OpenGL Basics

What is OpenGL?

- OpenGL is a software specification. It acts as a layer between our program and graphics driver.
- OpenGL is implemented as a software interface that allows the programmer to create 2D and 3D graphics images.

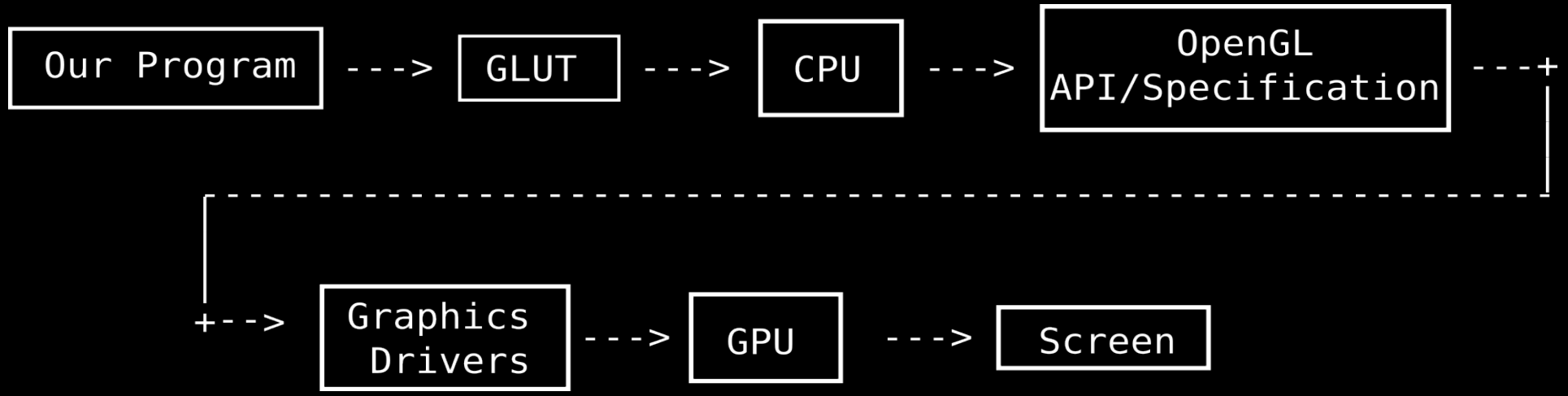
Why OpenGL?

- Device independence
- Platform independence
  - SGI Irix, Linux, Windows
- Abstractions (GL, GLU, GLUT)
- Open source
- Hardware-independent software interface
- Support of client-server protocol
- Other APIs
  - OpenInventor (object-oriented toolkit)
  - DirectX (Microsoft)
  - Java3D (Sun)

Modes of operation/design:

- Immediate mode: draw calls directly render to screen.
- Retained mode: draw calls fill a buffer(abstract data model) and OpenGL decide when to render.
- We’ll be dealing with immediate mode.

Immediate mode pipeline:



Primitives

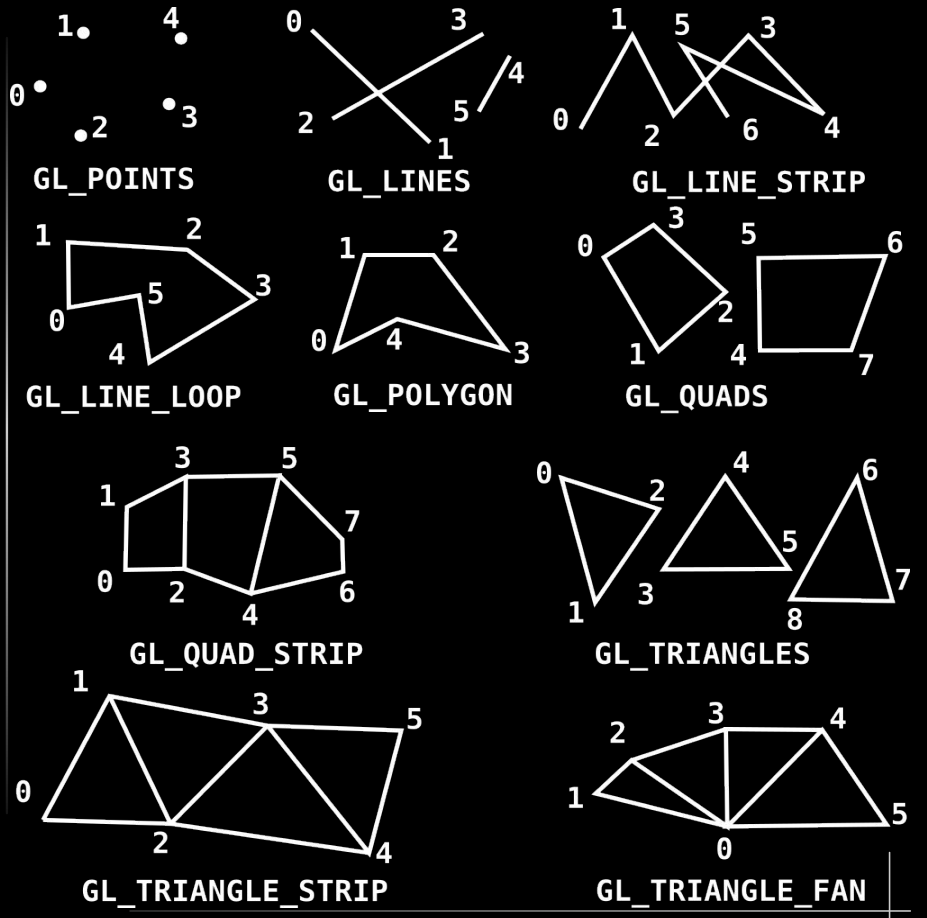
- Supports several basic primitives such as points, lines, triangles, quadrilaterals and general polygons.
- Primitives are specified using a sequence of vertices.
- Syntax:

```
glBegin(<primitive type>);
    glVertex2f(x0,y0);
    glVertex2f(x1,y1);
    .
    .
glEnd();
```

- Where x0, y0, x1, y1 are variables of GLfloat type.

Primitive types:

- The numbers indicate the order in which the vertices have been specified.

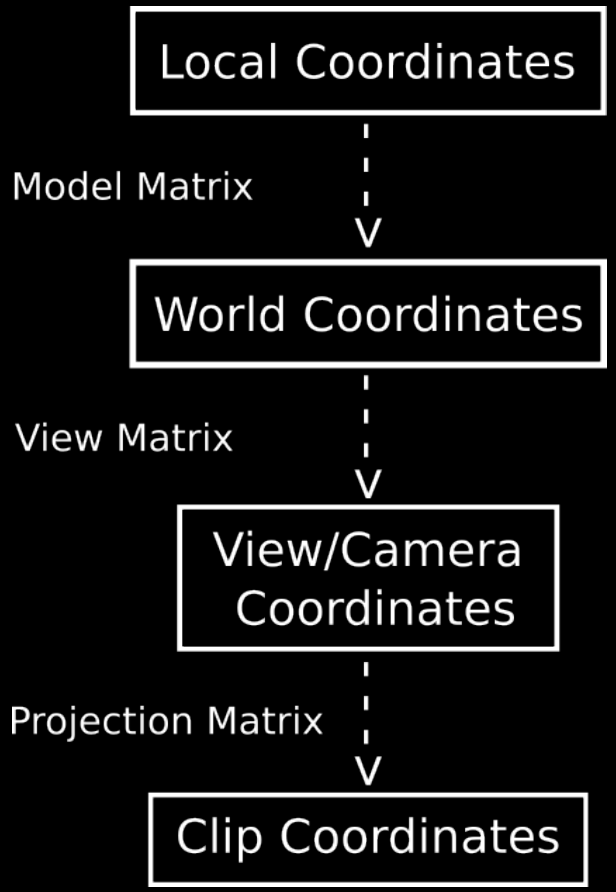


Colors in OpenGL

- Colors are specified in RGB/RGBA mode. Values of [0...1] or [0...225] are acceptable depending on the function called.
- Function used: glColor();
- Function format: glColor[data type nemonic]();
- glColor() has different forms depending upon the color values you want to specify.
  - For eg.:
  - glColor3f(0.0f, 1.0f, 0.0f); // sets green color using GLfloat.
  - GlColor3ui(255,0,0); // sets color to red using GLuint.
  - glColor4f(1.0f, 0.0f, 0.0f, 1.0f); // sets color to red using GLfloat and sets alpha value to max.
  - glColor3fv(cv); // sets color based on cv of type GLfloat *.

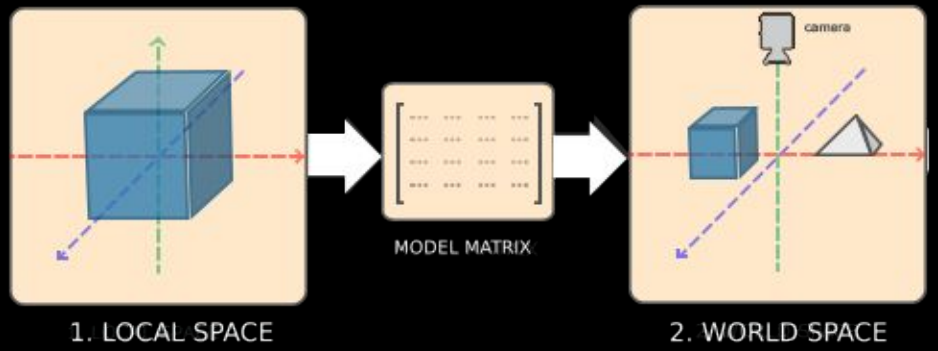
Coordinate system in OpenGL

- Different coordinates are used to make calculations easier.
- OpenGL expects coordinates to be in Normalised Device Coordinates(NDC) range -1 to 1.
- Models are defined based on their local origin.



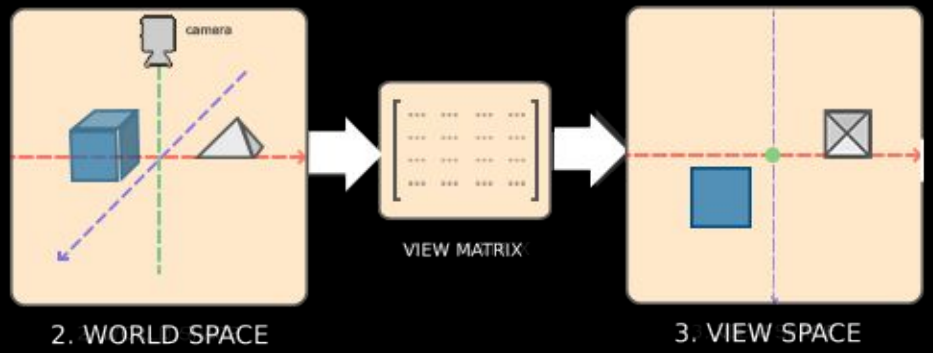
Model Matrix

- Local coordinates are relative to its local origin at (0,0,0).
- The model matrix transforms a position in a model to the position in the world by combining translation, scale and rotation.



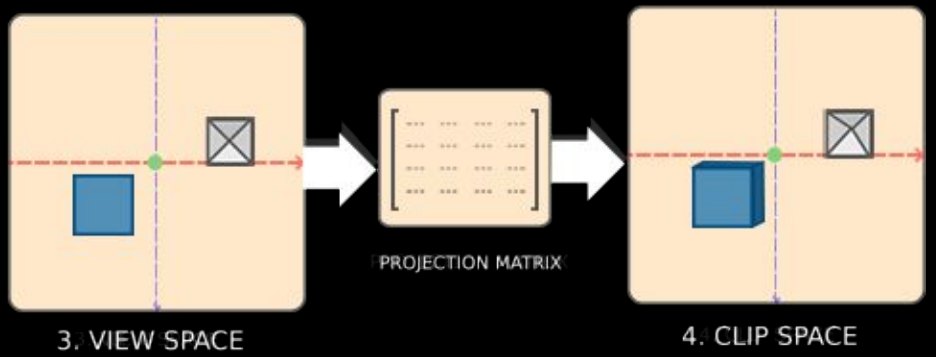
View / Camera matrix

- View space coordinates are relative to the viewer(camera or eye view).
- This martix transforms the objects/vertices to mimic a camera/observer location.



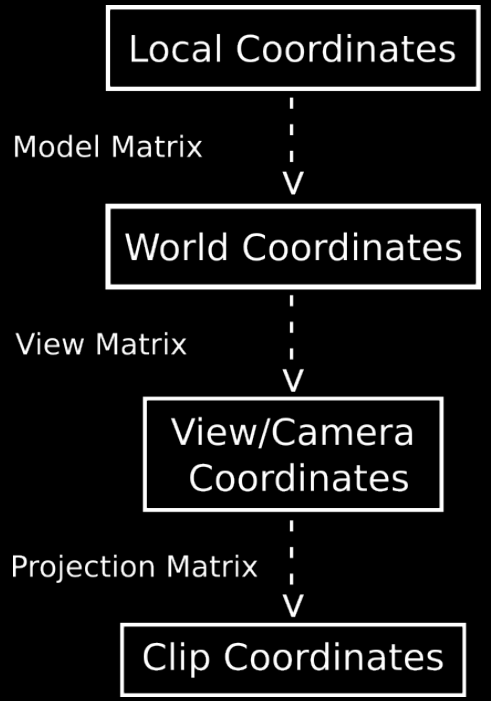
Projection matrix

- It is used to create a viewing box/frustum that defines what should be visible and what should be clipped based on viewspace coordinates.
- All the coordinates outside this frustum after transforming them to clip space get clipped.
- Projection matrix transforms the coordinates to normalised device coordinates(NDC) [-1...1].
- Projection matrices are of two types:
  - Perspective: objects farther away look smaller
  - Orthographic: objects are not affected.

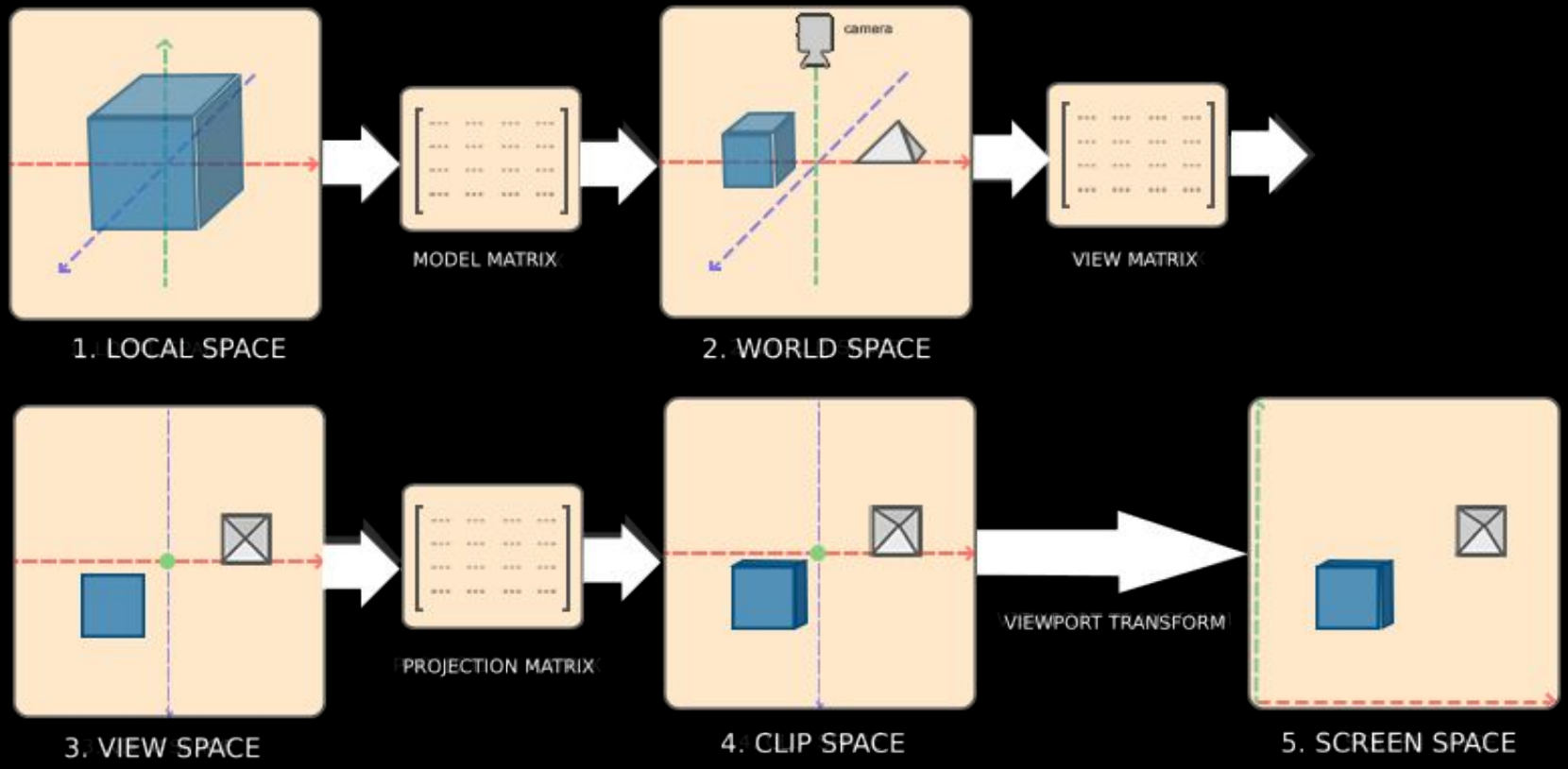


MVP matrix

- MVP martix: Model View Projection Matrix is a transformation matrix that is used to convert local coordinated to clip coordinates.
- $V_{clip} = M_{proj} \cdot M_{view} \cdot M_{model} \cdot V_{local}$



Coordinate system in OpenGL



Going 3D

- 3D graphics use the concept of MVP matrix, Lighting, shading, textures etc. to convert 3 dimensional vertices into 2D pixels.
-