Viewing in 3D

CSC 292:
Computer Graphics

Randal C. Nelson

Basic 3D Viewing Process

- Must get from 3D world to 2D display.
- First limit the 3D volume to specific *view volume*.
- Then go to a plane via *projection*.
- Finally, window on the plane.

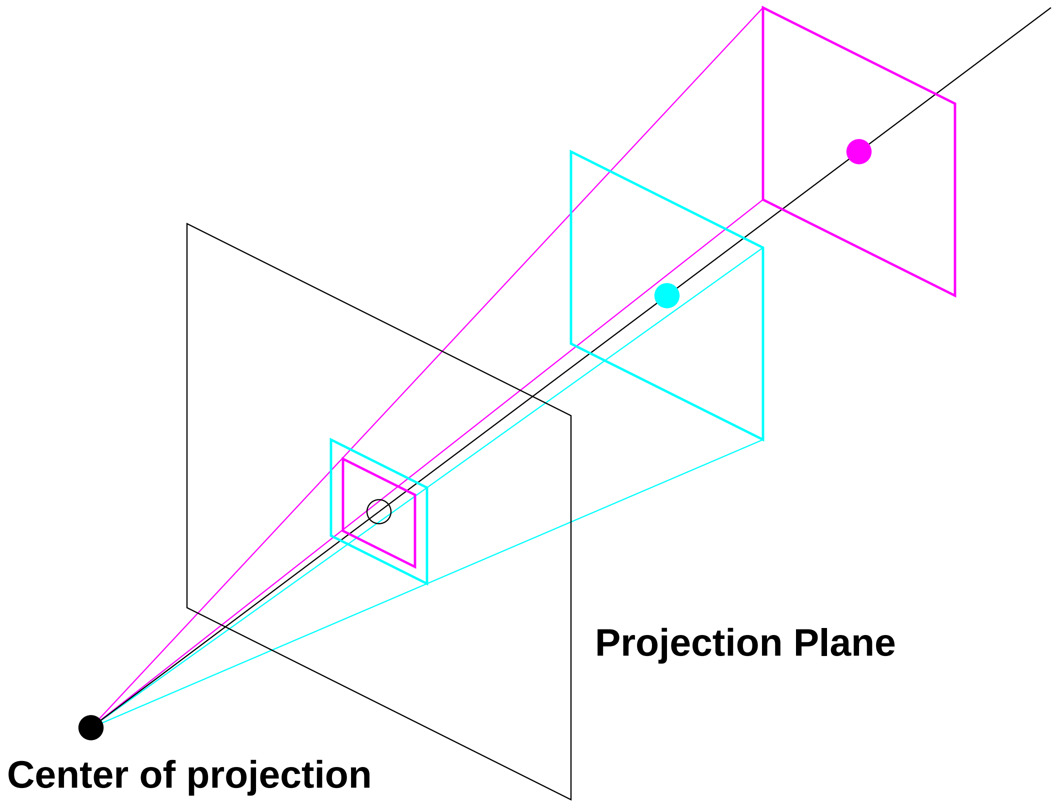
Projections

- A *projection* reduces the dimensionality of a space (e.g. from 3 to 2).
- Various sorts have been defined.
- We consider only *planar geometric* projections, (onto a plane, using straight lines that all are parallel or pass through a point).
- Could project to non-planar surface (e.g. a sphere).
- Could use some other projection curve or arrangement of lines (e.g. some map projections).

Planar Geometric Projections

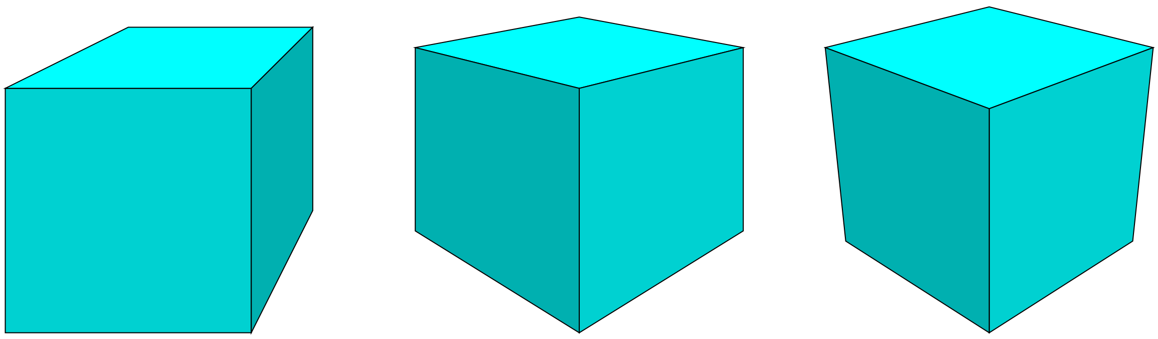
- Map 3-space onto a plane.
- Composed of two components:
  - A *projection plane*.
  - A set of straight rays called *projectors*.
- The projection maps all the points on each ray to the point where that ray intersects the projection plane.
- Two basic classes:
  - *Perspective*: The projectors all pass through a point.
  - *Parallel*: The projectors are parallel lines.

Perspective Projection



- Determined by the *center of projection* - the point through which all the projectors pass.
- Similar to cameras and human vision.
- Looks realistic.

- Can't make direct measurements of lengths or angles.
- Objects look smaller when farther away.
- Projected parallel lines intersect at a *vanishing point* (unless parallel to projection plane).
- The *axis vanishing points* are where lines parallel to x, y, and z axes appear to converge.
- A drawing is said to utilize 1, 2, or 3 point perspective, depending on how many of the principle axes intersect the projection plane.

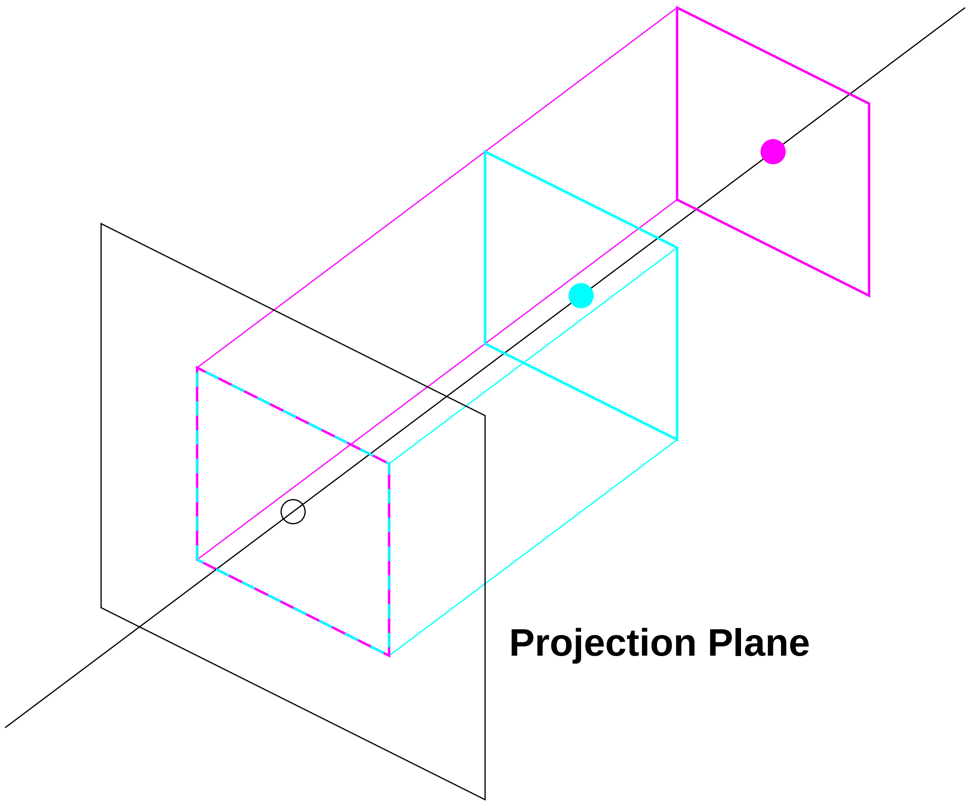


One point **Two Point** **Three point**

Parallel Projection

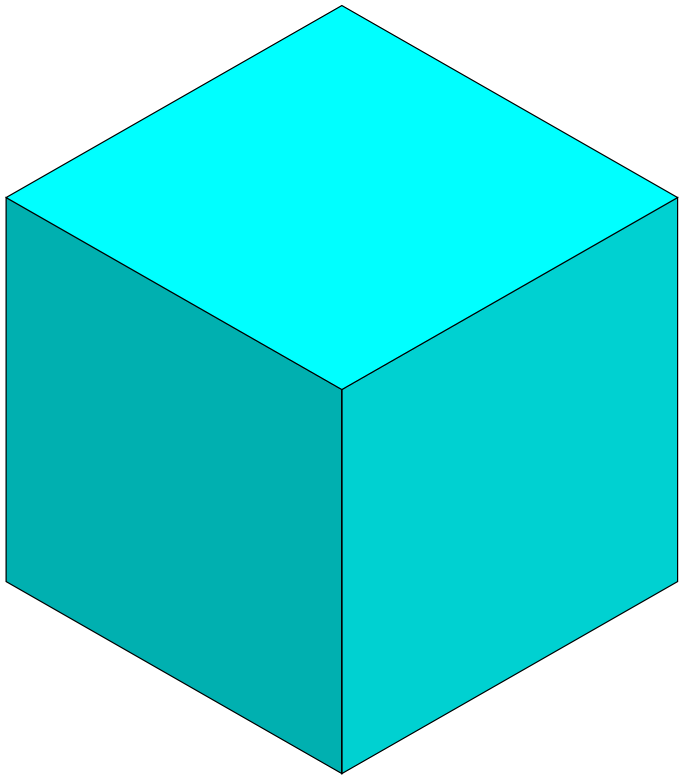
- Determined by a *direction of projection*.
- Not as realistic as perspective, but certain measurements of distances and angles can be made.
- Two main classes: *orthographic* and *oblique*.
- And further subclasses.

Orthographic projection

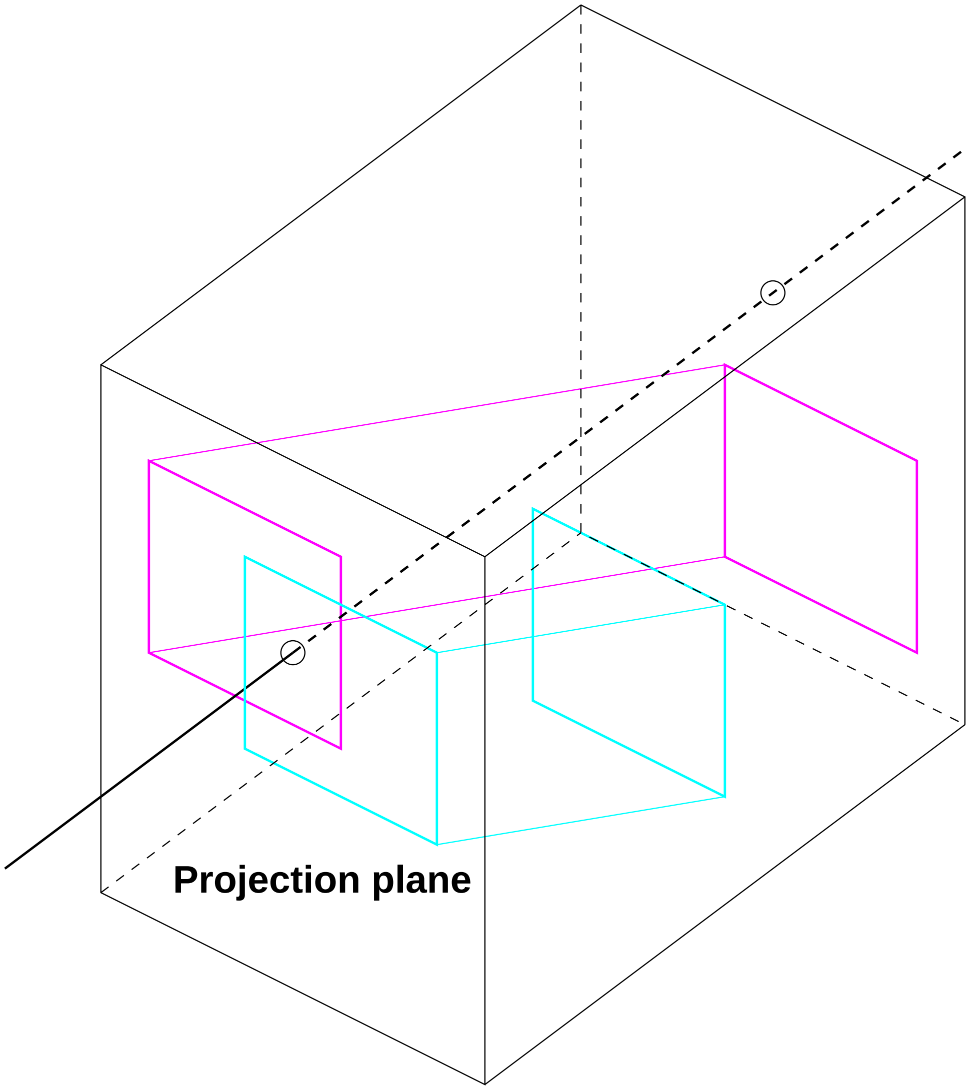


- Projectors are perpendicular to projection plane.
- Front, top, side views: Named faces are parallel to projection plane.
- Can measure distances, angles on named faces.

- *Axonometric*: All axes intersect plane at some angle, relative distances on axes can be measured with appropriate scale factor.
- *Isometric*: All three axes intersect projection plane at equal angles. Scale factors for axes are equal

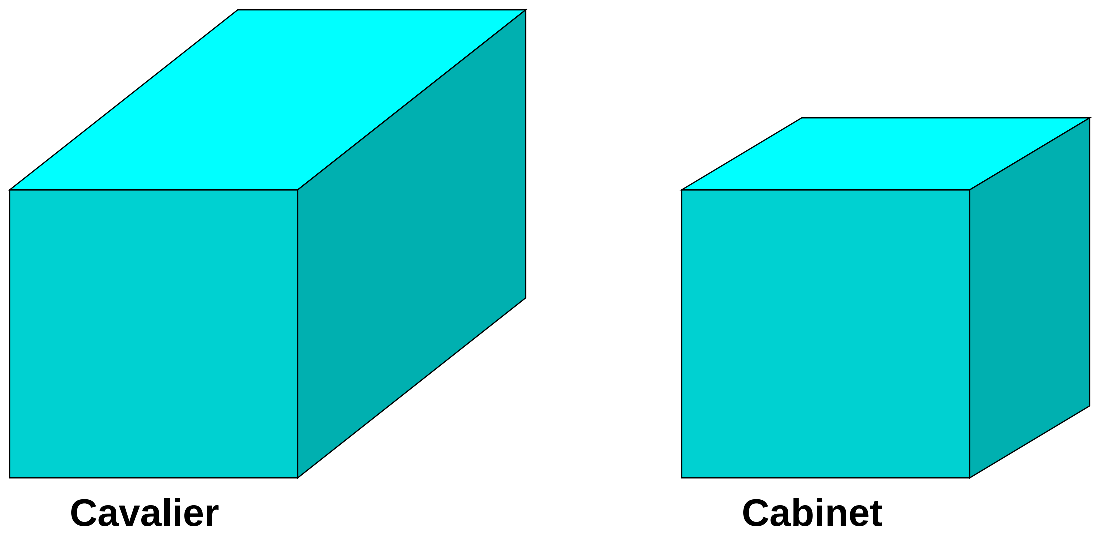


Oblique projection

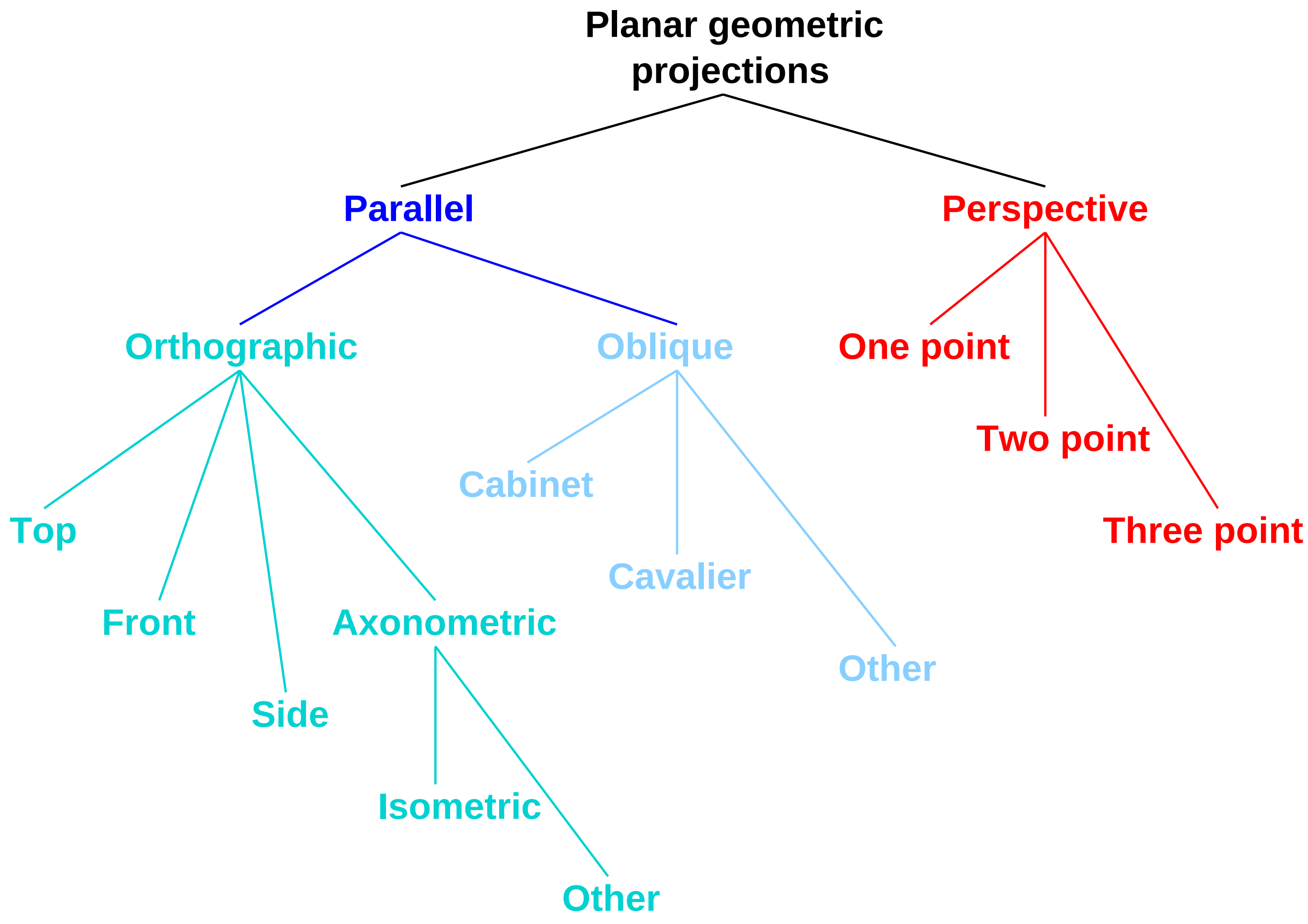


- Projectors form some angle other than 90 degrees with projection plane.
- Permits both an accurate face, and 3D structure to be displayed.

- Cavalier: (45 degree projection, equal scales for z, x, and y). Tends to look a bit stretched
- Cabinet: 63.4 (arctan2) projection, factor of 2 foreshortening of z relative to x and y. Looks more realistic.



Planar Projection Hierarchy



Specifying Views

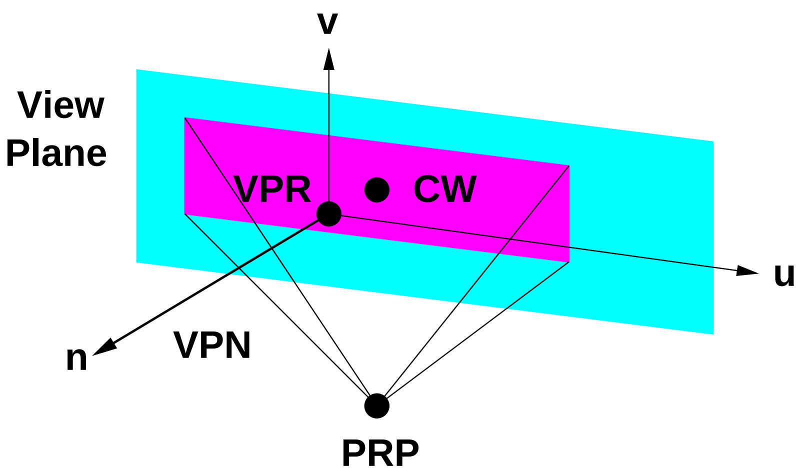
Problem: Given a 3D world, described in a 3D world coordinate system, how do we specify a view of this world?

The following terminology and development is from PHIGS:

- First, we specify the projection plane, now called the *view plane*, as follows.
  1. A point on the plane, called the *view reference point* or VRP.
  2. A (unit vector) normal to the plane called the *view plane normal* or VPN.

- Next we need a window on the view plane.
- To get this, we establish a local 3D coordinate system, called the *viewing-reference coordinate* system (VRC), keyed to the view plane as follows:
  - The origin is the VRP (view reference point).
  - One axis, the *n-axis* is the VPN (view plane normal).
  - A second axis, the *v-axis* is determined by projection of the *view up vector* or VUP onto the view plane.
  - The *u-axis* completes a right-handed coordinate system with the other two.
- We can now define the desired window, not necessarily centered on the VRP.

- The projection is defined by the *projection reference point* or PRP, and an indication of projection type (perspective or parallel).
- The PRP is defined in the local (VRC) system, so it moves with the viewing plane.
- If the projection is perspective, the PRP is the center of projection.
- If the projection is parallel, the direction of projection (DOP) is defined by the vector connecting the projection reference point (PRP) to the center of the window (CW).



- The *view volume* bounds the portion of the world that is projected on the view plane.
- For perspective projection the volume is a semi-infinite pyramid with apex at the PRP.
- For parallel projection it is an infinite parallelopiped.
- We can limit the view volume by truncating it with two planes parallel to the view plane: the *front clipping plane* and the *back clipping plane*.

- The view volume is displayed by first transforming it into a rectangular solid in *normalized projection coordinates* or NPC.
- This basically stretches the (truncated) view volume so that all the projection rays are parallel to the local z axis and run from the back to the front clipping planes.
- Display of a (wireframe) is then simply a matter of throwing out the z coordinate.
- Hidden surface removal involves looking at the z coordinate and displaying whatever is closer.

Thinking with a Camera

The viewing parameters can be related to a "virtual" camera position as follows:

- Pick a *camera position*. This corresponds to the center of projection.
- Pick a *center of attention*. The vector connecting this to the camera position can be used as the plane normal (VPN). (Can't really do oblique views with this model)
- Pick the "up" direction (VUP).
- Pick a distance D from the center of projection to the viewing plane. This basically determines the magnification or zoom of the camera.
- Pick the height and width of the window, which is assumed to be centered on the center of attention.

Viewing Specification

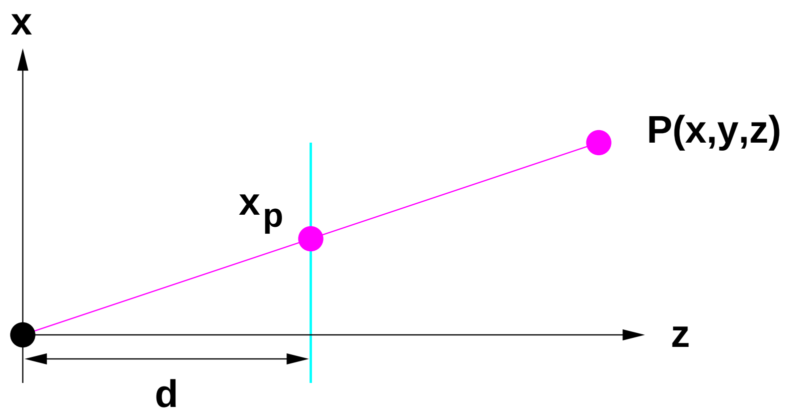
A viewing specification can be given by two 4×4 matrices.

- A *view orientation matrix* that combines the VRP, the VPN, and the VUP (reference point, normal, and local up direction) to provide a change of coordinate system from the world to the local VRC system.
- A *view mapping matrix* that combines the projection point and the local window to transform points in the VRC to points in normalized projection coordinates.

The Mathematics of Projection

Basic strategy: Start with simple case, and generalize by applying already known transformations

- Assume projection plane is normal to z axis, at distance d, and the projection point is at the origin.



- By similar triangles, we can relate the projection (x_p, y_p) of the point (x, y, z).

$$\frac{x_p}{d} = \frac{x}{z}; \frac{y_p}{d} = \frac{y}{z};$$

yielding

$$x_p = \frac{x}{z/d}; y_p = \frac{y}{z/d}$$

- Note how increasing z causes the projected values to decrease.

The transformation can be represented as a 4×4 matrix:

$$M_{per} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

If we multiply the homogeneous point $P = (x, y, z, 1)$ by this matrix to get the projected point $P_p = (x_p, y_p, z_p, W)$ we obtain:

$$P_p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix} = \begin{bmatrix} x/(z/d) \\ y/(z/d) \\ d \\ 1 \end{bmatrix}$$

As expected. Note the need to *homogenize* the result.

An alternative formulation places the projection plane at $z = 0$ and the projection point at $z = -d$.

In this case similar triangles gives us

$$\frac{x_p}{d} = \frac{x}{(z+d)}; \frac{y_p}{d} = \frac{y}{(z+d)};$$

yielding

$$x_p = \frac{x}{z/d+1}; y_p = \frac{y}{z/d+1}$$

and the matrix

$$M'_{per} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix}$$

Orthographic Projection

In the previous case, we can let d tend to infinity and obtain a matrix for orthographic parallel projection:

$$M_{ort} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Note that this essentially throws away the z coordinate.

Implementing Planar Projections

Problem: Given a view volume and a projection, how do we implement the actual clipping and projection of (say) line segments?

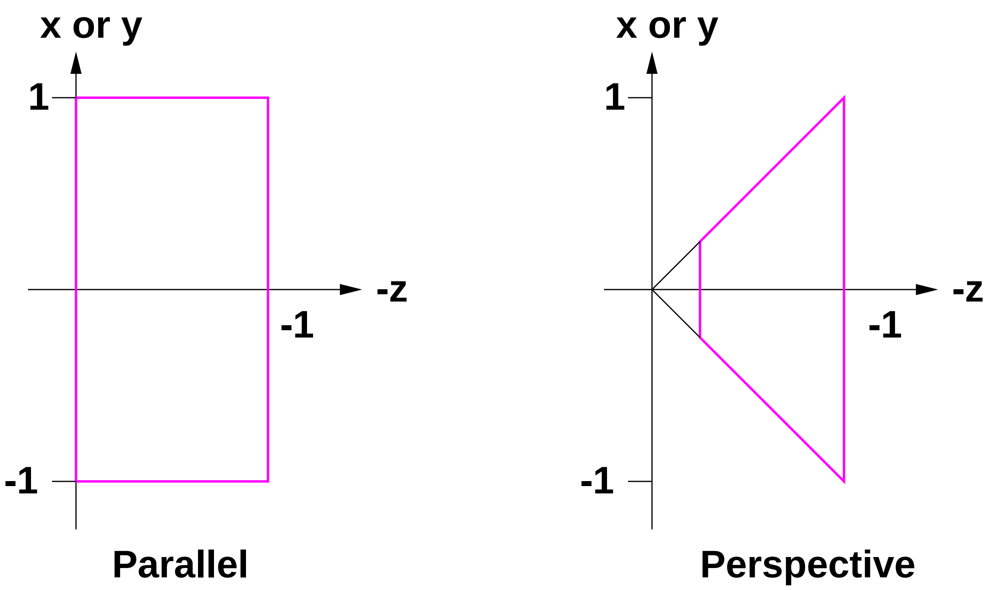
- Could brute-force compute intersections of lines with all 6 clipping planes and project the results.
- Alternatively, note that clipping against certain *canonical view volumes* is easy, transform space to match canonical volume, and clip and project with only a few operations.

Parallel canonical volume is bounded by the six planes:

$$x = -1, x = 1, y = -1, y = 1, z = 0, z = -1.$$

Perspective canonical volume is bounded by the six planes:

$$x = z, x = -z, y = z, y = -z, z = -x_{min}, z = -1.$$



3D Viewing Pipeline

Overall sequence of operations to obtain 3D view:

1. Apply normalizing transformation.
2. Clip against canonical view volume.
3. Project onto projection plane.
4. Transform into viewport in 2D device coordinates.

Normalizing Parallel Projection

The normalizing transformation N_{par} for parallel projections can be obtained by composing primitive transformations that effect the following steps.

1. Translate the VRP to the origin.
2. Rotate the VRC such that the n axis (VPN) becomes the z axis, the v axis becomes the y axis, and the u axis becomes the x axis.
3. Shear so that the direction of projection becomes parallel to the z axis.
4. Translate and scale the viewing volume into the canonical parallel-projection volume.

1 and 2 define the *view-orientation matrix*, and 3 and 4 define the *view-mapping matrix* in PHIGS.

How To Do It

- Translation is just $T(-VRP)$
- To get the rotation, use orthonormal matrix properties: (the rows are the vectors that are rotated into the various axes)
  - R_z is just $VPN/\|VPN|$
  - R_x is $VUP \times R_z)/\|VUP \times R_z\|$
  - R_y is $R_z \times R_x$

To get at the shear, we solve the equation

$$DOP' = [0, 0, dop_z, 1]^T = SH_{par} \cdot DOP$$

where SH_{par} is the shear matrix

$$\begin{bmatrix} 1 & 0 & shx & 0 \\ 0 & 1 & shy & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

for shx and shy. This yields

$$shx = -\frac{dop_x}{dop_z}; shy = -\frac{dop_y}{dop_z}$$

Finally, we transform the front center of the view volume to the origin, and scale the size to the $2 \times 2 \times 1$ size of the final canonical view volume. The transformations are:

$$T_{par} = T\left(-\frac{u_{max} + u_{min}}{2}, -\frac{v_{max} + v_{min}}{2}, -F\right)$$

$$S_{par} = S\left(\frac{-2}{u_{max} + u_{min}}, \frac{-2}{v_{max} + v_{min}}, \frac{1}{F - B}\right)$$

where F and B are the front and back clipping distances.

The final normalizing transform is

$$N_{par} = S_{par} \cdot T_{par} \cdot SH_{par} \cdot R \cdot T(-VRP)$$

Normalizing Perspective Projection

The normalizing transformation N_{per} for perspective projections can be similarly obtained.

1. Translate the VRP to the origin.
2. Rotate the VRC such that the n axis (VPN) becomes the z axis, the v axis becomes the y axis, and the u axis becomes the x axis.
3. Translate so that the center of projection given by the PRP is at the origin.
4. Shear so that the centerline of the view volume becomes the z axis.
5. Translate and scale the viewing volume into the canonical parallel-projection volume.

Clipping

- Clipping against the canonical view volumes can be done using adaptations of the Cohen-Sutherland or the Cyrus-Beck algorithms.
- For the Cohen-Sutherland algorithm, a 6-bit outcode is used to determine trivial acceptance and rejection. The intersections can be calculated using the parametric representation for a line.
- For the Cyrus-Beck algorithm, we compute at most 6 values of the line parameter t, and classify them as potentially entering and potentially leaving exactly as before, and select the clip points.

- It is also possible to clip in homogeneous coordinates.
- First, we map the perspective pyramid into the parallel rectangle with an appropriate matrix.
- Then we substitute $x = X/W$, $y = Y/W$, and $z = Z/W$ into the bounding plane equations ($x > 0$ etc.
- This gives a 4-space pyramid that can be clipped against.
- Useful because some transforms generate homogeneous representations that do not map well to 3-space.