7. 3D Transformations

Reading

Required:

- Hearn and Baker, Sections 11.1–11.4, 12.1–12.5

Supplemental:

- Foley *et al.*, Chapter 5.6 and Chapter 6
- David F. Rogers and J. Alan Adams, *Mathematical Elements for Computer Graphics*, Second edition, McGraw-Hill, New York, 1990, Chapter 3.

The pinhole camera

The first camera ever made was the “camera obscura”: a dark room or chamber with a hole in the window shade.

The first camera that actually recorded an image simply replaced the back wall with material sensitive to light, i.e., film.

This kind of camera is called a “pinhole camera.”

Q: What does the pinhole camera have in common with a conventional camera or the human eye?

Q: How is it different?

The pinhole camera (cont'd)

Today, we typically use the pinhole camera for creating images of synthetic scenes.

However, since we are not bound by the laws of physics, we conveniently place the image plane in front of the pinhole.

Q: Why is this convenient?

Rendering a 3D scene

Q: What do we need to create an image of a 3D scene?

3D Drawing

What should "3D MacDraw" look like?

- User interactively creates a number of 3D primitives
- User can scale, translate, and rotate objects, as well as group them together.
- User specifies a viewpoint, image plane, and which portions of space are considered too far or too close to be seen.
- User specifies the light sources (brightness, position, orientation).
- User specifices surface properties (color, shininess, texture).
- Normals can be specified, though they may be implicit in the geometry of the primitives.
- User *does not* specify the ordering of objects.
- The program draws shaded, perspective correct, occlusion correct image on the screen.

3D drawing (cont'd)

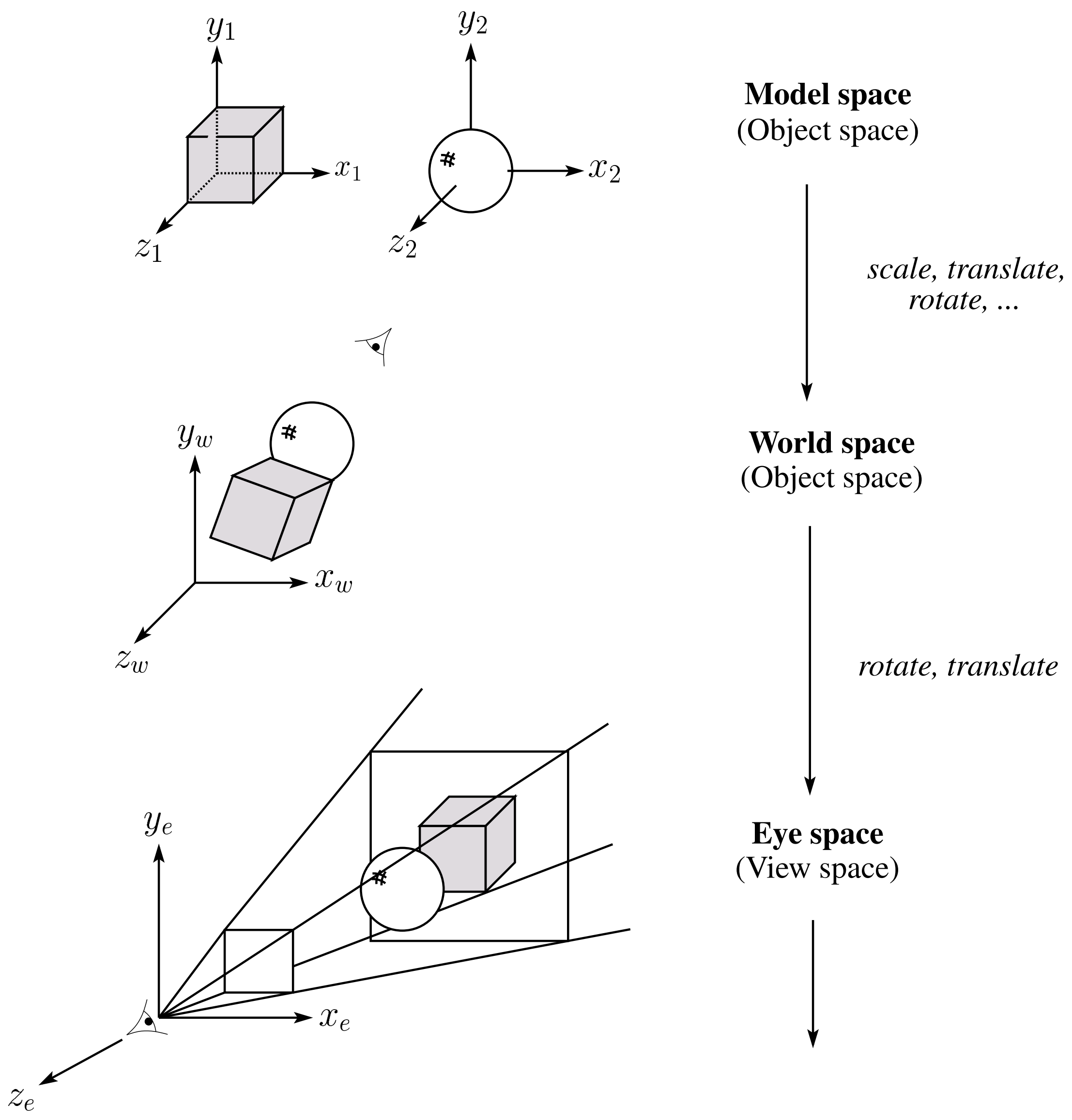
What are some of the key ingredients needed to make this work?

- A sequence of transformations, some of them stored in hierarchies corresponding to groups of primitives.
- Viewing specification.
- Lighting specification
- Perspective mapping to the screen.
- Software or hardware to handle:
  - Shading
  - Perspective
  - Occlusions
  - Rasterization

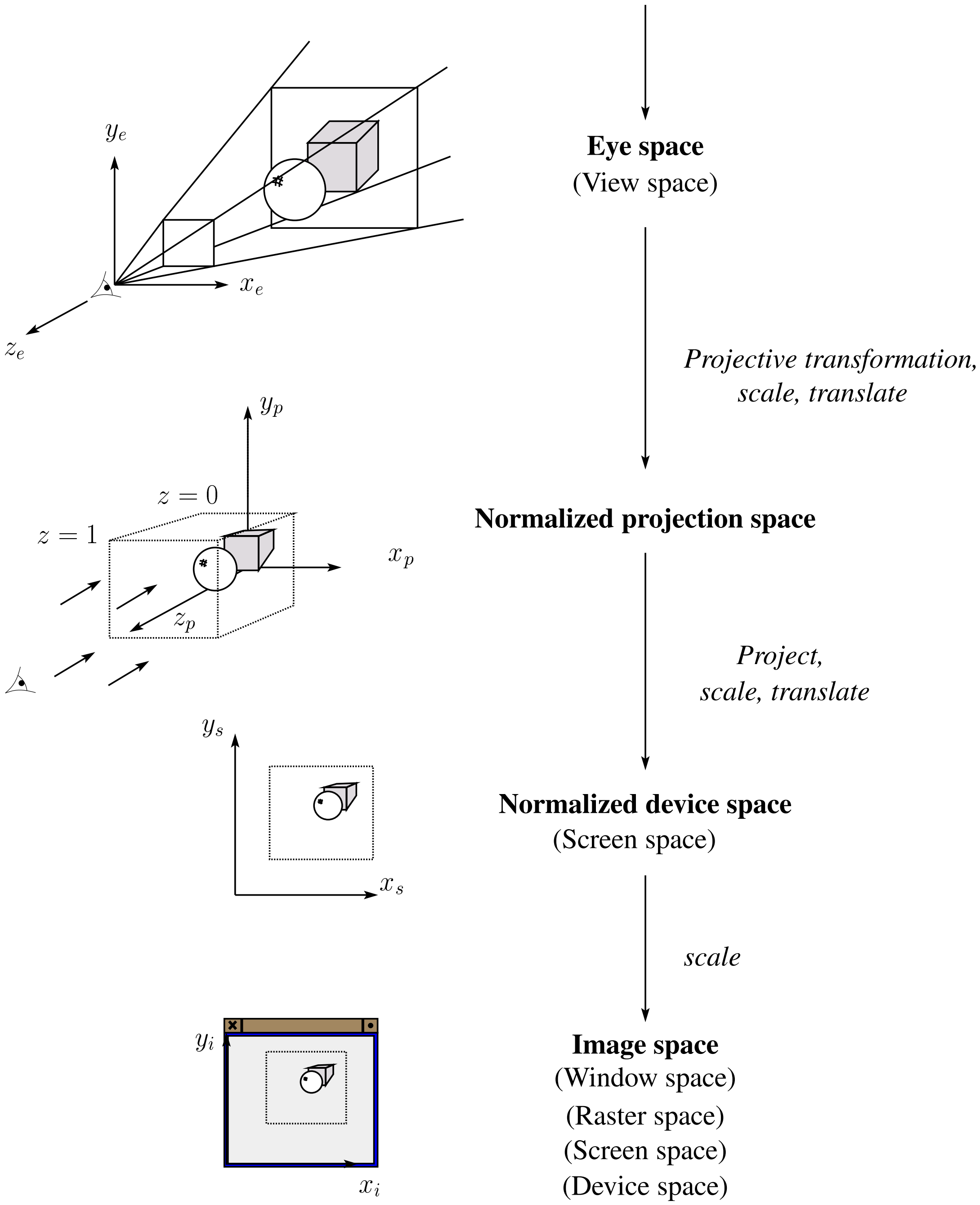
3D graphics API's support all of these features.

3D Geometry Pipeline

Let's think about this in terms of a set of coordinate systems:



3D Geometry Pipeline (cont'd)

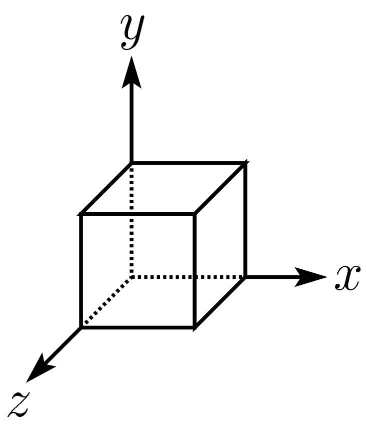


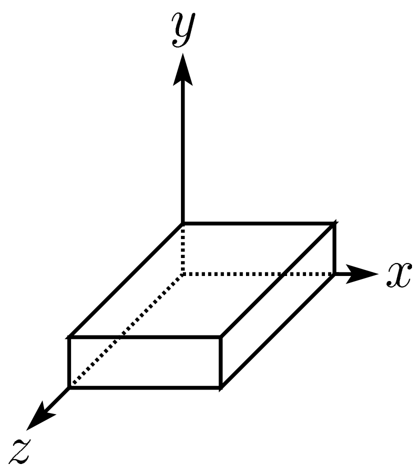
Basic 3-D transformations: scaling

Some of the 3-D transformations are just like the 2-D ones. For example, scaling:

Scaling

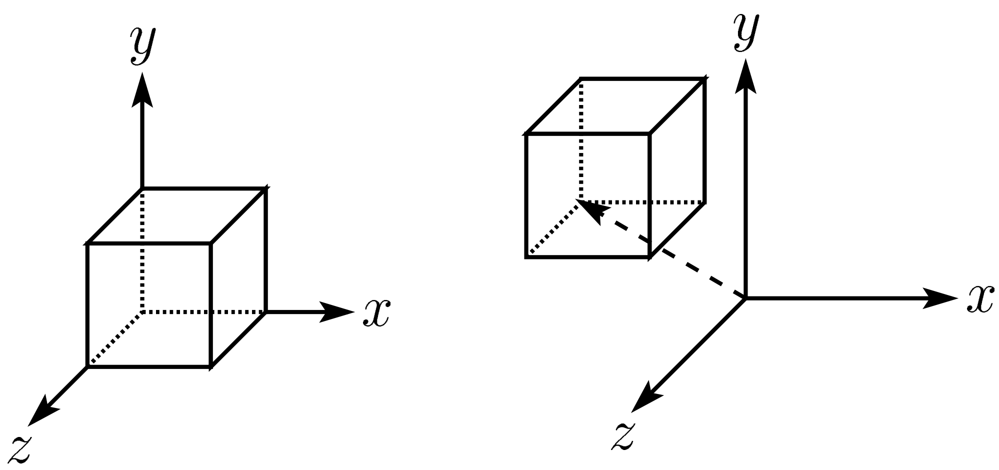
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$





Translation in 3D

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



Rotation in 3D

Rotation now has more possibilities in 3D:

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

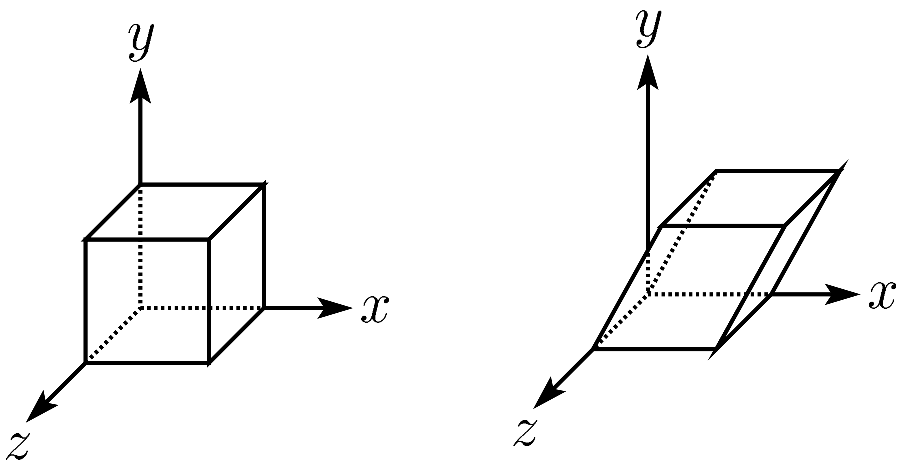
$$R_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Shearing in 3D

Shearing is also more complicated. Here is one example:

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



Projections

"Projections" transform points in n-space to m-space, where $m < n$.

In 3-D, we map points from 3-space to the "projection plane" (PP) along "projectors" emanating from the "center of projection" (COP):

The center of projection is exactly the same as the pinhole in a pinhole camera.

There are two basic types of projections:

- "Perspective" — distance from COP to PP finite
- "Parallel" — distance from COP to PP infinite

Parallel projections

For parallel projections, we specify a "direction of projection" (DOP) instead of a COP.

There are two types of parallel projections:

- "Orthographic projection" — DOP perpendicular to PP
- "Oblique projection" — DOP not perpendicular to PP

We can write orthographic projection onto the $z = 0$ plane with a simple matrix.

Normally, we do not zero out or drop the z value right away. Why not?

Oblique parallel projections

There are two standard kinds of oblique projections:

- "Cavalier projection"
  - DOP makes 45° angle with PP
  - Does not foreshorten lines perpendicular to PP
- "Cabinet projection"
  - DOP makes 63.4° angle with PP
  - Foreshortens lines perpendicular to PP by one-half

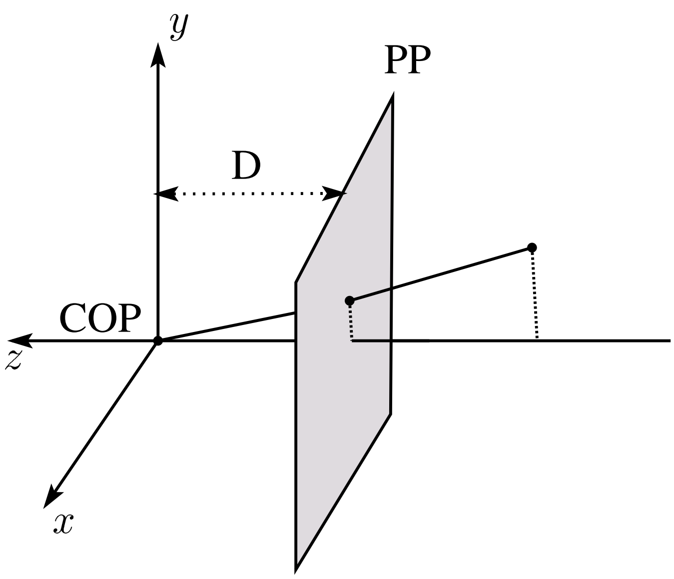
Properties of parallel projection

Properties of parallel projection:

- Not realistic looking
- Good for exact measurements
- Are actually a kind of affine transformation
  - Parallel lines remain parallel
  - Angles not (in general) preserved
- Most often used in CAD, architectural drawings, etc., where taking exact measurement is important

Derivation of perspective projection

Consider the projection of a point onto the projection plane:



By similar triangles, we can compute how much the x and y coordinates are scaled:

Homogeneous coordinates revisited

Remember how we said that affine transformations work with the last coordinate always set to one.

What happens if the coordinate is not one?

We divide all the coordinates by w:

If $w = 1$, then nothing changes.

Sometimes we call this division step the "perspective divide". Why?

Homogeneous coordinates and perspective projection

Now we can re-write the perspective projection as a matrix equation:

After division by w, we get:

Projection implies dropping the z coordinate to give a 2D image, but we usually keep it around a little while longer. Why?

Projective normalization

After applying the perspective transformation and dividing by w, we are free to do a simple parallel projection (drop the z) to get the 2D image.

What does this imply about the shape of things after the perspective transformation + divide?

Vanishing points

What happens to two parallel lines that are not parallel to the projection plane?

Think of train tracks receding into the horizon...

The equation for a line is:

After perspective transformation we get:

Vanishing points (cont'd)

Dividing by w:

Letting t go to infinity:

We get a point!

The point does not depend on the the choice of $\mathbf{p}$, so any line in the direction $\mathbf{v}$ will go to the same point.

Therefore, all parallel lines intersect at a point on the PP. This point is called the "vanishing point".

Q: How many vanishing points are there?

Principal vanishing points

If we define a set of "principal axes" in world coordinates, i.e., the x_w, y_w, and z_w axes, then it's possible to choose the viewpoint such that these axes will converge to different vanishing points.

The vanishing points of the principal axes are called the "principal vanishing points". Example:

Perspective drawings are often classified by the number of principal vanishing points.

- One-point perspective — simplest to draw
- Two-point perspective — gives better impression of depth
- Three-point perspective — most difficult to draw

All three types are equally simple with computer graphics.

Properties of perspective projections

The perspective projection is an example of a "projective transformation."

Here are some properties of projective transformations:

- Lines map to lines
- Parallel lines do <u>not</u> necessarily remain parallel
- Ratios are <u>not</u> preserved

One of the advantages of perspective projection is that size varies inversely with distance — looks realistic.

A disadvantage is that we can't judge distances as exactly as we can with parallel projections.

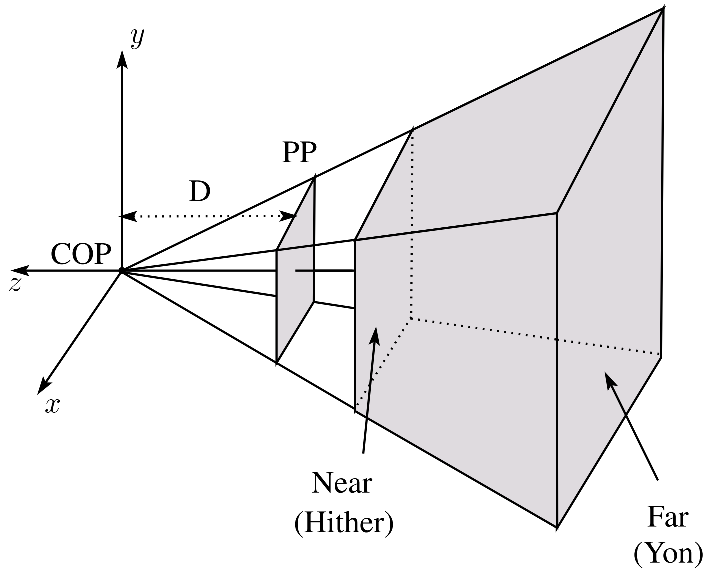
Q: Why did nature give us eyes that perform perspective projections?

Q: Do our eyes "see in 3D"?

Clipping and the viewing frustum

The center of projection and the portion of the projection plane that map to the final image form an infinite pyramid. The sides of the pyramid are “clipping planes”.

Frequently, additional clipping planes are inserted to restrict the range of depths. These clipping planes are called the “near” and “far” or the “hither” and “yon” clipping planes.



All of the clipping planes bound the the “viewing frustum”.

Summary

What to take away from this lecture:

- All the underlined names and names in quotations.
- How 2D affine transformations generalize into 3D
- How we use homogeneous coordinates to represent perspective projections.
- The mathematical properties of projective transformations.
- The classification of different types of projections.
- The concepts of vanishing points and one-, two-, and three-point perspective.
- An appreciation for the various coordinate systems used in computer graphics.
- How the persepctive transformation works.