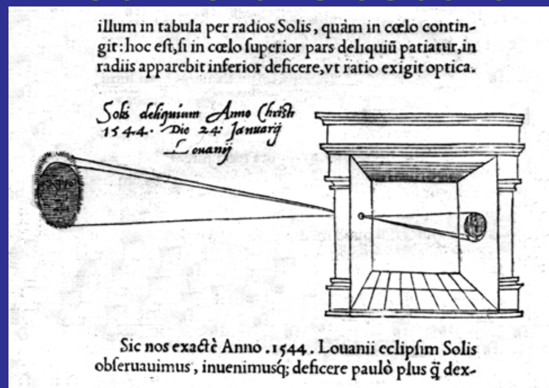
Camera Obscura



"When images of illuminated objects ... penetrate through a small hole into a very dark room ... you will see [on the opposite wall] these objects in their proper form and color, reduced in size ... in a reversed position, owing to the intersection of the rays".

Da Vinci

http://www.acmi.net.au/AIC/CAMERA_OBSCURA.html (Russell Naughton)

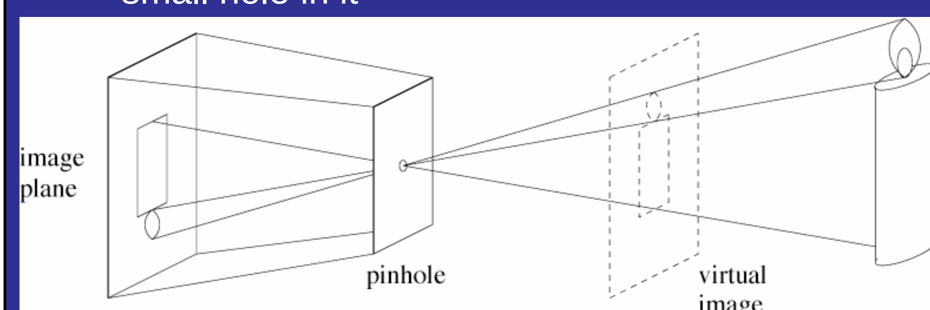
- Used to observe eclipses (eg., Bacon, 1214-1294)
- By artists (eg., Vermeer).

Jetty at Margate England, 1898.

http://brightbytes.com/cosite/collection2.html (Jack and Beverly Wilgus)

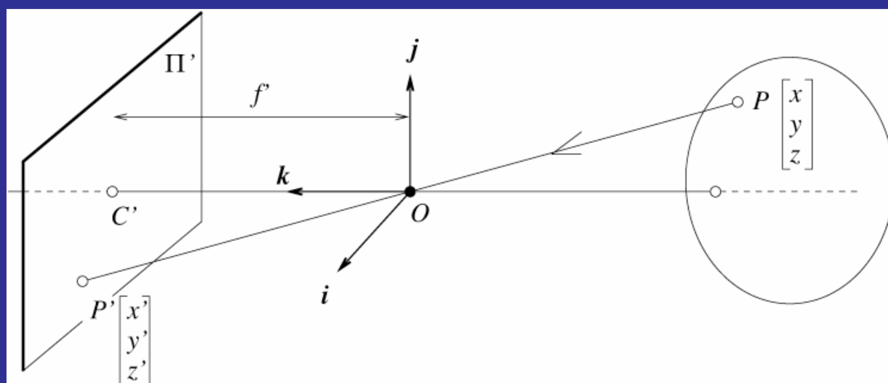
Pinhole cameras

- Abstract camera model - box with a small hole in it
- Pinhole cameras work in practice



(Forsyth & Ponce)

The equation of projection



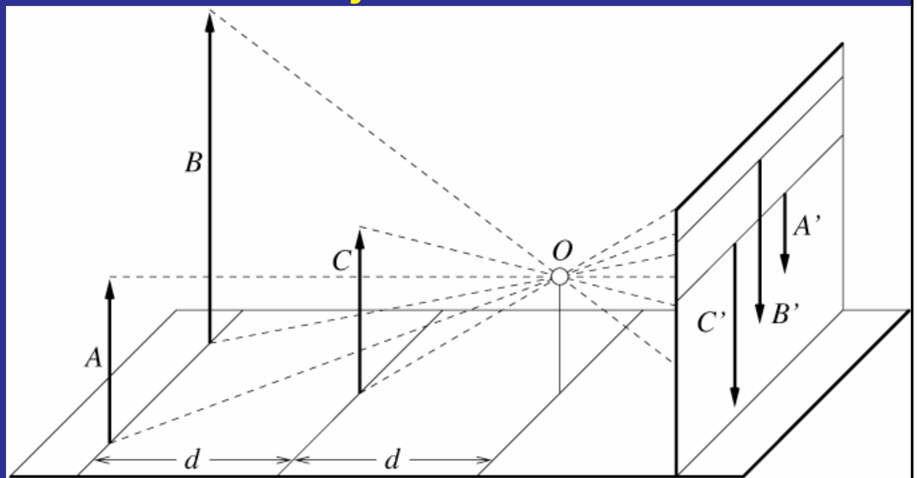
(Forsyth & Ponce)

The equation of projection

- Cartesian coordinates:
  - We have, by similar triangles, that (x, y, z) -> (f x/z, f y/z, f)
  - Ignore the third coordinate, and get

$$(x,y,z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

Distant objects are smaller



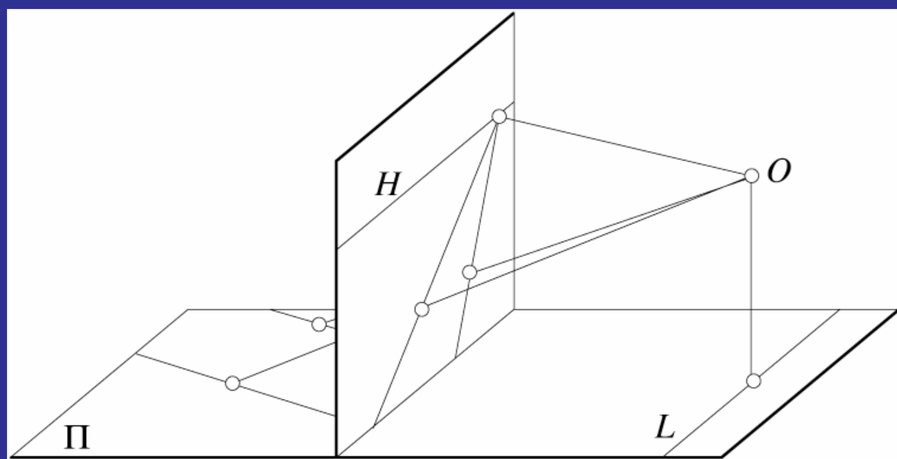
(Forsyth & Ponce)

For example, consider one line segment from (x,0,z) to (x,y,z), and another from (x,0,2z) to (x,y,2z). These are the same length.

These project in the image to a line from (fx/z,0) to (fx/z, fy/z) and from (fx/z,0) to (fx/2z, fy/2z), where we can rewrite the last point as: (1/2)(fx/z,fy/z). The second line is half as long as the first.

Parallel lines meet

Common to draw image plane *in front* of the focal point.
Moving the image plane merely scales the image.



(Forsyth & Ponce)

Vanishing points

- Each set of parallel lines meets at a different point
  – The *vanishing point* for this direction
- Sets of parallel lines on the same plane lead to *collinear* vanishing points.
  – The line is called the *horizon* for that plane

For example, let's consider a line on the floor. We describe the floor with an equation like: y = -1. A line on the floor is the intersection of that equation with x = az + b. Or, we can describe a line on the floor as: (a, -1, b) + t(c, 0, d) (Why is this correct, and why does it have more parameters than the first way?)

As a line gets far away, z -> infinity. If (x,-1,z) is a point on this line, its image is f(x/z,-1/z). As z -> infinity, -1/z -> 0. What about x/z? x/z = (az+b)/z = a + b/z -> a. So a point on the line appears at: (a,0). Notice this only depends on the slope of the line x = az + b, not on b. So two lines with the same slope have images that meet at the same point, (a,0), which is on the horizon.

Properties of Projection

- Points project to points
- Lines project to lines
- Planes project to the whole image
- Angles are not preserved
- Degenerate cases
  - Line through focal point projects to a point.
  - Plane through focal point projects to line
  - Plane perpendicular to image plane projects to part of the image (with horizon).

Projective Transformation

- Represent projection with nice math:
  - Linear
    - Represent motion with matrices
    - Represent projection by a trick.
  - Group
    - Must compose projections. Dimension must be fixed. Get a superset of perspective views.
    - Invertible. No degenerate projections.
  - One-to-one
    - Points behind camera get projected.
    - Points at infinity

2D motion with matrices

$$\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ & & & \end{pmatrix}\begin{pmatrix} x \\ y \\ 0 \\ 1 \end{pmatrix} + \text{Perspective Projection} \equiv \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

So, 3D rotation, translation of a *planar* set of points can be represented as multiplication by a 3x3 matrix.

Note that scaling the matrix changes nothing, so we can assume WLOG that the entry in the last row and column is always 1.

Homogenous Coordinates

- Instead of saying (x,y,z) -> (x/z, y/z)
- We say that (x,y,z) == k(x,y,z) for any k.
- This is like just always putting off division.
- It means that motion and projection of planar objects can be represented with matrix operations

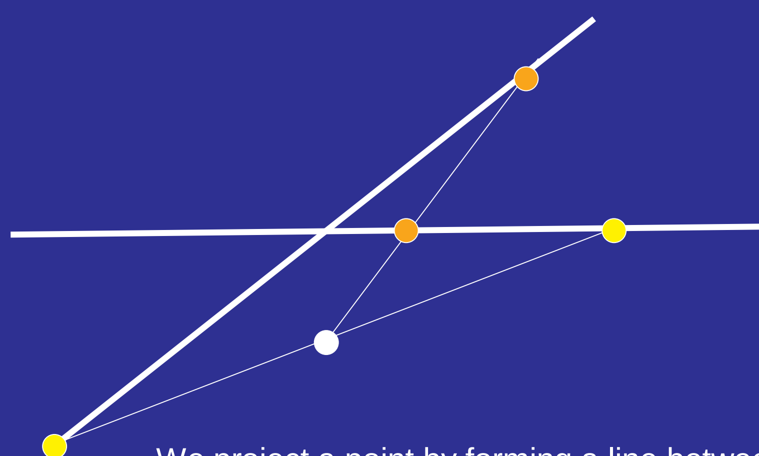
Projection as a Group

$$\begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- For this to be a rotation, the first two columns must be orthogonal, and of the same length.
- If we ignore this constraint, we get linear transformations, and a superset of perspective views of the object.
- We call these *Projective* transformations.
- It turns out these are the views we can get if we view images of the object. Projective views form a group, perspective ones don't.
- Note that scaling the matrix changes nothing, so we can assume WLOG that the entry in the last row and column is always 1.
- Note also that this matrix generically has an inverse.

Projection as a one-to-one transformation

- With perspective, some points are behind the camera, and so not visible.
  - This means taking a picture is not a one-to-one mapping of the world to an image.
  - With projective transformations, points behind the camera also project.



We project a point by forming a line between it and the focal point, and mapping the point to the intersection of this line and the w=1 line (for space x-y-w). This works for points in front or behind the "camera".

The point at infinity

- Points on the horizon are at infinity on the original plane, but not on the new plane.
- These are projections of points at (kx, ky, 1) for k -> infinity.
- We can write these as (x,y,0).

To understand this, let's think about transforming the z=0 plane to another one, going through it first without using homogenous coordinates. Let's consider taking this plane that is parallel to the image plane and transforming it so that it is slanting diagonally, with a surface normal of (0,1,1)/sqrt(2). We can consider the original plane to be the ordinary x-y plane, with coordinates written as (x,y,0), for all (x,y). To map this to a slanted plane, we can imagine rotating the plane around the x axis and translating it 1 in the z direction (why do weird things happen if we don't also translate it?) with a matrix like:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Let's consider what happens to a point that is very far away from us, such as a point that starts out at (0,k,0), for a very large value of k. When we multiply this point by our matrix, we get: (0, k*sqrt(2)/2, 1+k*sqrt(2)). If we then apply perspective projection, we get the point (0, ksqrt(2)/2 / (1+ksqrt(2)/2)). The y coordinate is slightly less than 1, and approaches 1 as k approaches infinity. This is the vanishing point of the line x = 0. Similarly, we can see that the whole horizon is y = 1. Now let's do this with homogenous coordinates. We can represent this transformation with a matrix that ignores the third column and 4th row of this, obtaining the projective transformation matrix: :

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2} & 0 \\ 0 & \frac{\sqrt{2}}{2} & 1 \end{pmatrix}$$

Our original plane isn't represented as a plane in 3D, but as a plane in 2D with homogenous coordinates. So a point very high up has coordinates (0,k,1) for a large value of k. Transforming this point, we get the same thing as before. No, notice that the point (0,k,1) is the same as the point (0,1, 1/k). As k gets big, this approaches the point (0,1,0). We call this a point at infinity. Transforming it produces (0, sqrt(2)/2, sqrt(2)/2), which is the same point as (0,1,1). So the point at infinity is mapped to a point that is not at infinity.

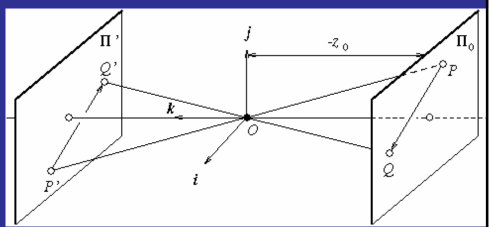
This does have some weird side effects. It means that points that are infinitely far away in opposite directions can be mapped to be nearby, since they are both near the line at infinity. In the last example, (0,1000,1) will be mapped to (0, 1000sqrt(2)/2, 1000sqrt(2)/2 + 1), and the point (0, -1000,1) will be mapped to -1000sqrt(2)/2, -1000sqrt(2)/2 + 1). These are almost the same point, very close to (0,1,1).

3D Projective Transformations

- We can do the same thing with 3D.
- Matrix is 4x4
- Points are 4D vectors.

Weak perspective (scaled orthographic projection)

- Issue
  - perspective effects, but not over the scale of individual objects
  - collect points into a group at about the same depth, then divide each point by the depth of its group



(Forsyth & Ponce)

The Equation of Weak Perspective

$$(x, y, z) \rightarrow s(x, y)$$

- *s* is constant for all points.
- Parallel lines no longer converge, they remain parallel.

Parallel Projection

Project on the plane, *z = 0*

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

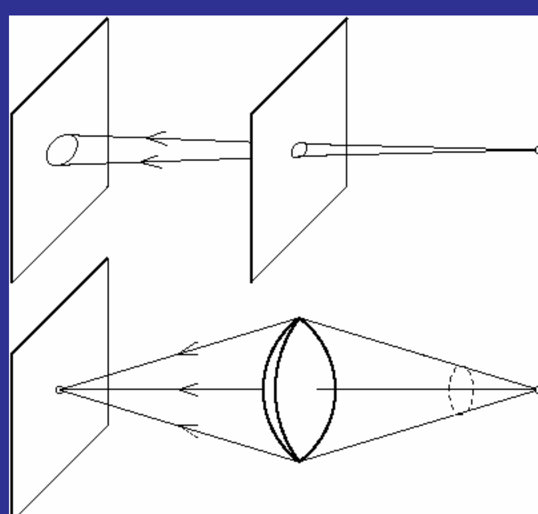
Affine transformations as a subset of Projective

- These can be represented with:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 0 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & t_z \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- This transforms points, but then projects them orthographically so z coordinates are all 1 (matrix can also represent scaling).
- Note that any point at infinity, (x,y,0), is transformed to another point at infinity (x',y',0). This implies parallel lines stay parallel.

Cameras with Lenses



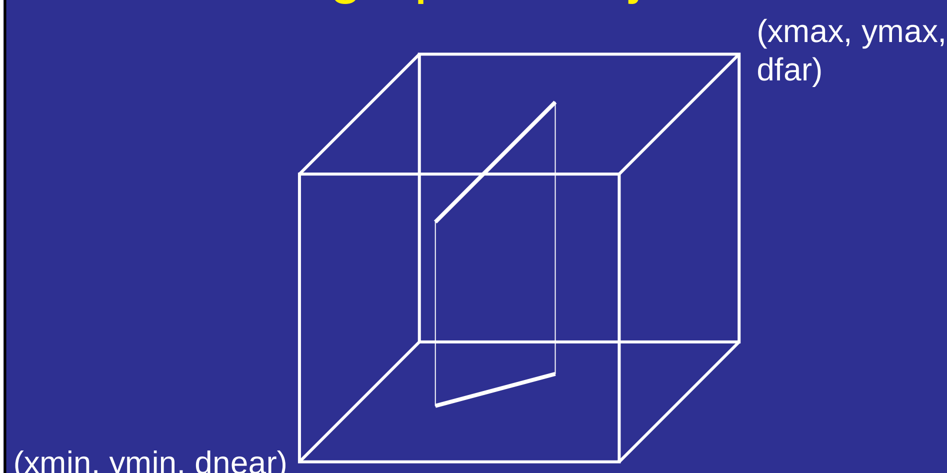
(Forsyth & Ponce)

OpenGL for Projection

- Projection specifies clipping windows.

- glMatrixMode(GL_PROJECTION)
- glOrtho(xmin,xmax,ymin,ymax,dnear,dfar)
- gluPerspective(theta,aspect,dnear,dfar)

Orthographic Projection



Orthographic Projection

- Instead of figuring out how to do projection for a given set of parameters, we transform the world to a canonical form, then always do projection the same way.
- Xmin, Ymin, Zmin -> -1, Xmax, Ymax, Zmax -> 1
- This can be done with a linear transformation:

$$\begin{pmatrix} \frac{2}{x_{\max} - x_{\min}} & 0 & 0 & 0 \\ 0 & \frac{2}{y_{\max} - y_{\min}} & 0 & 0 \\ 0 & 0 & \frac{2}{z_{\max} - z_{\min}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -\frac{x_{\max} + x_{\min}}{2} \\ 0 & 1 & 0 & -\frac{y_{\max} + y_{\min}}{2} \\ 0 & 0 & 1 & -\frac{z_{\max} + z_{\min}}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

To see why this is, first notice that the right-most matrix translates the scene so that the center of the visible area is at (0,0,0).

Next notice that $\begin{pmatrix} \frac{2}{x_{\max} - x_{\min}} & 0 & 0 & 0 \\ 0 & \frac{2}{y_{\max} - y_{\min}} & 0 & 0 \\ 0 & 0 & \frac{2}{z_{\max} - z_{\min}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ scales the x, y and z axes so that each side of the view volume has a length of 2.

The whole orthographic projection pipeline

- Transform the world so that the viewer is the origin and the viewing direction is the z axis (using matrix created by gluLookAt).
- Transform the viewing volume so that it is the cube with corners (-1,-1,-1) and (1,1,1).
- Project by removing the z coordinate.
- Points with smallest z value are closest to the camera (more on this later).

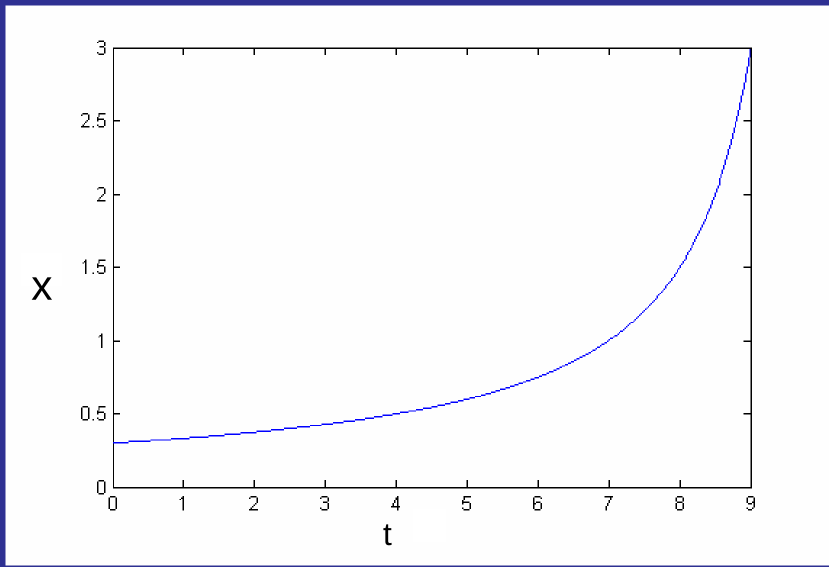
Perspective Review

Just to review orthographic and perspective projection, let's go over a practice problem (this was a question I got asked on my Phd qualifying exam). Suppose we are viewing a scene from the origin, with the z=1 plane as the image plane. Suppose there is a small ball in the scene at (3,0,10), and it is moving in the negative z direction at speed 1 unit per second, so that after 1 second it's at (3,0,9). What are the image coordinates of this ball, as a function of time.

First we'll write the world coordinates as a function of time. We can say (x,y,z) = (3,0,10) + t(0,0,-1). Or we could just say x = 3, y = 0, z = 10 – t.

Next, let's consider what happens with orthographic projection. This just involves removing the z coordinate, so we always get position (3,0), until the ball passes behind us after 9 seconds. In fact, it doesn't really make sense to talk about the image plane with orthographic projection, because points would project to the same place on any plane that is parallel to z = 1. Rather, we could just set dnear =1, and then after 9 seconds the ball would disappear.

What about with perspective projection. At time t, the image coordinates will be (3/(10-t), 0/(10-t)). Or we can say, y = 0, and x = 3/(10-t) => 10x – xt – 3 = 0. We graph this on the next slide. This makes sense. First of all, as the point is far away it is more in the center of the image (what is the vanishing point of the lines parallel to the z axis). As it gets closer, it moves to the side, and the rate and which this happens gets faster and faster. This is our ordinary perception of the world. Think about standing on a train platform. When a train is far away, it hardly seems to be moving; as it goes by the station it seems to zoom by.



Perspective Viewing Volume

- The trick is to transform the viewing volume so that a line through the focal point becomes a line in the z direction.

For example: Suppose the focal point is at (0,0,0), and we can see everything between z = 2 and z = 1. If we apply a transformation that takes (0,0,0,1) to (0,0, - 1,0), leaves the z = 2 plane fixed and leaves z = 1 values at 1.

$$\begin{pmatrix} 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 3/2 & -1 \\ 0 & 0 & 1/2 & 0 \end{pmatrix}$$

After applying this matrix, we can then perform orthographic projection. To believe this, you have to believe that (x,y,2,1) winds up at with the same (x,y) coordinates as (x/2, y/2, 1, 1) and that a linear transformation maps a line to a line.

Another way to look at this is to consider the point (x,y,z). With perspective, this projects to (x/z, y/z). If we apply the matrix we defined to (x,y,z,1) we wind up with it at (x/2, y/2, 3z/2 – 1, z/2), which is the same point as

(x/z,y/z,3-2/z, 1). So then we can take these transformed points and project them orthographically, producing the same effect as if we'd projected the origanal points with perspective. This is convenient, because we can build hardware to perform orthographic projection.

This transformation is very much like the one used to represent perspective projection, except that that transformation results in all points on the same line corresponding to an identical point, with the same z value, while this transformation retains information about which points are closer to the viewer.

Perspective Projection after this mapping

$$\begin{bmatrix} x'' \\ y'' \\ z'' \\ w'' \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 0 \end{pmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Homogenize (divide by $w'' = z / f$) to get:
  - $x' = x / (z / f) = x'' / w''$
  - $y' = y / (z / f) = y'' / w''$
  - $z' = z / (z / f) = z'' / w'' = f$