Lecture 4-5
3D Transformations and Projections

Xin (Shane) Li
Sep. 3, 2009

http://www.ece.lsu.edu/xinli/teaching/EE4700Fall2009.htm

Outline for Today

- A brief review and wrap-up for last class on 2D transformations
- 3D transformations
- Transformations from 3D → 2D: Projections

Review of Last class

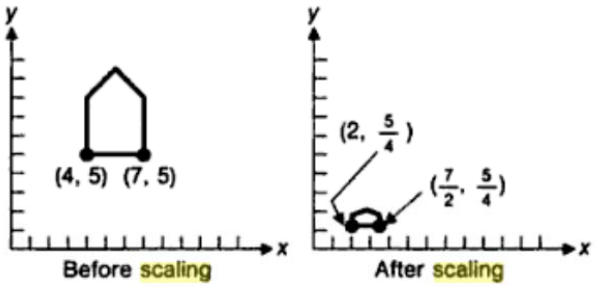
- Set up GLUT in your system
  - Compile and run the "hello world" program
- Review related linear algebra
  - Points, vectors, frames
  - Transformations
  - Homogeneous Coordinates

Review of Last class (cont.)

- Derivation of 2D rotation transformation matrix
- Scaling transformation matrix
- They are about the origin
  - Rotation: the whole space rotates
  - Scaling: the object will be further or closer to the origin depending on the scales
- Rotation/Scaling Center
  - To transform around the shape center, or any given point p: (3-step Composition)
    - Translate p to the origin
    - Conduct the transformation
    - Translate back

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$



Composition of Transformations

With homogeneous coordinates, affine transformations and their composition can be represented by matrices, and their product

Examples:

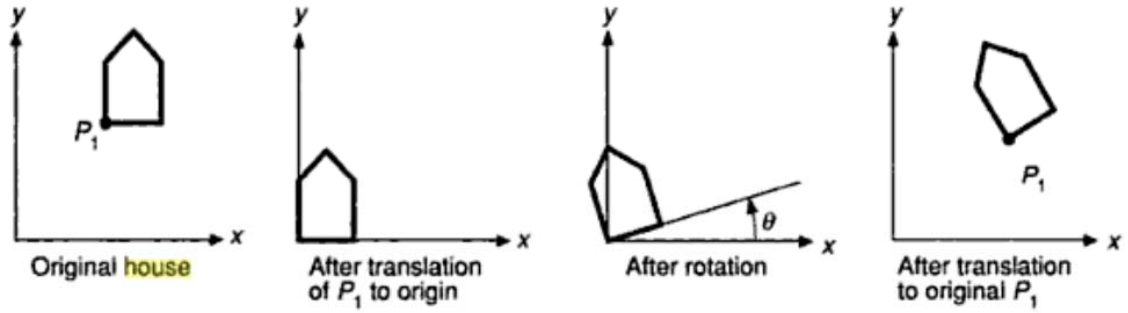
$$\begin{bmatrix} 1 & & d_{x2} \\ & 1 & d_{y2} \\ & & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & & d_{x1} \\ & 1 & d_{y1} \\ & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & d_{x1}+d_{x2} \\ & 1 & d_{y1}+d_{y2} \\ & & 1 \end{bmatrix} \quad \begin{bmatrix} s_{x2} & & \\ & s_{y2} & \\ & & 1 \end{bmatrix} \bullet \begin{bmatrix} s_{x1} & & \\ & s_{y2} & \\ & & 1 \end{bmatrix} = \begin{bmatrix} s_{x1}\cdot s_{x2} & & \\ & s_{y1}\cdot s_{y2} & \\ & & 1 \end{bmatrix}$$

- Rigid Transformation
  - ← product of an arbitrary sequence of rotation and translation matrices
  - preserving length, angles
- Affine transformation
  - ← product of an arbitrary sequence of rotation, translation, scale, and shear matrices
  - preserving parallelism

Composition of Transformations (cont.)

- Combine fundamental R, S, and T matrices → produce desired general affine transformation
- Gain efficiency by applying a single composed transformation, rather than a series of transformations

Examples: Rotating a house about a point P_1 by an angle

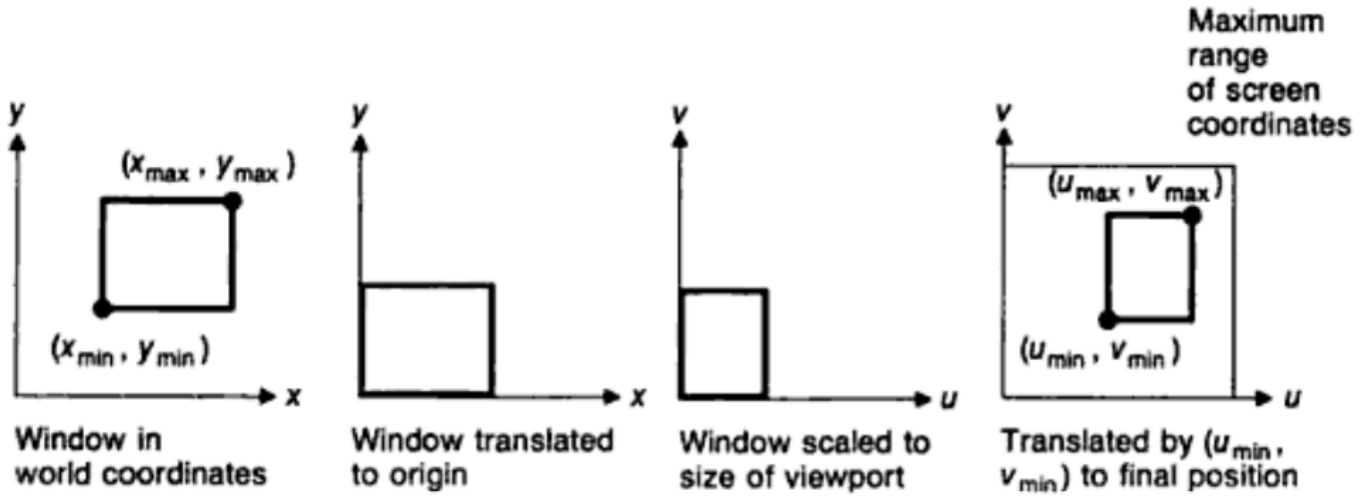


$$T(x_1, y_1) \cdot R(\theta) \cdot T(-x_1, -y_1) = \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_1(1 - \cos\theta) + y_1\sin\theta \\ \sin\theta & \cos\theta & y_1(1 - \cos\theta) - x_1\sin\theta \\ 0 & 0 & 1 \end{bmatrix}.$$

2D Window-To-Viewport Transformation

- We get objects (2D) represented in world-coordinate system
- We need to map them onto screen coordinates
- A common question:
  - Given a rectangular region in world coordinates (world-coordinate window)
  - A corresponding rectangular region in screen coordinates (viewport)
  - To find the transformation



Sometimes, we want the isometry.

Matrices for 3D Transformations

- 3D transformations → 4 by 4 matrices, using homogeneous coordinates
  - A point (x,y,z) → points (Wx, Wy, Wz, W), or (x/W, y/W, z/W, 1)
  - W=0 → corresponds to the point at infinity
- 3D Translation and Scaling are extended from 2D case straightforwardly.
- As for Rotation:

The positive direction of 3D Rotations in right-handed system:

positive rotations = when looking from a positive axis toward the origin, a 90 degree counterclockwise rotation will transform one positive axis into the other, according to the following table:

Rotation axis = x → positive rotation is from y to z

axis = y → from z to x

axis = z → from x to y

Matrices for 3D Transformations

- 3D Rotation:
  - Previous 2D rotation $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$ can be treated as a 3D rotation about z axis
  - i.e. $\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$
  - Rotation matrices about x and y axes: $R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ $R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

1. All these basic transformation matrices have inverse, therefore any affine transformation composed by them does too.
2. Any number of rotation, scaling, and translation matrices can be multiplied together. The result always has the form:

$$M = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Transforming lines and planes

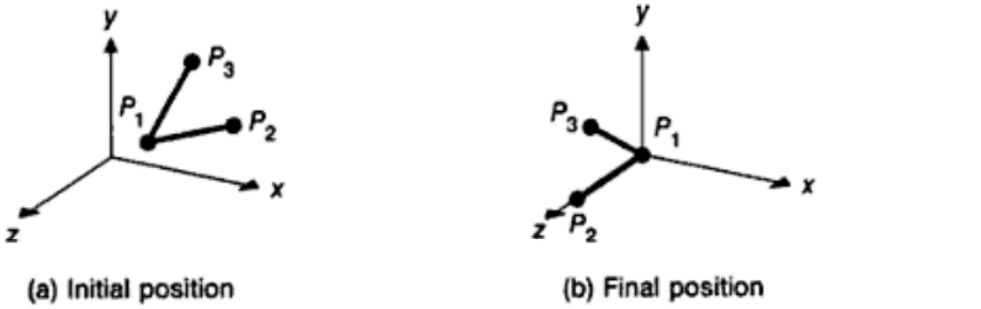
Each transformation matrix applies on vectors, or individual points.

- Transforming lines by transforming the endpoints
- Transforming triangles by transforming the three vertices.
- Transforming planes (represented using $N=[A\ B\ C\ D]^T$, and defined through $\{P|N \bullet P=0\}$) by transforming its normal.

Composition of 3D Transformations

Example: A common situation when we setup a local 3D coordinate system.

Given the directed line segments P_1P_2 and P_1P_3 in (a), find the transformation to transform it to their ending positions in (b): P_1 at the origin, P_1P_2 on positive z-axis, and P_1P_3 in the positive y axis half of the (y,z) plane



Transforming P_1, P_2, and P_3 from their initial (a) to their final (b) position.

Show two ways to solve it:

(1) Apply a sequential primitive transformations: translation + rotations…

(2) Using the properties of orthogonal matrices

Method 1

(a) Initial position (b) Final position

Transforming P_1, P_2, and P_3 from their initial (a) to their final (b) position.

Break it into simpler sub-problems, we can get it in four steps:

1. Translate P_1 to the origin
2. Rotate about the y-axis → P_1P_2 lies in the (y,z) plane
3. Rotate about the x-axis → P_1P_2 lies on the z-axis
4. Rotate about the z-axis → P_1P_3 lies in the (y,z) plane

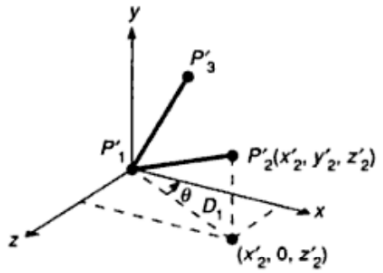
Step 1. Translate.

$$T_1(-x_1,-y_1,-z_1)=\begin{bmatrix}1&0&0&-x_1\\0&1&0&-y_1\\0&0&1&-z_1\\0&0&0&1\end{bmatrix}$$

$$P'_1 = T(-x_1, -y_1, -z_1) \cdot P_1 = \begin{bmatrix}0\\0\\0\\1\end{bmatrix},$$

$$P'_2 = T(-x_1, -y_1, -z_1) \cdot P_2 = \begin{bmatrix}x_2 - x_1\\y_2 - y_1\\z_2 - z_1\\1\end{bmatrix},$$

$$P'_3 = T(-x_1, -y_1, -z_1) \cdot P_3 = \begin{bmatrix}x_3 - x_1\\y_3 - y_1\\z_3 - z_1\\1\end{bmatrix},$$



Method 1

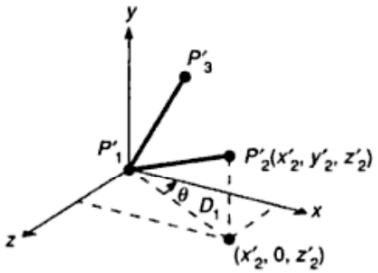
(a) Initial position (b) Final position

Transforming P_1, P_2, and P_3 from their initial (a) to their final (b) position.

Break it into simpler sub-problems, we can get it in four steps:

1. Translate P_1 to the origin
2. Rotate about the y-axis → P_1P_2 lies in the (y,z) plane
3. Rotate about the x-axis → P_1P_2 lies on the z-axis
4. Rotate about the z-axis → P_1P_3 lies in the (y,z) plane

Step 2. Need to rotate about y-axis by –(90-θ)=(θ-90)



$$R_2 = R_y(\theta - 90^\circ)$$

$$\cos(\theta - 90) = \sin\theta = \frac{z_2'}{D_1} = \frac{z_2 - z_1}{D_1},$$

$$\sin(\theta - 90) = -\cos\theta = -\frac{x_2'}{D_1} = -\frac{x_2 - x_1}{D_1},$$

where

$$D_1 = \sqrt{(z_2')^2 + (x_2')^2} = \sqrt{(z_2 - z_1)^2 + (x_2 - x_1)^2}.$$

Method 1

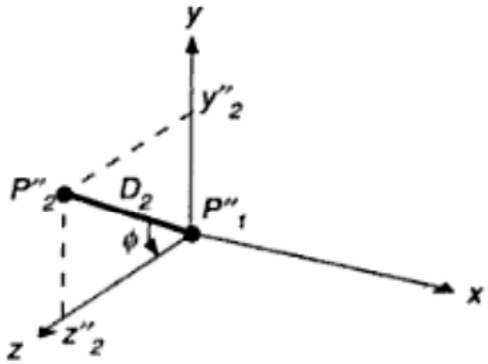
(a) Initial position (b) Final position

Transforming P_1, P_2, and P_3 from their initial (a) to their final (b) position.

Break it into simpler sub-problems, we can get it in four steps:

1. Translate P_1 to the origin
2. Rotate about the y-axis → P_1P_2 lies in the (y,z) plane
3. Rotate about the x-axis → P_1P_2 lies on the z-axis
4. Rotate about the z-axis → P_1P_3 lies in the (y,z) plane

Step 3. Need to rotate about x-axis by Φ



$$R_3 = R_x(\phi)$$

$$\cos\phi = \frac{z_2''}{D_2}, \quad \sin\phi = \frac{y_2''}{D_2},$$

where

$$D_2 = |P_1''P_2''| = |P_1P_2| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}.$$

Method 1

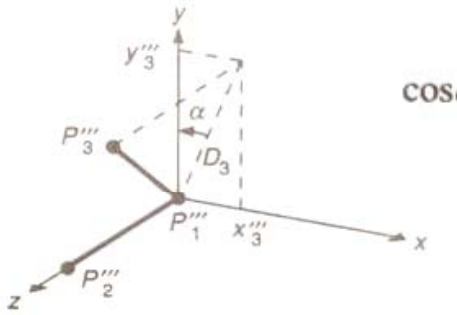
(a) Initial position (b) Final position

Transforming P_1, P_2, and P_3 from their initial (a) to their final (b) position.

Break it into simpler sub-problems, we can get it in four steps:

1. Translate P_1 to the origin
2. Rotate about the y-axis → P_1P_2 lies in the (y,z) plane
3. Rotate about the x-axis → P_1P_2 lies on the z-axis
4. Rotate about the z-axis → P_1P_3 lies in the (y,z) plane

Step 4. Need to rotate about z-axis by α



$$R_4 = R_z(\alpha)$$

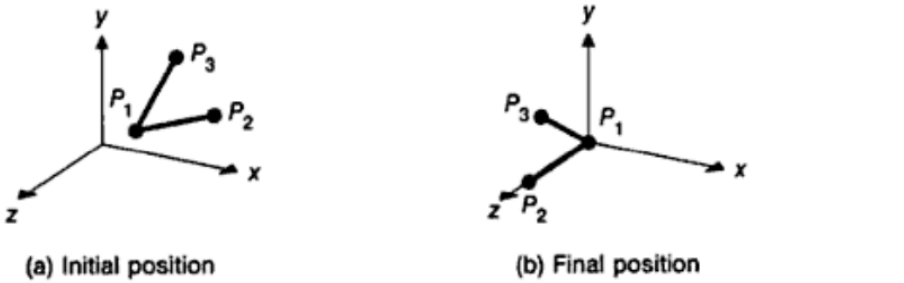
$$\cos\alpha = y_3'''/D_3, \quad \sin\alpha = x_3'''/D_3, \quad D_3 = \sqrt{x_3'''^2 + y_3'''^2}.$$

So the final composited transformation is:

$$R_z(\alpha) \cdot R_x(\phi) \cdot R_y(\theta - 90) \cdot T(-x_1, -y_1, -z_1)$$

Method 2

A generally useful quicker way to get the rotation matrix.



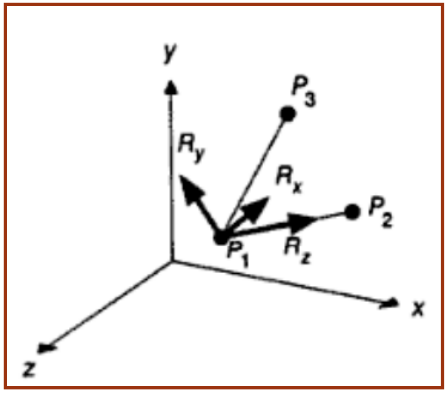
Transforming P_1, P_2, and P_3 from their initial (a) to their final (b) position.

To obtain the rotation matrix R using the properties of orthogonal matrices

1) Each row is a unit vector
2) Each row is perpendicular to the other
3) R's transpose is its inverse matrix
4) The i^{th} row (except the last row) as a vector, will be rotated by R(θ) to lie on the positive axis $\mathbf{e}_i$

$$\begin{bmatrix} r_{1x} & r_{2x} & r_{3x} & 0 \\ r_{1y} & r_{2y} & r_{3y} & 0 \\ r_{1z} & r_{2z} & r_{3z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

If R rotates a normalized vector v to $\mathbf{e}_i$, then its i^{th} row is v^T



1. we want to rotate P_1P_2 to "+z"-axis, so the 3rd row should be the normalized P_1P_2 :

$$R_z = [r_{1z} \quad r_{2z} \quad r_{3z}]^T = \frac{P_1P_2}{|P_1P_2|}.$$

2. R_x unit vector is perpendicular to the plane of P_1, P_2 and P_3, and will rotate into the "+x"-axis, so the 1st row:

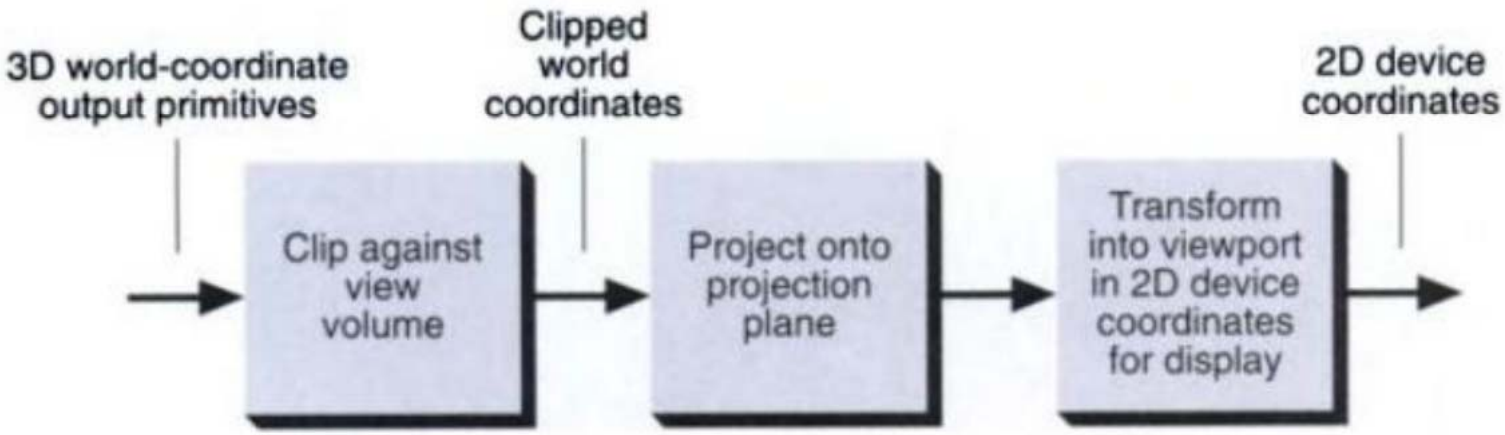
$$R_x = [r_{1x} \quad r_{2x} \quad r_{3x}]^T = \frac{P_1P_3 \times P_1P_2}{|P_1P_3 \times P_1P_2|}$$

3. Finally:

$$R_y = [r_{1y} \quad r_{2y} \quad r_{3y}]^T = R_z \times R_x$$

Viewing in 3D

2D shapes → window clip, translate, scale, translate → 2D viewport
3D shapes → projection → 2D viewport
(view volume clip → projection → 2D transform)



Conceptual model of the 3D viewing process.

Projections

Generally:

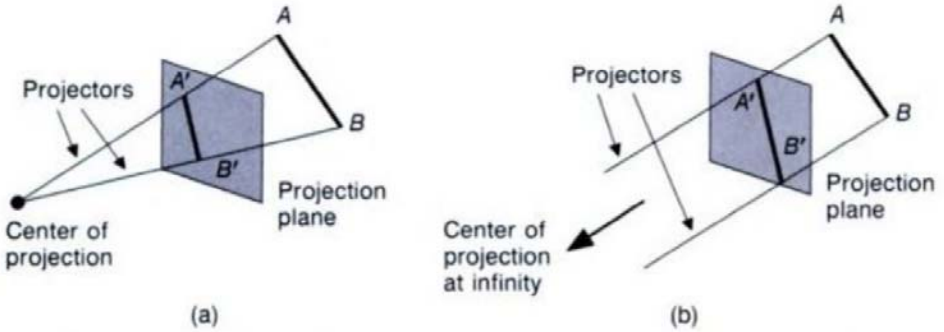
❑Projections transform points in a n-D coordinate system into points in a m-D coordinate system (m<n)

❑Computer Graphics has long been used for studying n-D objects by projecting them into lower dimensional (especially, 2D) space.

Noll, M. , "A Computer Technique for Displaying N-dimensional hyperobjects", CACM, 10(8), Aug. 1967, 469-473.

Here:

❑We focus on projections from 3D to 2D.



(a) Line *AB* and its perspective projection *A′B′*. (b) Line *AB* and its parallel projection *A′B′*. Projectors *AA′* and *BB′* are parallel.

❑Projection :

- ❑straight projection rays (projectors) emanating from a center of projection
- ❑Passing through each point of the object
- ❑Intersecting a projection plane to form the projection image

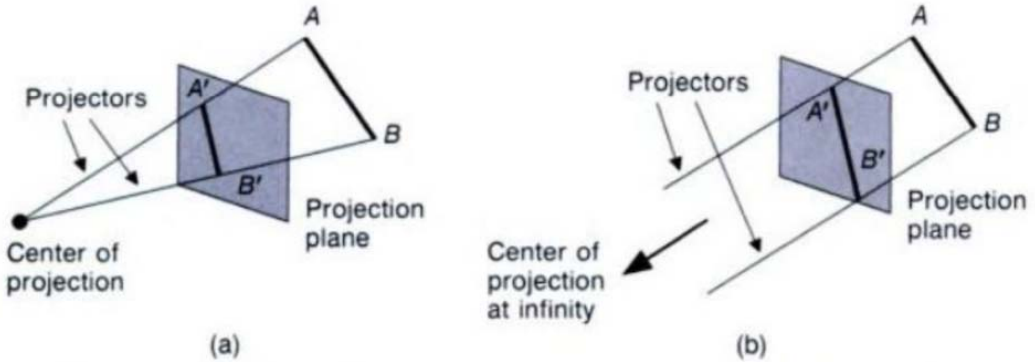
Classification of Projections

General Classification:

- We deal with planar geometric projections
- Non-planar projection: the projection plane is a curved surface (e.g. many cartographic projections)
- Non-geometric projection: the projection rays are curved (e.g. the Omnimax film)

Planar Geometric Projections:

- Parallel projection: projection center is infinitely far away
  - so that all projectors are parallel
  - We only need to specify direction of projection
- Perspective projection: projection center is finite distance away
  - Need to specify projection center



(a) Line AB and its perspective projection $A'B'$. (b) Line AB and its parallel projection $A'B'$. Projectors AA' and BB' are parallel.

Classification of Projections

General Classification:

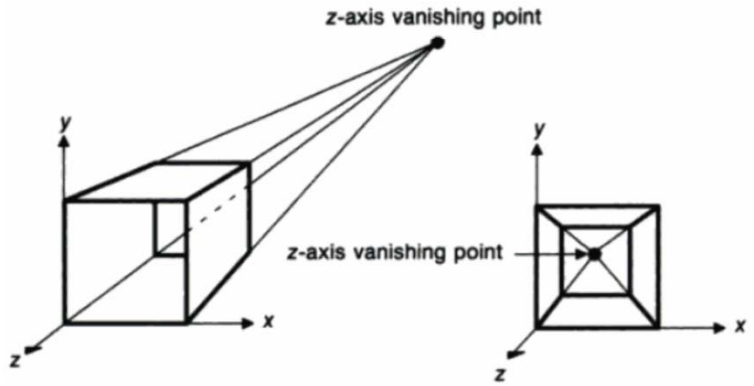
Planar Geometric Projections:

- Perspective projection:
  - Visually : perspective foreshortening (the object size varies inversely with the distance from the projection center), similar to human visual system
  - Measurement:
    - not good for recording exact shape,
    - angles are preserved only on those faces of the object parallel to the projection plane,
    - parallel lines generally are not projected to be parallel
- Parallel projection:
  - Visually : less realistic
  - Measurement:
    - good for exact measurement,
    - parallel lines remain parallel,
    - angles only preserved on faces that are parallel to the projection plane

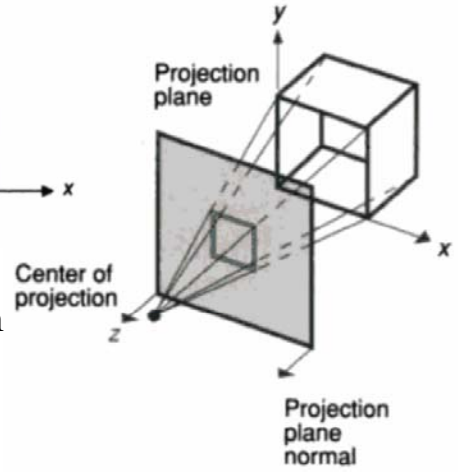
For more detailed discussions, check: Carlbom, I. and J. Paciorek, "Planar Geometric Projections and Viewing Transformations", Computing Surveys, 10(4), Dec. 1978, pp. 465-502.

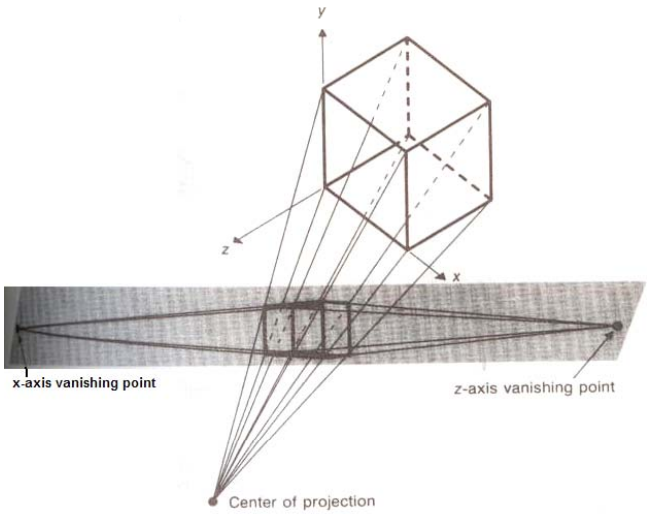
Perspective Projections

- Any set of parallel lines, that are not parallel to the projection plane, converge to a vanishing point.
- Axis vanishing point: if the set of lines that converges is parallel to one of the three principal axes
- There are at most three such points
  - e.g. if the projection plane cuts only the z-axis, then only the z axis has a principal vanishing point (lines parallel to either x or y axes have no vanishing point)
- Perspective projections are categorized by # of principal vanishing pts (i.e. by the # of axes the projection plane cuts)



One point perspective projection





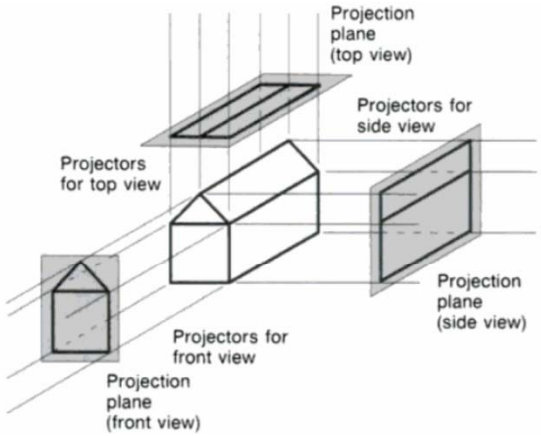
Two point perspective projection

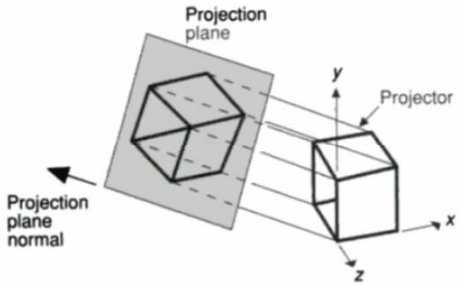
Parallel Projections

- Categorized into two types, depending on the relation between (1) the direction of projection and (2) the normal to the projection plane:
  - **Orthographic parallel projections:** (1) and (2) have the same direction
  - Oblique parallel projections: (1) and (2) have different directions

Orthographic Parallel projections:

- Front,top,side-elevation projections:
  - Often used in engineering drawings
  - Distances and angles can be measured
  - But…
- Axonometric orthographic projections:
  - Projections not normal to a principal axis
  - Parallelism of lines preserved, angles not preserved
  - Distances can be measured along each principal axis (with different scales)
  - A commonly used type: isometric projection
    → projction direction makes equal angles with each axis (e.g. direction (1,1,1), every axis pair looks like 120 degree)



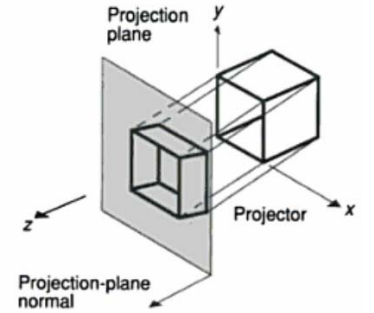


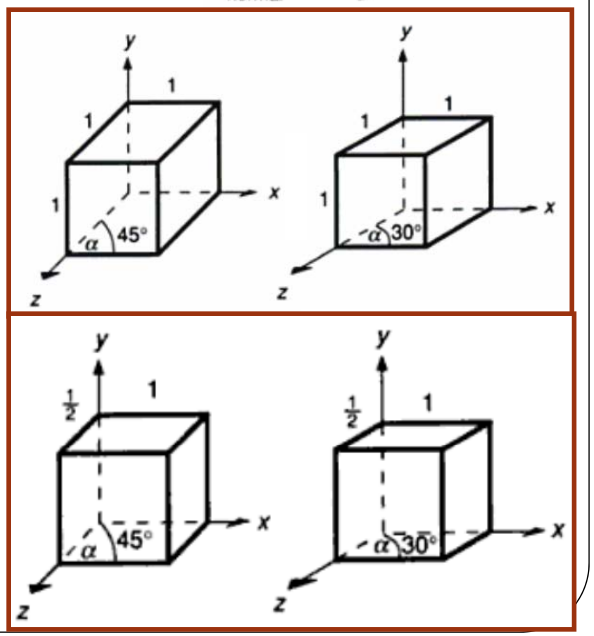
Parallel Projections (cont.)

- Categorized into two types, depending on the relation between (1) the direction of projection and (2) the normal to the projection plane:
  - Orthographic parallel projections: (1) and (2) have the same direction
  - **Oblique parallel projections:** (1) and (2) have different directions

Oblique Parallel projections:

- Cavalier projections:
  - Projection direction makes a 45° angle with the projection plane
  - Result: line perpendicular to the projection plane preserves length, no foreshortening
  - 2 examples, right middle 2 figs.
- Cabinet projections:
  - Projection direction makes a 63.4° angle with the projection plane
  - Result: line perpendicular to the projection plane has half length
  - Visually more realistic, right bottom 2 figs





Planar Geometric Projections