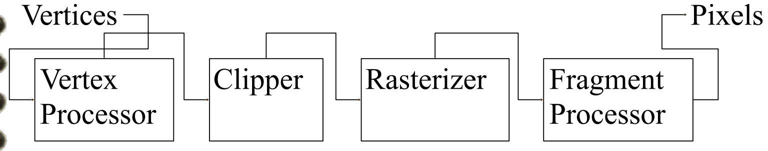
Execution in Browser



Graphics Pipeline



Vector Processor

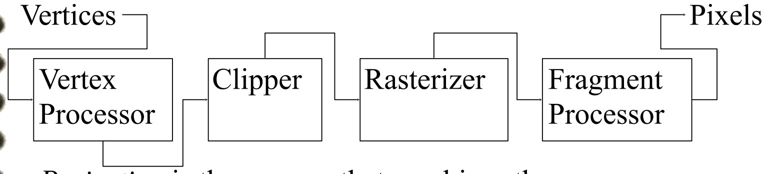


Converts object representations from one coordinate system to another

- Object coordinates
- Camera (eye) coordinates
- Screen coordinates

Computes vertex colors

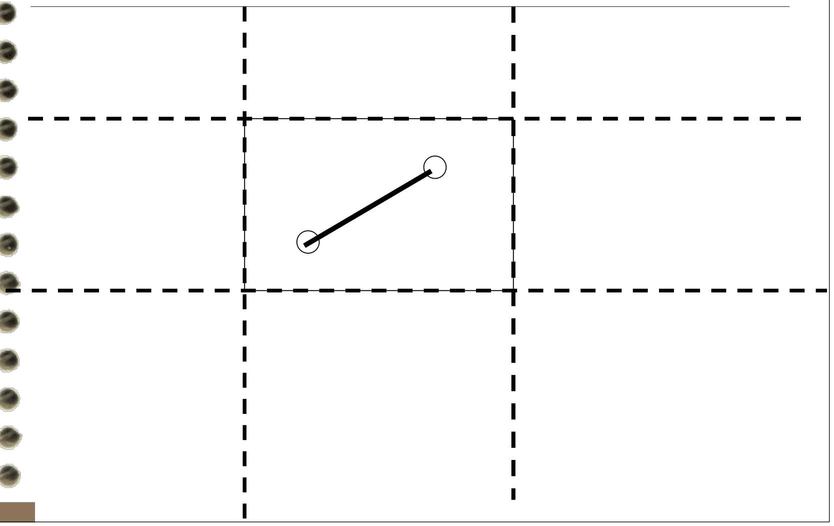
Clipper



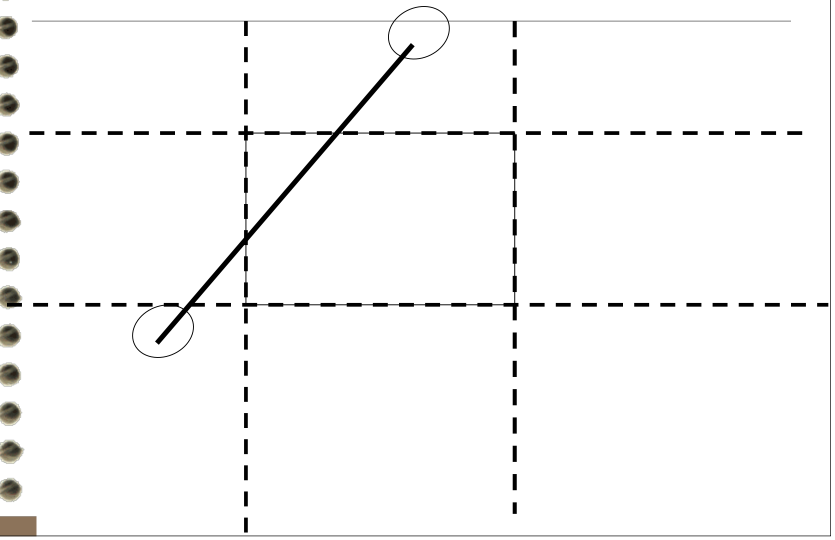
Projection is the process that combines the 3D viewer with the 3D objects to produce the 2D image

The primitives(lines and polygons) are assembled and determined what is in view.

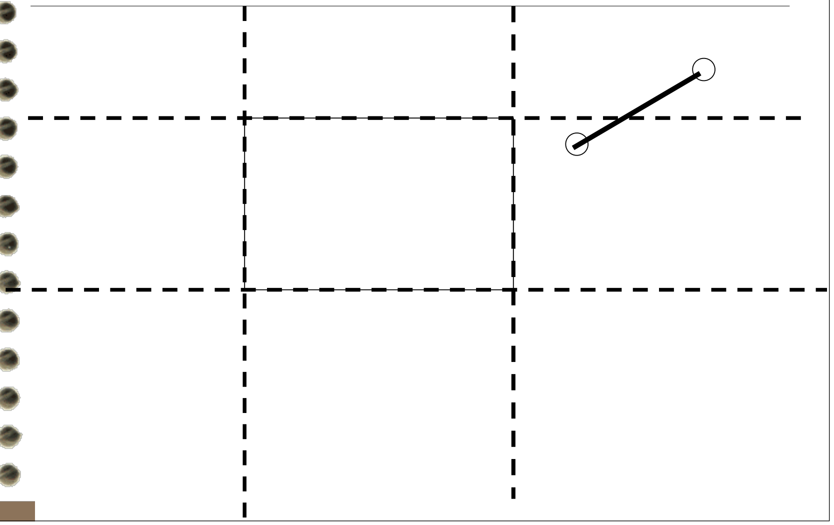
Clipping Regions



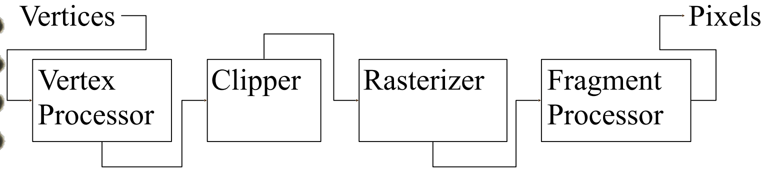
Clipping Regions



Clipping Regions



Rasterizer



Vertices

Vertex Processor

Clipper

Rasterizer

Fragment Processor

Pixels

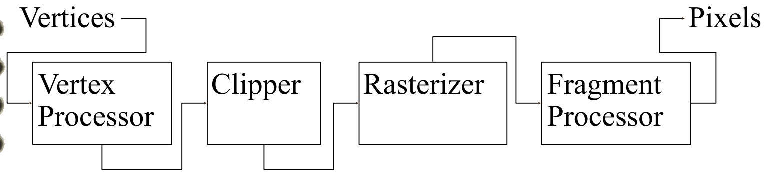
Rasterizer produces a set of fragments for each object

Fragments are “potential pixels”

- Have a location in framebuffer
- Color and depth attributes

Vertex attributes are interpolated over objects by the rasterizer

Fragment Processor



Vertices

Vertex Processor

Clipper

Rasterizer

Fragment Processor

Pixels

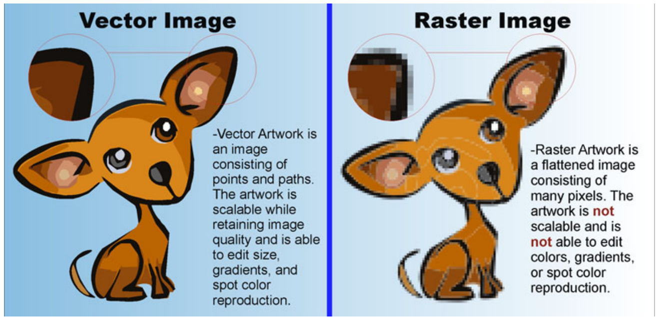
Fragments are processed to determine the color of the corresponding pixel in the frame buffer

Colors can be determined by texture mapping or interpolation of vertex colors

Fragments may be blocked by other fragments closer to the camera

Hidden-surface removal

Vector vs Raster Graphics



Vector Image

-Vector Artwork is an image consisting of points and paths. The artwork is scalable while retaining image quality and is able to edit size, gradients, and spot color reproduction.

Raster Image

-Raster Artwork is a flattened image consisting of many pixels. The artwork is **not** scalable and is **not** able to edit colors, gradients, or spot color reproduction.

Image from:

Vector vs Raster Graphics

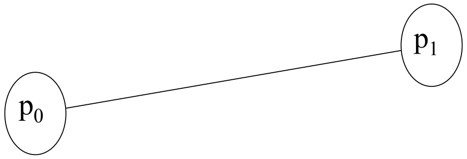
- Vector Graphics: Mathematical representation of lines and shapes (files: .ai, .pdf, .obj)
  - Pros:
    - Scalable
    - Smaller file size
    - Editable
  - Cons:
    - Limited details
    - Limited Effects
- Raster Graphics: Image as an array of pixels (Files: .gif, .jpg, .bmp)
  - Pros:
    - Detail on pixel level
    - Edit on pixel level
  - Cons:
    - Not Scalable. Lose detail when enlarged.
    - Large file size.

Vector Graphics: Equation of Line

The vector equation of a line containing the points p_0 and p_1:

$$s(t) = (p_1 - p_0)t + p_0$$

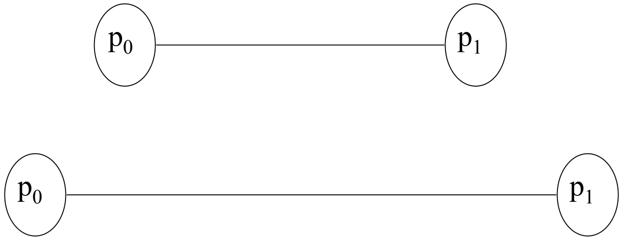
We get the line segment p_0p_1 when we limited t to the interval [0,1].



What should be stored to represent this line?

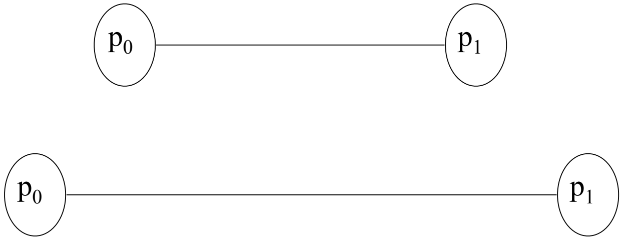
Vector Graphics: Scalable

How do you scale lines?



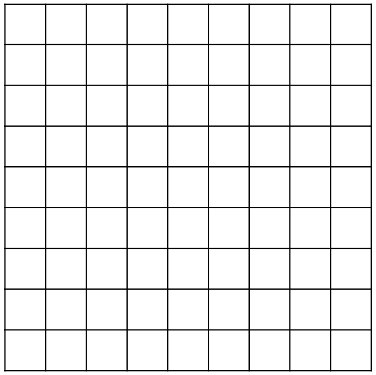
Vector Graphics: Scalable

How do you scale lines?



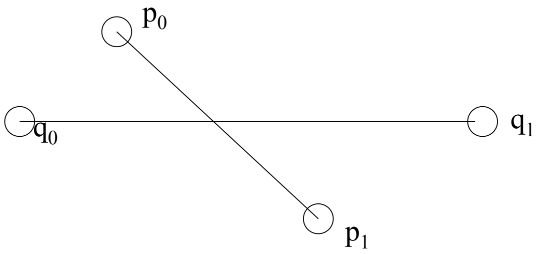
Scalar multiply the points.

Raster Graphics: A Line



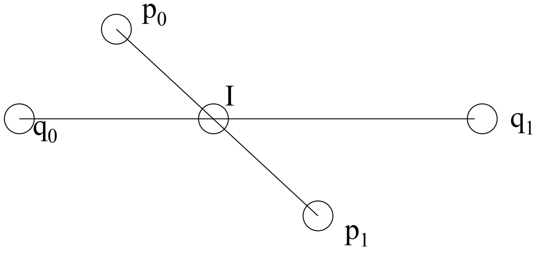
How do you scale these lines?

Vector Graphics: Computing the Intersection of two lines

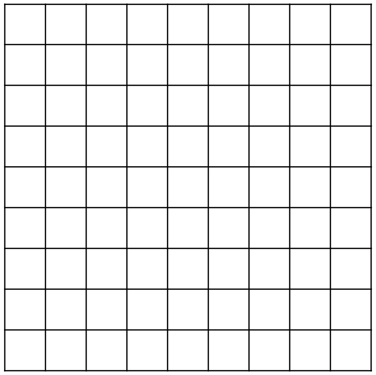


Vector Graphics: Computing the Intersection of two lines

$$I = (p_1 - p_0)t_1 + p_0$$
$$I = (q_1 - q_0)t_1 + q_0$$



Raster Graphics: Intersection



We can “see” this intersection…but how do we compute it?