Lecture slides (CT4201/EC4215 – Computer Graphics)

Graphics Pipeline

Lecturer: Bochang Moon

So far…

- We studied the following:
  - Basic programming with OpenGL
  - Transformations with some mathematical background

- Now you are ready to study a graphics pipeline

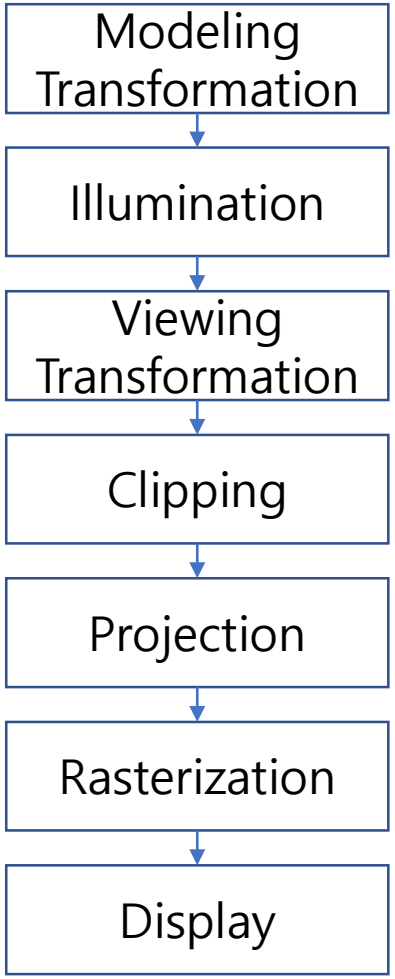
Why Graphics Pipeline?

- Graphics (Rendering) Pipeline
  - Objective: draw virtual objects (3D) in your screen (2D)
  - Commonly used for real-time applications
  - Consists of multiple transformations

- Scope for graphics pipeline in this class
  - Classic graphics pipeline for understanding basics of computer graphics

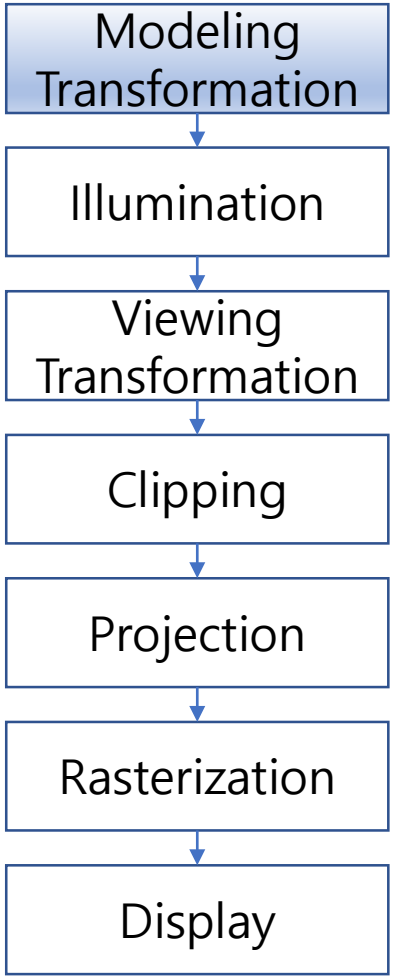
- Example applications

Traditional Rendering Pipeline

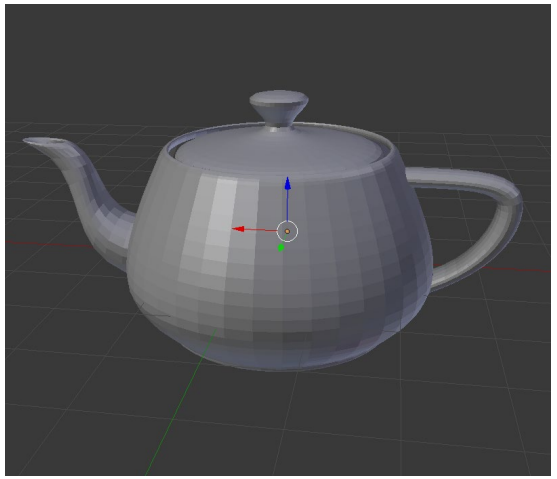


- Input:
  - Geometric model
    - e.g., primitives
- Output:
  - Colors (e.g., 24-bit RGB value at each pixel)

Modeling Transformation

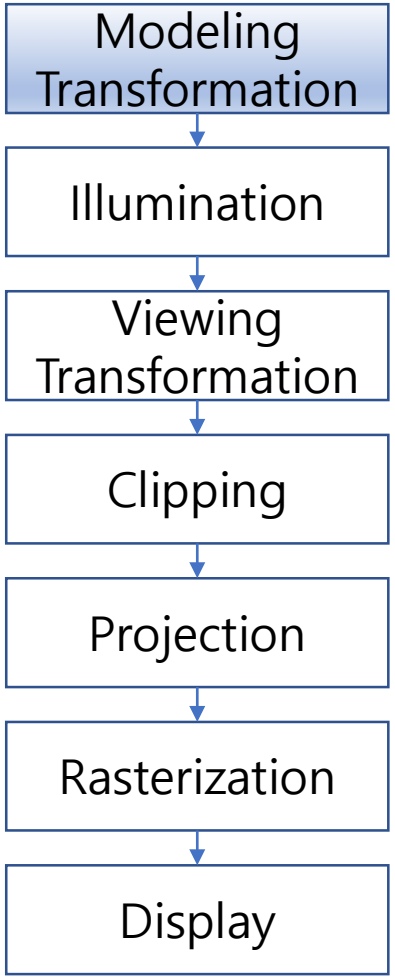


- 3D models are defined in a object space



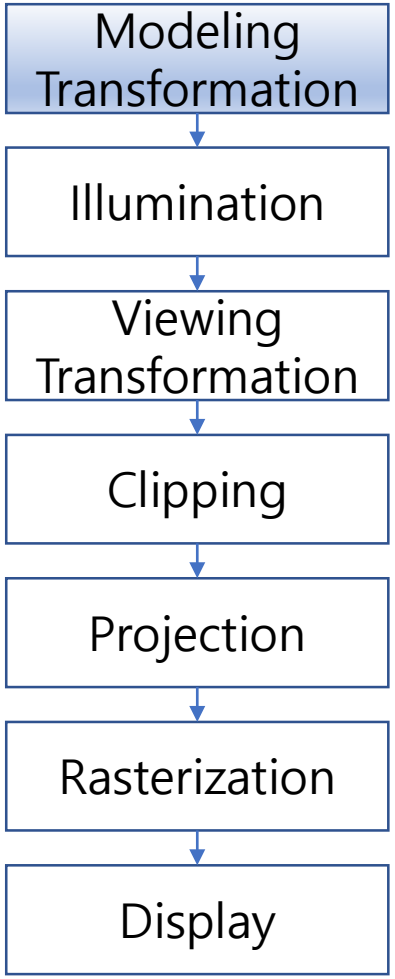
  - How do we design each model?
    - Usual modeling tools: 3DS MAX and Maya (commercial), Blender (free to use)
  - Where do we get 3D models?
    - e.g., pbrt.org and many websites (free for your research), TurboSquid (commercial)

Modeling Transformation



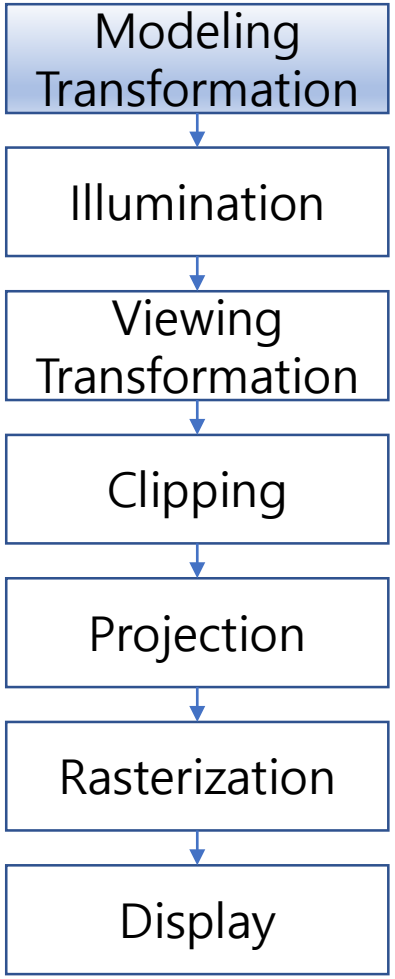
- 3D models are defined in *object spaces*
- We usually want to render a scene that contains multiple objects
  - Need to arrange all your 3D models in a unique space (*world space*)
- In the world space
  - All 3D models
  - Light sources
  - Camera

Modeling Transformation



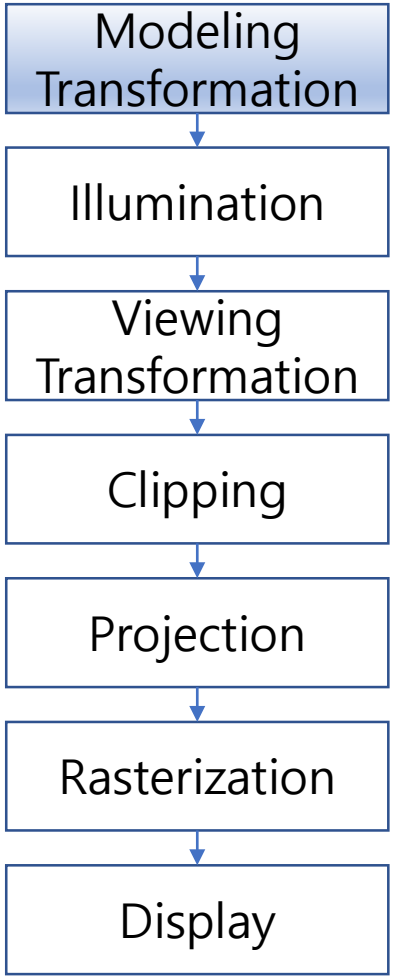
- Why do we need to use modeling transformation?
  - Example scene that does not perform a proper modeling transformation

Modeling Transformation

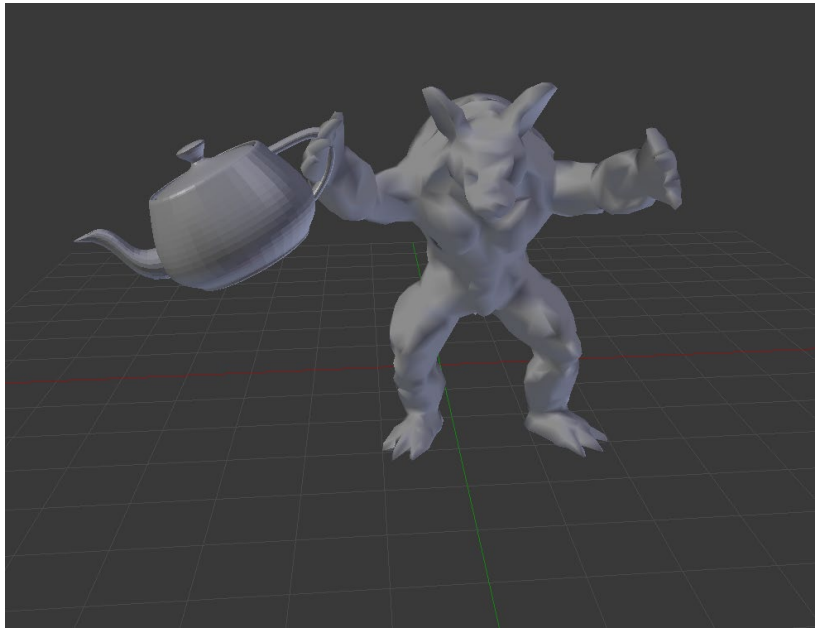


- Why do we need to use modeling transformation?
  - Example scene that does not perform a proper modeling transformation
  - After
    - Some modeling transformations (e.g., scaling, translation)

Modeling Transformation

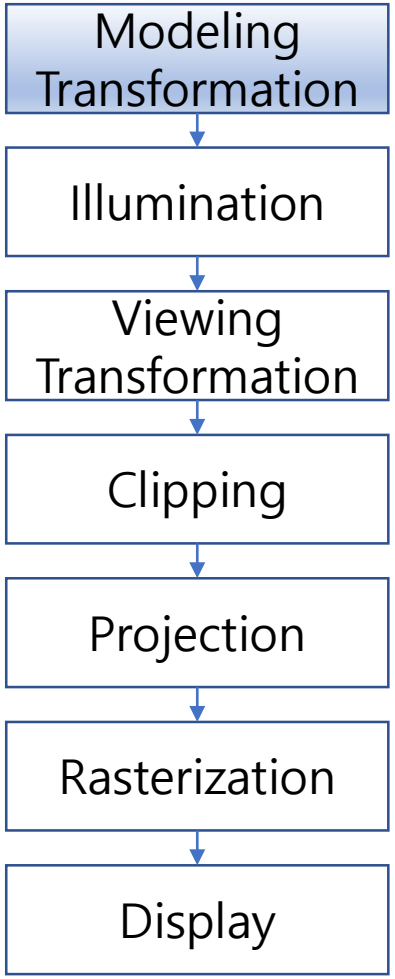


- Why do we need to use modeling transformation?
  - Example scene that does not perform a proper modeling transformation
  - After
    - Some modeling transformations (e.g., scaling, translation)
    - More modeling transformations

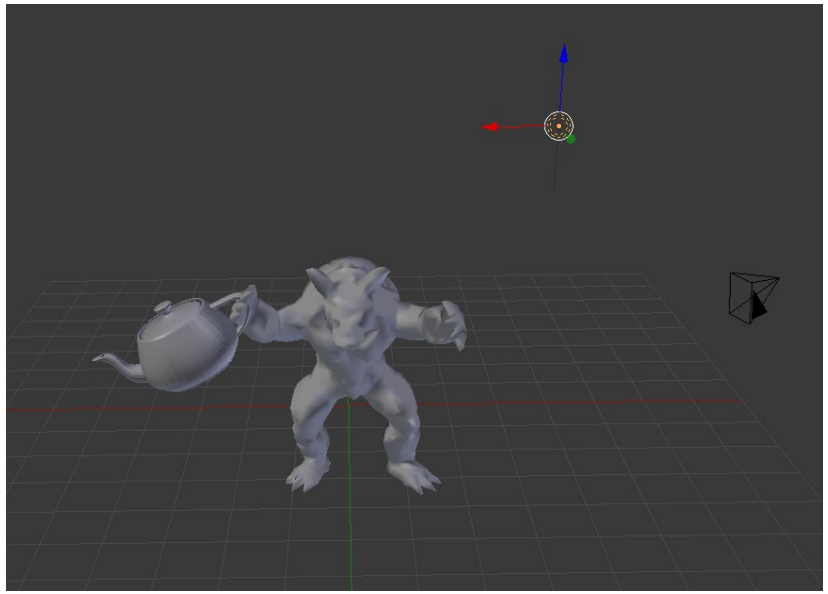


I designed it based on two existing models!

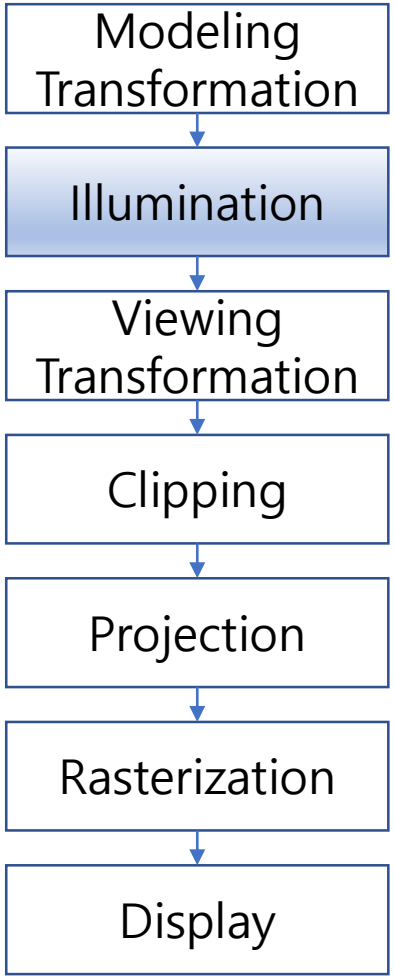
Modeling Transformation



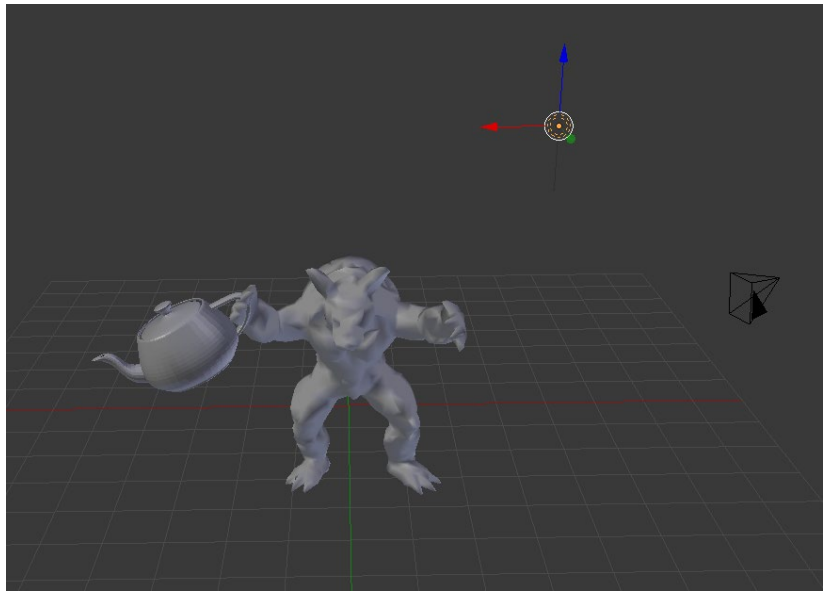
- Need to arrange:
  - All models in world space
  - Light sources
  - Camera



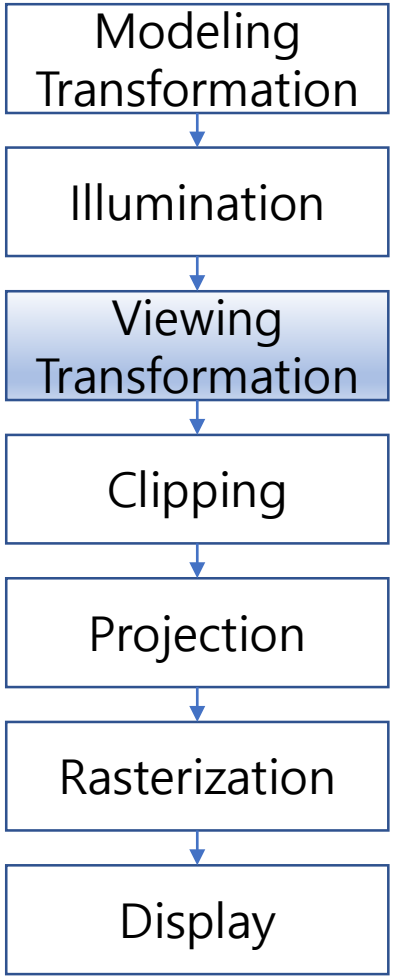
Illumination



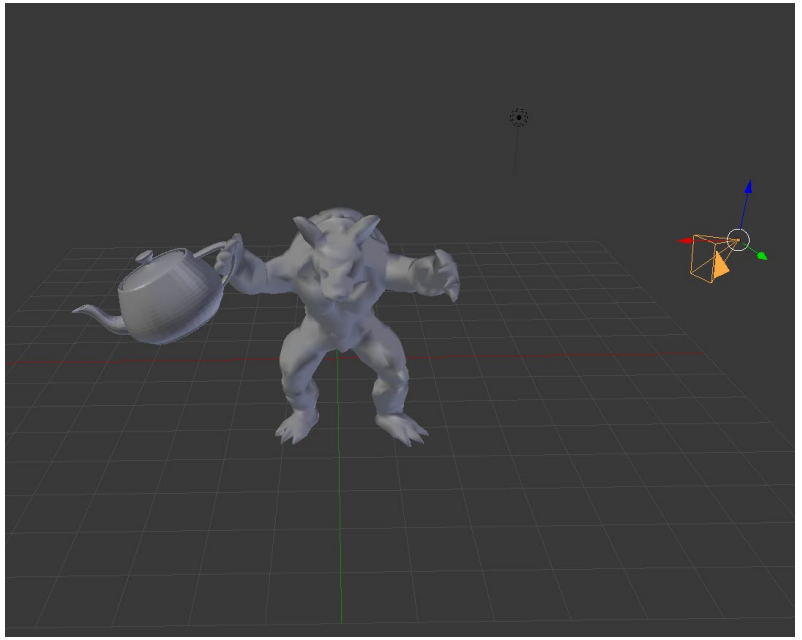
- Illuminate 3D objects according to lighting and reflectance
  - Q. When do we need to define materials of 3D objects?
    - Generally define materials of each object when designing models



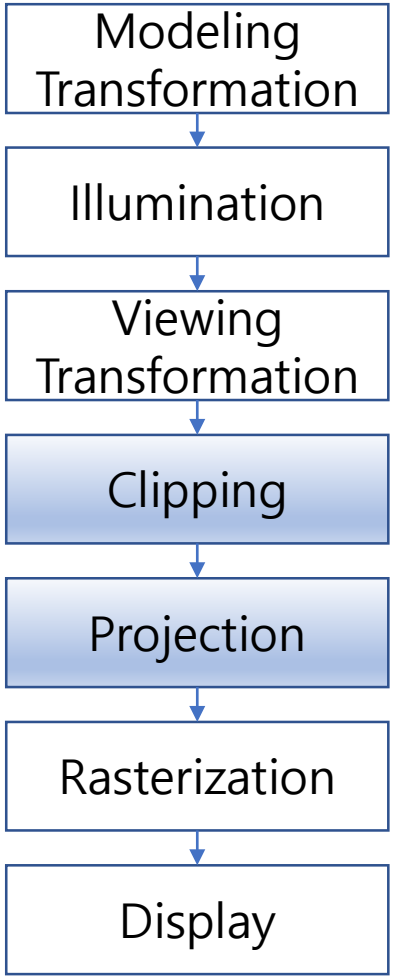
Viewing Transformation



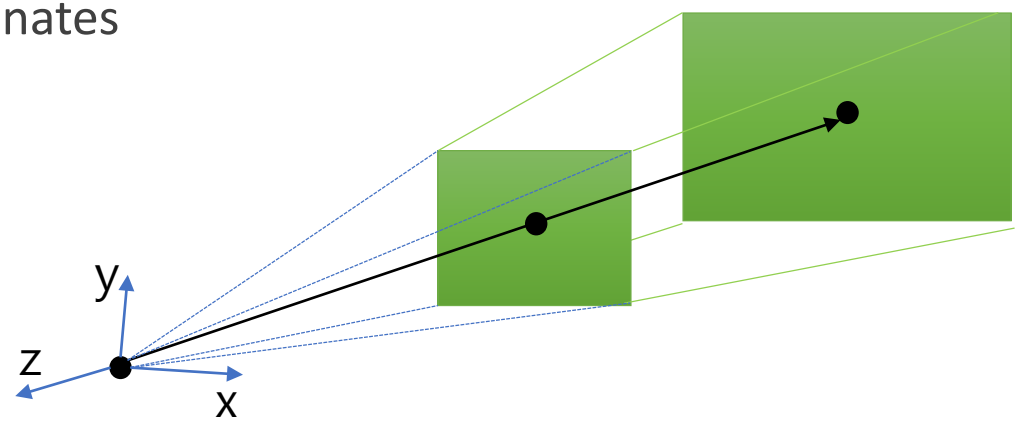
- Transform all points from world space to *eye space*
  - Camera position transforms into the origin



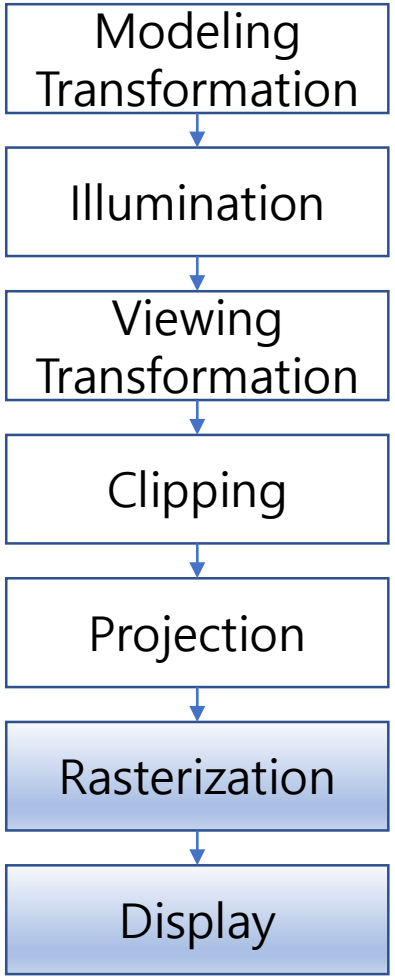
Clipping and Projection



- A volume, *viewing frustum*, is specified from the camera
- Map the frustum to the unit cube
- Clip objects against the volume (remove non-visible geometry from your eye)
- Project objects into 2D plane
- Transform from eye space to normalized device coordinates



Clipping and Projection



- Transform normalized device coordinates to screen space
- Rasterize the objects to fill color values at pixels

- We have observed:
  - Most components in the graphics pipeline are transformations.