
CIS 454 Computer Graphics

Lecture 12, 10/17/2006

Li Shen
Computer and Information Science
UMass Dartmouth

Notes

- HW2 due today
-

Geometry

- Introduce the elements of geometry
 - Scalars
 - Vectors
 - Points
 - Develop mathematical operations among them in a coordinate-free manner
 - Define basic primitives
 - Line segments
 - Polygons
-

Linear Algebra

- Basic matrix operations (+, -, *)
 - Cross and dot products
 - Determinants and inverses
-

Basic Operations

- Transpose: Swap rows with columns

$$M = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \quad M^T = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

$$V = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad V^T = [x \quad y \quad z]$$

Basic Operations

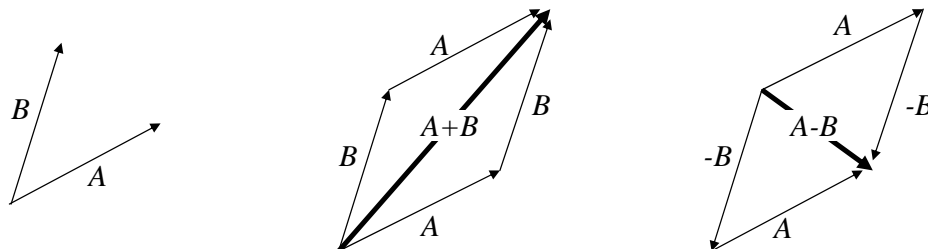
- Addition and Subtraction

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} - \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a-e & b-f \\ c-g & d-h \end{bmatrix}$$

Just add elements

Just subtract elements



Basic Operations

- Multiplication

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix} \quad \text{Multiply each row by each column}$$

An $m \times n$ can be multiplied by an $n \times p$ matrix to yield an $m \times p$ result

- Goal: transform column vectors

$$b = M a$$

Vectors: Dot Product

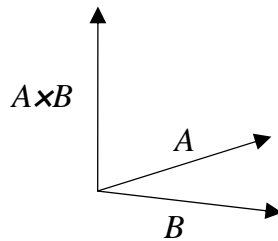
$$A \cdot B = AB^T = \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} d \\ e \\ f \end{bmatrix} = ad + be + cf \quad \text{Think of the dot product as a matrix multiplication}$$

$$\|A\|^2 = AA^T = aa + bb + cc \quad \text{The magnitude is the dot product of a vector with itself}$$

$$A \cdot B = \|A\| \|B\| \cos(\theta) \quad \text{The dot product is also related to the angle between the two vectors}$$

Vectors: Cross Product

The cross-product can be computed as a specially constructed determinant



$$A \times B = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = \begin{bmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{bmatrix}$$

$$\|A \times B\| = \|A\| \|B\| \sin(\theta)$$

Inverse of a Matrix

- Identity matrix:
 $AI = A$
- Some matrices have an inverse, such that:
 $AA^{-1} = I$
- Inversion is tricky:
 $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$
Derived from non-commutativity property

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Determinant of a Matrix

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - afh - bdi - ceg$$

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix}$$

For a 3x3 matrix:
Sum from left to right
Subtract from right to left

Note: In the general case, the determinant has $n!$ terms

Representation

Objectives

- Introduce concepts such as dimension and basis
 - Introduce coordinate systems for representing vectors spaces and frames for representing affine spaces
 - Discuss change of frames and bases
 - Introduce homogeneous coordinates
-

Linear Independence

- A set of vectors v_1, v_2, \dots, v_n is *linearly independent* if

$$\alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n = 0 \text{ iff } \alpha_1 = \alpha_2 = \dots = 0$$

- If a set of vectors is linearly independent, we cannot represent one in terms of the others
 - If a set of vectors is linearly dependent, at least one can be written in terms of the others
-

Dimension

- In a vector space, the maximum number of linearly independent vectors is fixed and is called the *dimension* of the space
- In an n -dimensional space, any set of n linearly independent vectors form a *basis* for the space
- Given a basis v_1, v_2, \dots, v_n , any vector v can be written as

$$v = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$$

where the $\{\alpha_i\}$ are unique

Representation

- Until now we have been able to work with geometric entities without using any frame of reference, such as a coordinate system
 - Need a frame of reference to relate points and objects to our physical world.
 - For example, where is a point? Can't answer without a reference system
 - World coordinates
 - Camera coordinates
-

Coordinate Systems

- Consider a basis v_1, v_2, \dots, v_n
- A vector is written $v = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$
- The list of scalars $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ is the *representation* of v with respect to the given basis
- We can write the representation as a row or column array of scalars

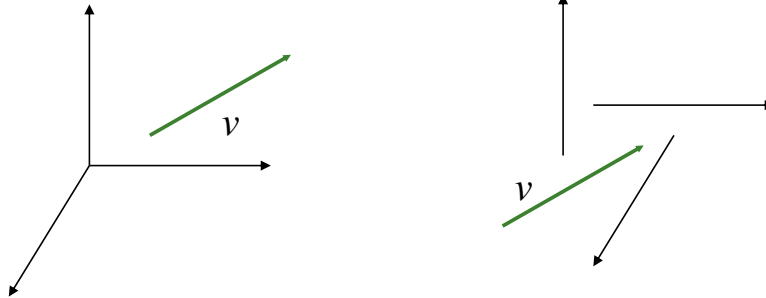
$$\mathbf{a} = [\alpha_1 \quad \alpha_2 \quad \dots \quad \alpha_n]^T = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \cdot \\ \alpha_n \end{bmatrix}$$

Example

- $v = 2v_1 + 3v_2 - 4v_3$
- $\mathbf{a} = [2 \ 3 \ -4]^T$
- Note that this representation is with respect to a particular basis
- For example, in OpenGL we start by representing vectors using the object basis but later the system needs a representation in terms of the camera or eye basis

Coordinate Systems

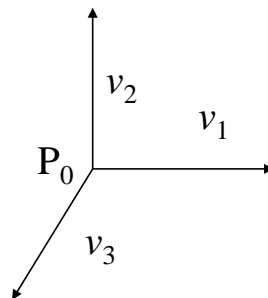
- Which is correct?



- Both are because vectors have no fixed location

Frames

- A coordinate system is insufficient to represent points
- If we work in an affine space we can add a single point, the *origin*, to the basis vectors to form a *frame*



Representation in a Frame

- Frame determined by (P_0, v_1, v_2, v_3)
- Within this frame, every vector can be written as

$$v = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$$

- Every point can be written as

$$P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \dots + \beta_n v_n$$

Confusing Points and Vectors

Consider the point and the vector

$$P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \dots + \beta_n v_n$$

$$v = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$$

They appear to have the similar representations

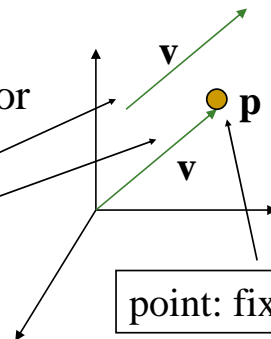
$$\mathbf{P} = [\beta_1 \beta_2 \beta_3] \quad \mathbf{v} = [\alpha_1 \alpha_2 \alpha_3]$$

which confuses the point with the vector

A vector has no position

Vector can be placed anywhere

point: fixed



A Single Representation

If we define $0 \cdot P = \mathbf{0}$ and $1 \cdot P = P$ then we can write

$$\mathbf{v} = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 = [\alpha_1 \ \alpha_2 \ \alpha_3 \ 0] [v_1 \ v_2 \ v_3 \ P_0]^T$$

$$P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3 = [\beta_1 \ \beta_2 \ \beta_3 \ 1] [v_1 \ v_2 \ v_3 \ P_0]^T$$

Thus we obtain the four-dimensional *homogeneous coordinate* representation

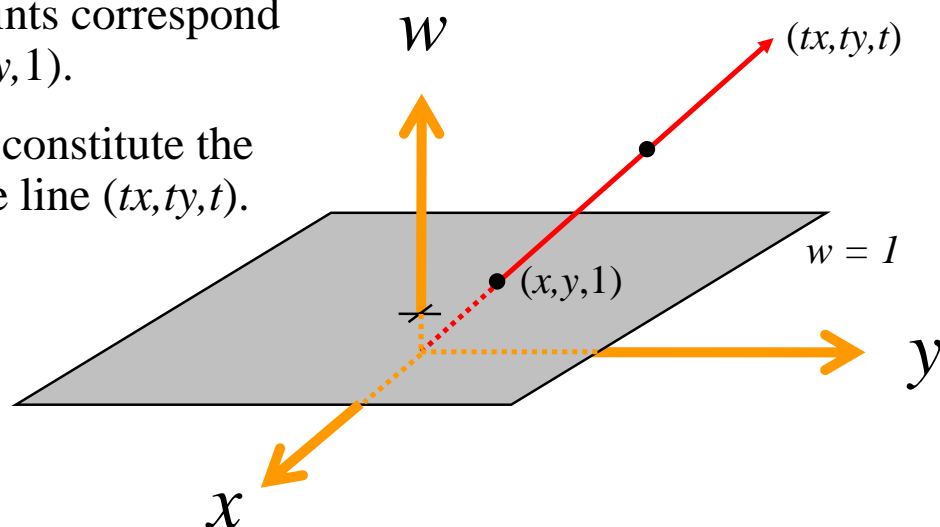
$$\mathbf{v} = [\alpha_1 \ \alpha_2 \ \alpha_3 \ 0]^T$$

$$\mathbf{p} = [\beta_1 \ \beta_2 \ \beta_3 \ 1]^T$$

Homogeneous Coordinates: 2D

An infinite number of points correspond to $(x, y, 1)$.

They constitute the whole line (tx, ty, t) .



Homogeneous Coordinates: 3D

The homogeneous coordinates form for a three dimensional point $[x \ y \ z]$ is given as

$$\mathbf{p} = [x' \ y' \ z' \ w]^T = [wx \ wy \ wz \ w]^T$$

We return to a three dimensional point (for $w \neq 0$) by

$$x \leftarrow x'/w$$

$$y \leftarrow y'/w$$

$$z \leftarrow z'/w$$

If $w=0$, the representation is that of a vector

Note that homogeneous coordinates replaces points in three dimensions by lines through the origin in four dimensions

For $w=1$, the representation of a point is $[x \ y \ z \ 1]$

Homogeneous Coordinates and Computer Graphics

- Homogeneous coordinates are key to all computer graphics systems
 - All standard transformations (rotation, translation, scaling) can be implemented with matrix multiplications using 4×4 matrices
 - Hardware pipeline works with 4 dimensional representations
 - For orthographic viewing, we can maintain $w=0$ for vectors and $w=1$ for points
 - For perspective we need a *perspective division*

Change of Coordinate Systems

- Consider two representations of a the same vector with respect to two different bases. The representations are

$$\mathbf{a} = [\alpha_1 \ \alpha_2 \ \alpha_3]^T$$

$$\mathbf{b} = [\beta_1 \ \beta_2 \ \beta_3]^T$$

where

$$\begin{aligned} \mathbf{v} &= \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \alpha_3 \mathbf{v}_3 = [\alpha_1 \ \alpha_2 \ \alpha_3] [\mathbf{v}_1 \ \mathbf{v}_2 \ \mathbf{v}_3]^T \\ &= \beta_1 \mathbf{u}_1 + \beta_2 \mathbf{u}_2 + \beta_3 \mathbf{u}_3 = [\beta_1 \ \beta_2 \ \beta_3] [\mathbf{u}_1 \ \mathbf{u}_2 \ \mathbf{u}_3]^T \end{aligned}$$

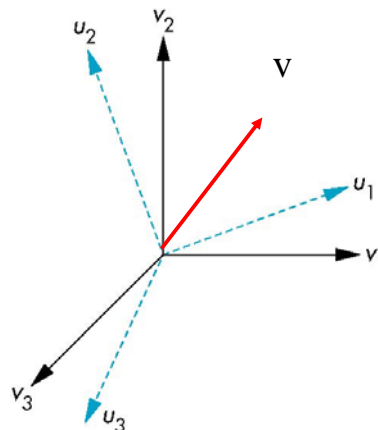
Representing second basis in terms of first

Each of the basis vectors, $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$, are vectors that can be represented in terms of the first basis

$$\mathbf{u}_1 = \gamma_{11} \mathbf{v}_1 + \gamma_{12} \mathbf{v}_2 + \gamma_{13} \mathbf{v}_3$$

$$\mathbf{u}_2 = \gamma_{21} \mathbf{v}_1 + \gamma_{22} \mathbf{v}_2 + \gamma_{23} \mathbf{v}_3$$

$$\mathbf{u}_3 = \gamma_{31} \mathbf{v}_1 + \gamma_{32} \mathbf{v}_2 + \gamma_{33} \mathbf{v}_3$$



Matrix Form

The coefficients define a 3 x 3 matrix

$$\mathbf{M} = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix}$$

and the bases can be related by

$$\mathbf{a} = \mathbf{M}^T \mathbf{b}$$

see text for numerical examples

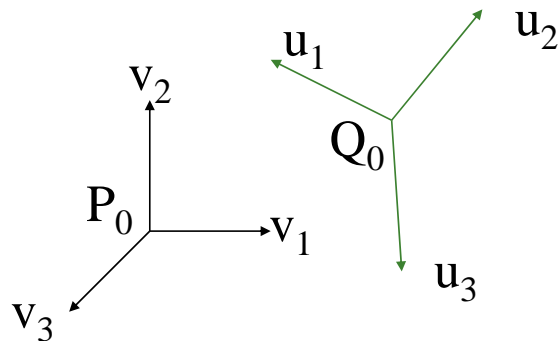
Change of Frames

- We can apply a similar process in homogeneous coordinates to the representations of both points and vectors

Consider two frames:

(P_0, v_1, v_2, v_3)

(Q_0, u_1, u_2, u_3)



- Any point or vector can be represented in either frame
- We can represent Q_0, u_1, u_2, u_3 in terms of P_0, v_1, v_2, v_3

Representing One Frame in Terms of the Other

Extending what we did with change of bases

$$\mathbf{u}_1 = \gamma_{11}\mathbf{v}_1 + \gamma_{12}\mathbf{v}_2 + \gamma_{13}\mathbf{v}_3$$

$$\mathbf{u}_2 = \gamma_{21}\mathbf{v}_1 + \gamma_{22}\mathbf{v}_2 + \gamma_{23}\mathbf{v}_3$$

$$\mathbf{u}_3 = \gamma_{31}\mathbf{v}_1 + \gamma_{32}\mathbf{v}_2 + \gamma_{33}\mathbf{v}_3$$

$$\mathbf{Q}_0 = \gamma_{41}\mathbf{v}_1 + \gamma_{42}\mathbf{v}_2 + \gamma_{43}\mathbf{v}_3 + \mathbf{P}_0$$

defining a 4 x 4 matrix

$$\mathbf{M} = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} & 0 \\ \gamma_{21} & \gamma_{22} & \gamma_{23} & 0 \\ \gamma_{31} & \gamma_{32} & \gamma_{33} & 0 \\ \gamma_{41} & \gamma_{42} & \gamma_{43} & 1 \end{bmatrix}$$

Working with Representations

Within the two frames any point or vector has a representation of the same form

$\mathbf{a} = [\alpha_1 \ \alpha_2 \ \alpha_3 \ \alpha_4]$ in the first frame

$\mathbf{b} = [\beta_1 \ \beta_2 \ \beta_3 \ \beta_4]$ in the second frame

where $\alpha_4 = \beta_4 = 1$ for points and $\alpha_4 = \beta_4 = 0$ for vectors and

$$\mathbf{a} = \mathbf{M}^T \mathbf{b}$$

The matrix \mathbf{M} is 4 x 4 and specifies an affine transformation in homogeneous coordinates

Affine Transformations

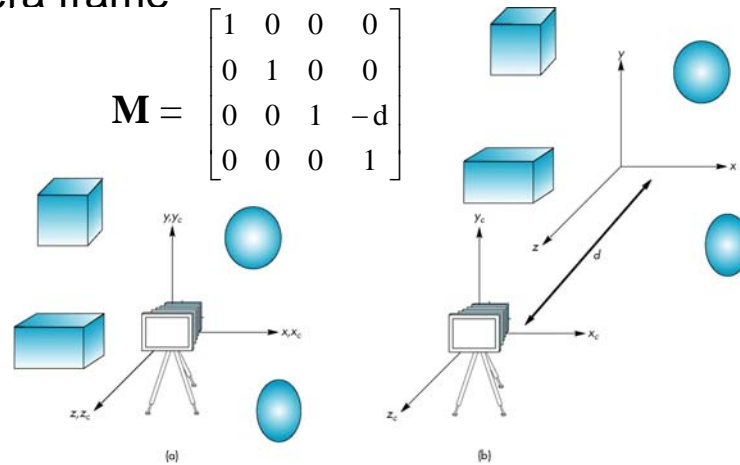
- Every linear transformation is equivalent to a change in frames
 - Every affine transformation preserves lines
 - However, an affine transformation has only 12 *degrees of freedom* because 4 of the elements in the matrix are fixed and are a subset of all possible 4 x 4 linear transformations
-

The World and Camera Frames

- When we work with representations, we work with n-tuples or arrays of scalars
 - Changes in frame are then defined by 4 x 4 matrices
 - In OpenGL, the base frame that we start with is the world frame
 - Eventually we represent entities in the camera frame by changing the world representation using the model-view matrix
 - Initially these frames are the same ($\mathbf{M}=\mathbf{I}$)
-

Moving the Camera

If objects are on both sides of $z=0$, we must move camera frame



Summary: Representation

- Introduce concepts such as dimension and basis
- Introduce coordinate systems for representing vectors spaces and frames for representing affine spaces
- Discuss change of frames and bases
- Introduce homogeneous coordinates

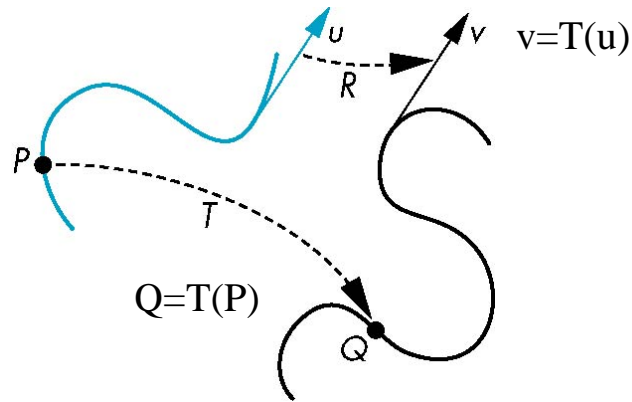
Transformations

Objectives

- Introduce standard transformations
 - Rotation
 - Translation
 - Scaling
 - Shear
 - Derive homogeneous coordinate transformation matrices
 - Learn to build arbitrary transformation matrices from simple transformations
-

General Transformations

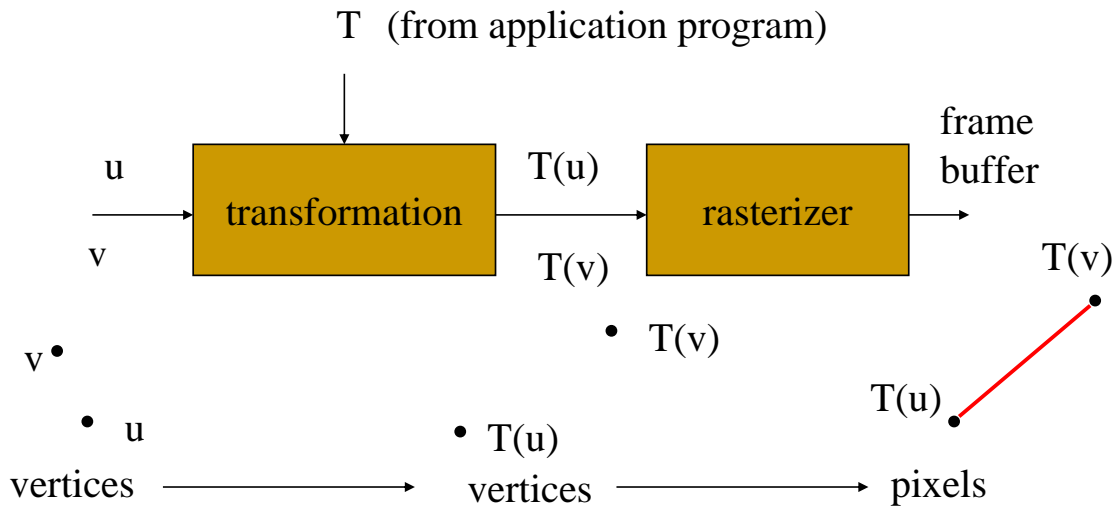
A transformation maps points to other points and/or vectors to other vectors



Affine Transformations

- Line preserving
- Characteristic of many physically important transformations
 - Rigid body transformations: rotation, translation
 - Scaling, shear
- Importance in graphics is that we need only transform endpoints of line segments and let implementation draw line segment between the transformed endpoints

Pipeline Implementation



Notation

We will be working with both coordinate-free representations of transformations and representations within a particular frame

P, Q, R : points in an affine space

u, v, w : vectors in an affine space

α, β, γ : scalars

$\mathbf{p}, \mathbf{q}, \mathbf{r}$: representations of points

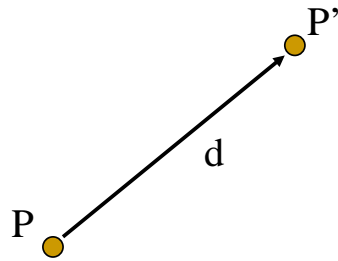
-array of 4 scalars in homogeneous coordinates

$\mathbf{u}, \mathbf{v}, \mathbf{w}$: representations of points

-array of 4 scalars in homogeneous coordinates

Translation

- Move (translate, displace) a point to a new location

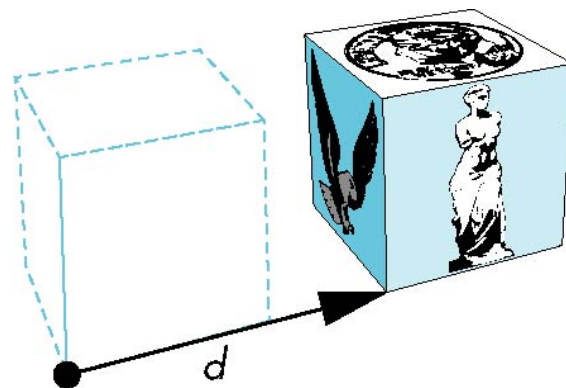


- Displacement determined by a vector d
 - Three degrees of freedom
 - $P' = P + d$

Move to a New Location



object



translation: every point displaced
by same vector

Translation Using Representations

Using the homogeneous coordinate representation in some frame

$$\mathbf{p} = [x \ y \ z \ 1]^T$$

$$\mathbf{p}' = [x' \ y' \ z' \ 1]^T$$

$$\mathbf{d} = [dx \ dy \ dz \ 0]^T$$

Hence $\mathbf{p}' = \mathbf{p} + \mathbf{d}$ or

$$x' = x + d_x$$

$$y' = y + d_y$$

$$z' = z + d_z$$

note that this expression is in four dimensions and expresses point = vector + point

Translation Matrix

We can also express translation using a 4 x 4 matrix \mathbf{T} in homogeneous coordinates

$\mathbf{p}' = \mathbf{T}\mathbf{p}$ where

$$\mathbf{T} = \mathbf{T}(d_x, d_y, d_z) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This form is better for implementation because all affine transformations can be expressed this way and multiple transformations can be concatenated together

After Class

- Read
 - Chapter 4
- Review book materials and prepare for midterm

