James’ Guide to Software Rendering

Episode 2: A quick guide to the rendering process

There are lots of steps to rendering 3d graphics on the screen. I think it’s useful to be able to know roughly what happens and in what order.

The order I describe here is based on one of my previous software rasterisers so I know this sequence works. If you look at the old OpenGL and DirectX pipelines you might find that things are slightly different but the basics are the same. If you look at modern programmable hardware you’re likely to find all sorts of extra stages as well.

# A simple pipeline

The pipeline metaphor gets used a lot. It’s not a bad metaphor either – when sending a shape to be drawn on the screen it has to go through a sequence of steps; each step is dependent on the previous one so.

My simple pipeline looked like this:

|  |  |
| --- | --- |
| Basic 3d geometry transformations | Various operations are applied to the 3d vertices:   * Rotation * Translation (moving) * Scaling * Skewing/shearing etc.   These can be used to set the camera position and perform animation. |
| Perspective division | The x and y coordinates are divided by the z coordinate to correctly account for depth. |
| Clipping & polygon set-up | Shapes are clipped against the x, y, and z clipping planes. |
| Rasterization | Clipped polygons are turned into colour pixels. |