Graphics & Visualization

Chapter 3

2D and 3D Coordinate Systems and Transformations

Introduction

- In computer graphics is often necessary to change:
  - the form of the objects
  - the coordinate system
- Examples:
  - In a model of a scene, the digitized form of a car may be used in several instances, positioned at various points & directions and in different sizes
  - In animation, an object may be transformed from frame to frame
  - As objects traverse the graphics pipeline, they change their coordinate system: object coordinates → world coordinates

    world coordinates → eye coordinates
  - Coordinates transformations:

    - tools of change

    - the most important & classic topic in computer graphics

Introduction (2)

- Points in 3D Euclidean space E^3: 3×1 column vectors (here)

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

- Linear transformations: - 3×3 matrices
  - they are **post-multiplied** by a point to produce another point

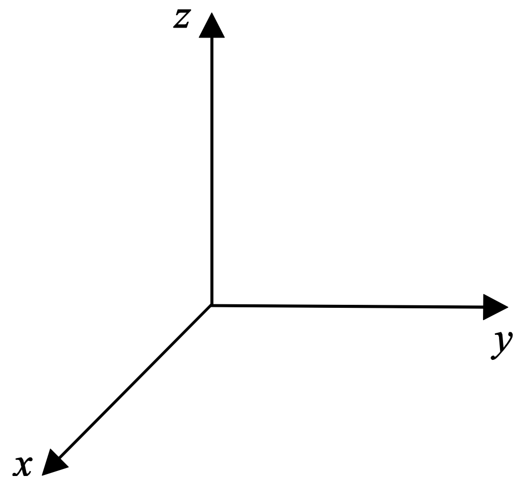
$$\begin{bmatrix} p_{x'} \\ p_{y'} \\ p_{z'} \end{bmatrix} = \begin{bmatrix} m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 \\ m_7 & m_8 & m_9 \end{bmatrix} \cdot \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

Introduction (3)

- If points were represented by 1×3 row vector $\mathbf{p} = [p_x \; p_y \; p_z]$ the linear transformations would be **pre-multiplied** by the point

$$\begin{bmatrix} p_{x'} & p_{y'} & p_{z'} \end{bmatrix} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix} \cdot \begin{bmatrix} m_1 & m_4 & m_7 \\ m_2 & m_5 & m_8 \\ m_3 & m_6 & m_9 \end{bmatrix}$$

- Right-handed three-dimensional coordinate systems are used:



Affine Transformations - Combinations

- Mathematics: **Transformations**:

  mappings whose domain & range are the same set (e.g. E^3 to E^3)

- Computer graphics & visualization: **Affine Transformations**:

  transformations which preserve important geometric properties of the objects being transformed

- Affine transformations preserve affine combinations
- Examples of Affine combinations:
  - line segments, convex polygons, triangles, tetrahedra → the building blocks of our models

Affine Combinations

- An affine combination of points $\mathbf{p_0}, \mathbf{p_1}, \ldots, \mathbf{p_n} \in E^3$ is a point $\mathbf{p} \in E^3$:

$$\mathbf{p} = \sum_{i=0}^{n} a_i \ \mathbf{p_i}$$

$a_0, a_1, \ldots, a_n \in \mathbb{R}$: affine coordinates of $\mathbf{p}$ with respect to $\mathbf{p_0}, \mathbf{p_1}, \ldots, \mathbf{p_n}$

where $\sum_{i=0}^{n} a_i = 1$

- Convex affine combination:
  - if all $a_i \geq 0$, $i = 0,1,\ldots,n$
  - The affine combination $\mathbf{p}$ is within the convex hull of the original points $\mathbf{p_0}, \ldots, \mathbf{p_n}$
  - E.g.1: Line segment between points $\mathbf{p_1}$ and $\mathbf{p_2}$ is the set of points $\mathbf{p}$:

    $\mathbf{p} = a_1 \cdot \mathbf{p_1} + a_2 \cdot \mathbf{p_2}$ with $0 \leq a_1 \leq 1$ and $a_2 = 1 - a_1$
  - E.g. 2: Triangle with vertices $\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}$ is the set of points p:

    $\mathbf{p} = a_1 \cdot \mathbf{p_1} + a_2 \cdot \mathbf{p_2} + a_3 \cdot \mathbf{p_3}$ with $0 \leq a_1, a_2, a_3 \leq 1$ and $a_1 + a_2 + a_3 = 1$

Affine Transformations

- Affine transformation:
  - transformation which preserves affine combinations
  - it retains the inter-relationship of the points
- A transformation $\Phi : E^3 \rightarrow E^3$ is affine if $\Phi(\mathbf{p}) = \sum_{i=0}^{n} a_i \Phi(\mathbf{p_i})$

  where $\mathbf{p} = \sum_{i=0}^{n} a_i \ \mathbf{p_i}$: an affine combination

- The result of the application of an affine transformation onto the result **p** of an affine combination *should equal* the affine combination of the result of performing the affine transformation on the defining points with the same weights a_i
- E.g

$$\text{midpoint of a line segment} \xrightarrow[\text{transformation}]{\text{affine}} \text{midpoint of the transformed line segment}$$

Affine Transformations (2)

- Practical consequence:
  - Internal points need **not** be transformed
  - It suffices to transform the defining points
- E.g. to perform an affine transformation on a triangle:
  - Transform its three vertices only, not its (infinite) interior points

General affine transformation

Mappings of the form $\Phi(\mathbf{p}) = \mathbf{A}\cdot\mathbf{p} + \vec{\mathbf{t}}$ (1) where $\mathbf{A}$ is a 3×3 matrix

$\vec{\mathbf{t}}$ is a 3×1 matrix

are affine transformations in E^3.

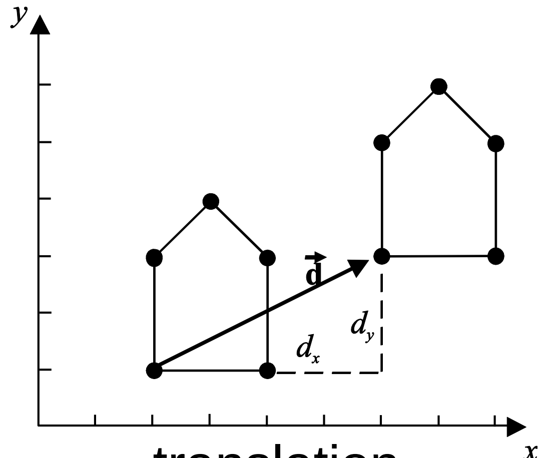
Proof: we shall show that (1) preserves affine combinations

$$\Phi\left(\sum_{i=0}^{n} a_i \mathbf{p_i}\right) = \mathbf{A}\left(\sum_{i=0}^{n} a_i \mathbf{p_i}\right) + \vec{\mathbf{t}} = \sum_{i=0}^{n} a_i \mathbf{A}\mathbf{p_i} + \sum_{i=0}^{n} a_i \vec{\mathbf{t}}$$

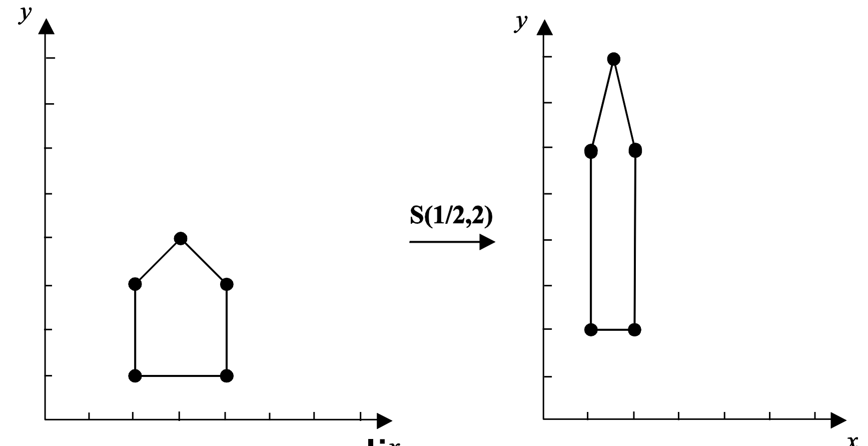
$$= \sum_{i=0}^{n} a_i (\mathbf{A}\,\mathbf{p_i} + \vec{\mathbf{t}}) = \sum_{i=0}^{n} a_i \Phi(\mathbf{p_i}).$$

2D Affine Transformations

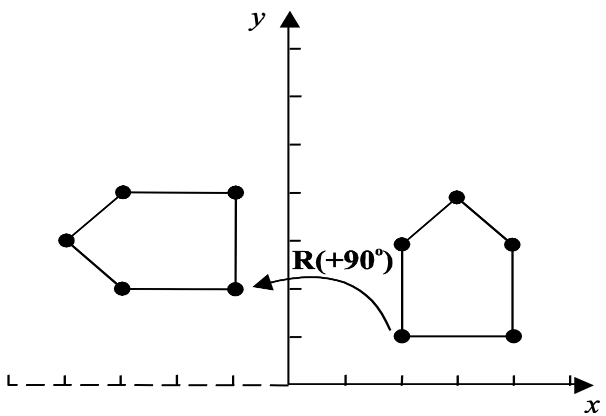
- 2D results generalize to 3D
- Four basic affine transformations:



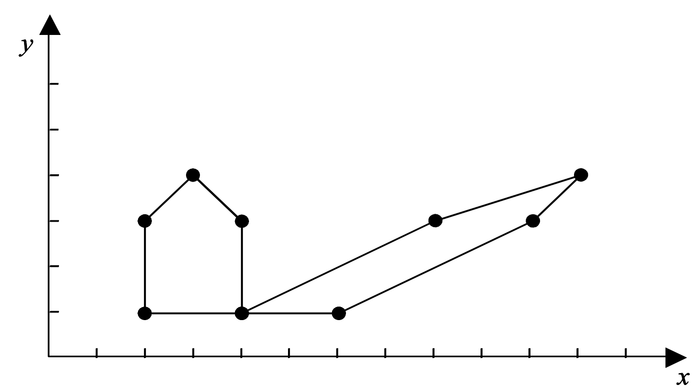
translation



scaling



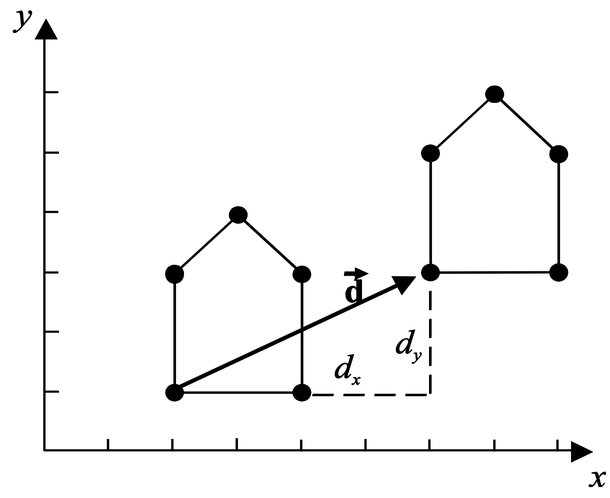
rotation



shearing

2D Translation

- Defines a movement by a certain distance in a certain direction, both specified by the translation vector
- The translation of a 2D point $\mathbf{p}=[x,y]^T$ by a vector $\vec{\mathbf{d}}=[d_x, d_y]^T$ gives another point $\mathbf{p}'=[x',y']^T$:



$$\mathbf{p}' = \mathbf{p} + \vec{\mathbf{d}}$$

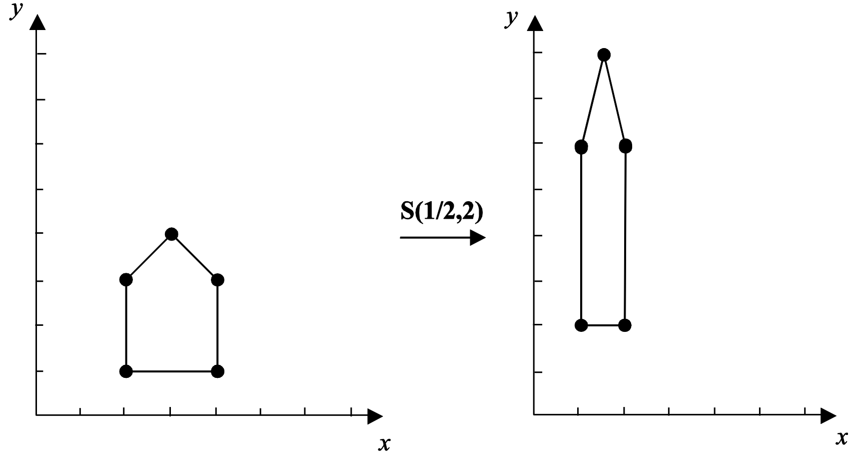
- Instantiation of $\Phi(\mathbf{p}) = \mathbf{A}\cdot\mathbf{p} + \vec{\mathbf{t}}$ where $\mathbf{A} = \mathbf{I}$ and $\vec{\mathbf{t}} = \vec{\mathbf{d}}$ (**I** is the 2×2 identity matrix)

2D Scaling

- Changes the size of objects
- Scaling factor: specifies the change in each direction
- 2D: s_x & s_y are the scaling factors which are multiplied by the respective coordinates of **p**=[x, y]^T
- Scaling of 2D point **p**=[x, y]^T to give **p'**=[x', y']^T :

$$\mathbf{p}' = \mathbf{S}(s_x, s_y)\cdot \mathbf{p} \text{ where } \mathbf{S}(s_x, s_y) = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

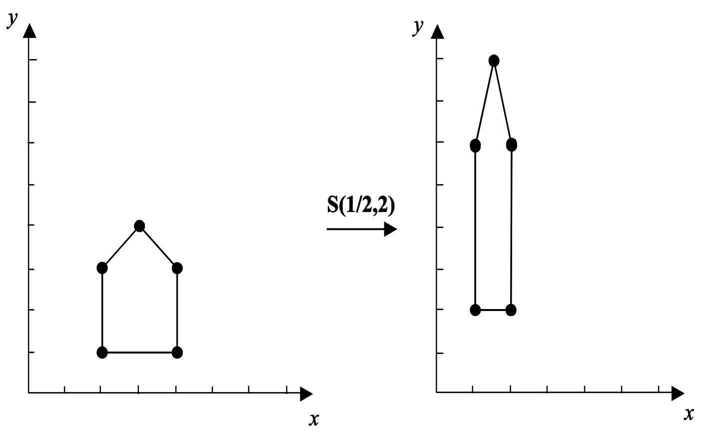
- Instantiation of $\Phi(\mathbf{p}) = \mathbf{A}\cdot\mathbf{p} + \vec{\mathbf{t}}$ where $\mathbf{A} = \mathbf{S}(s_x, s_y)$ and $\vec{\mathbf{t}} = \vec{\mathbf{0}}$



2D Scaling (2)

- Not possible to observe the effect on a single point
- If scaling factor < 1 → <u>reduce</u> the object's size

  scaling factor > 1 → <u>increase</u> the object's size
- Translation side-effect (proportional to the scaling factor):
  - The object has moved toward the origin of the x-axis ($s_x < 1$)
  - The object has moved away from the origin of the y-axis ($s_y > 1$)
- Isotropic scaling:
  - If all the scaling factors are equal (in 2D $s_x = s_y$)
  - Preserves the similarity of objects (angles)
- Mirroring:
  - Special case, using a -1 scaling factor
  - About x-axis: **S**(1,-1) and y-axis: **S**(-1,1)



2D Rotation

- Turns the objects about the origin
- The *distance* from the origin does *not* change, only the *orientation* changes
- Counterclockwise rotation is positive
- We can estimate $\mathbf{p}' = [x', y']^T$ from $\mathbf{p} = [x, y]^T$:

$$x' = l\cos(\phi+\theta) = l(\cos\phi \ \cos\theta - \sin\phi \ \sin\theta) = x \ \cos\theta - y \ \sin\theta$$

$$y' = l\sin(\phi+\theta) = l(\cos\phi \ \sin\theta + \sin\phi \ \cos\theta) = x \ \sin\theta + y \ \cos\theta$$

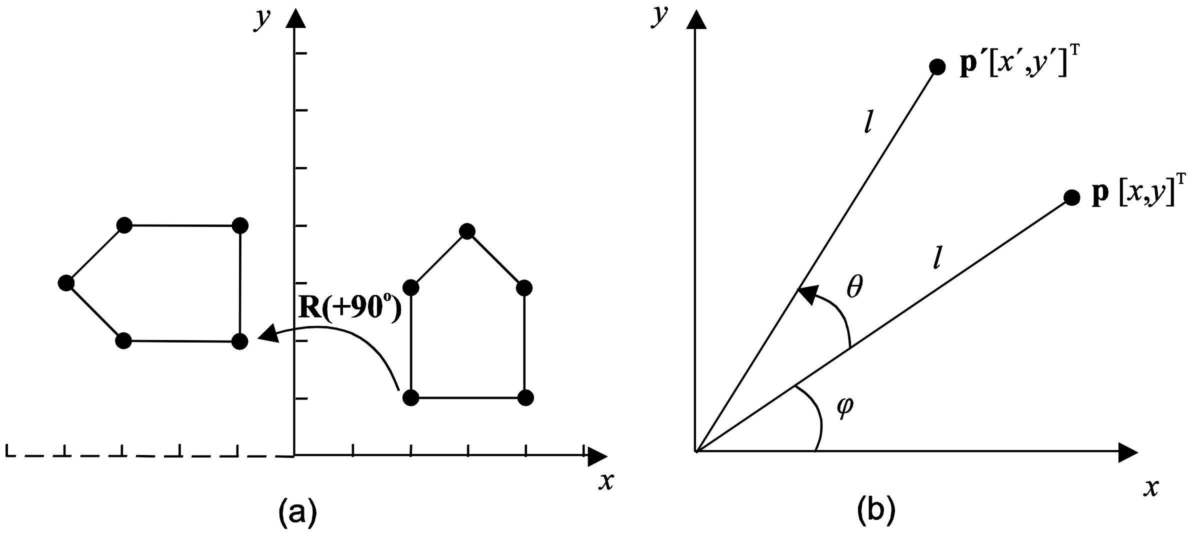
Thus:

$$\mathbf{p}' = \mathbf{R}(\theta) \cdot \mathbf{p}$$

2D Rotation (2)

- where:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

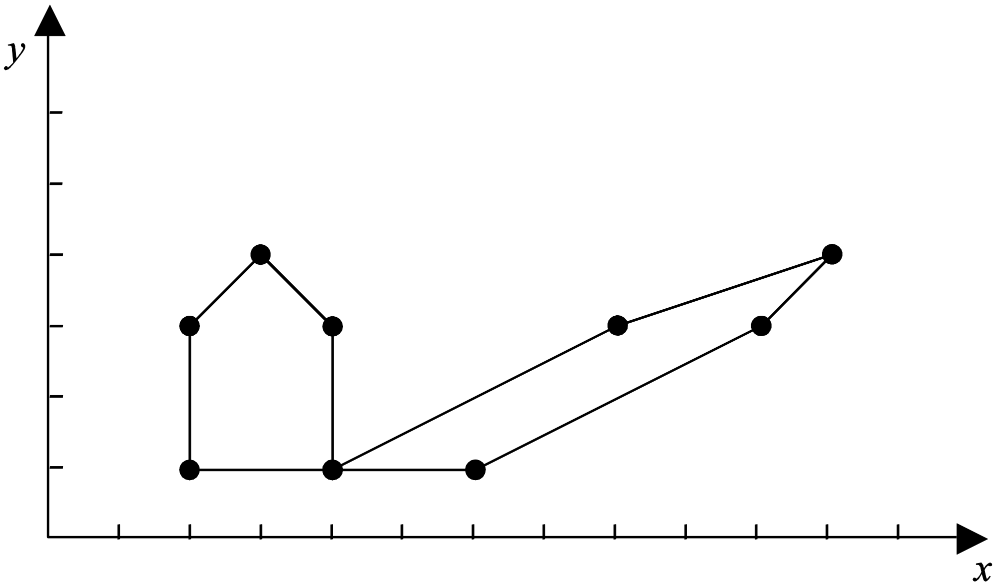


- Rotation is an instantiation of the general affine transformation

$\Phi(\mathbf{p}) = \mathbf{A}\cdot\mathbf{p} + \vec{\mathbf{t}}$ where $A = \mathbf{R}(\theta)$ and $\vec{\mathbf{t}} = \vec{\mathbf{0}}$

2D Shear

- Increases one of the object's coordinates by an amount equal to the other coordinate times a shearing factor
- Physical example: cards placed flat on a table and then tilted by a hard book.



2D Shear (2)

- We can estimate $\mathbf{p}' = [x', y']^T$ from $\mathbf{p} = [x, y]^T$:

  shear along x axis → $x' = x + ay, \ \ y' = y$

  shear along y axis → $x' = x, \ \ y' = bx + y$

  where α and b are the respective shear factors
- In matrix form:

  shear on x axis → $\mathbf{p}' = \mathbf{SH_x}(a) \cdot \mathbf{p}$, where $\mathbf{SH_x}(a) = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}$

  shear on y axis → $\mathbf{p}' = \mathbf{SH_y}(b) \cdot \mathbf{p}$, where $\mathbf{SH_y}(b) = \begin{bmatrix} 1 & 0 \\ b & 1 \end{bmatrix}$

- Shear is an instantiation of the general affine transformation

  $\Phi(\mathbf{p}) = \mathbf{A} \cdot \mathbf{p} + \vec{\mathbf{t}}$ where $\mathbf{A} = \mathbf{SH_x}(a)$ or $\mathbf{A} = \mathbf{SH_y}(b)$ and $\vec{\mathbf{t}} = \vec{\mathbf{0}}$

Composite Transformations

- Useful transformations in computer graphics and visualization rarely consist of a single basic affine transformation
- All transformations must be applied to all objects of a scene
- Objects are defined by thousands or even millions of vertices
- EXAMPLE: Rotate a 2D object by 45^o and then isotropically scale it by a factor of 2
  - Apply the rotation matrix:

$$\mathbf{R}(45^\circ) = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$$

  - Then apply the scaling matrix:

$$\mathbf{S}(2,2) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Composite Transformations (2)

- It is possible to apply the matrices sequentially to every vertex **p**:
  S(2, 2) (**R**(45^o) **p**) → NOT EFFICIENT
- Another way is to exploit the *associative property* of matrix multiplication and apply the pre-computed composite to the vertices: (**S**(2, 2) **R**(45^o) **)p** → MORE EFFICIENT
- Thus the composite transformation is only computed once and the composite matrix is applied to the vertices
- Matrix multiplication is not *commutative* → the order of multiplying the transformation matrices is important
- Having chosen the column representation of points → transformation matrices are right-multiplied by the points→ write the matrix composition in the reverse of the order of application

Composite Transformations (3)

- To apply the sequence of transformations **T1,T2, ...,Tm,** we compute the composite matrix **Tm ·... ·T2 ·T1**

- Problem with the translation transformation: Translation cannot be described by a linear transformation matrix such as:

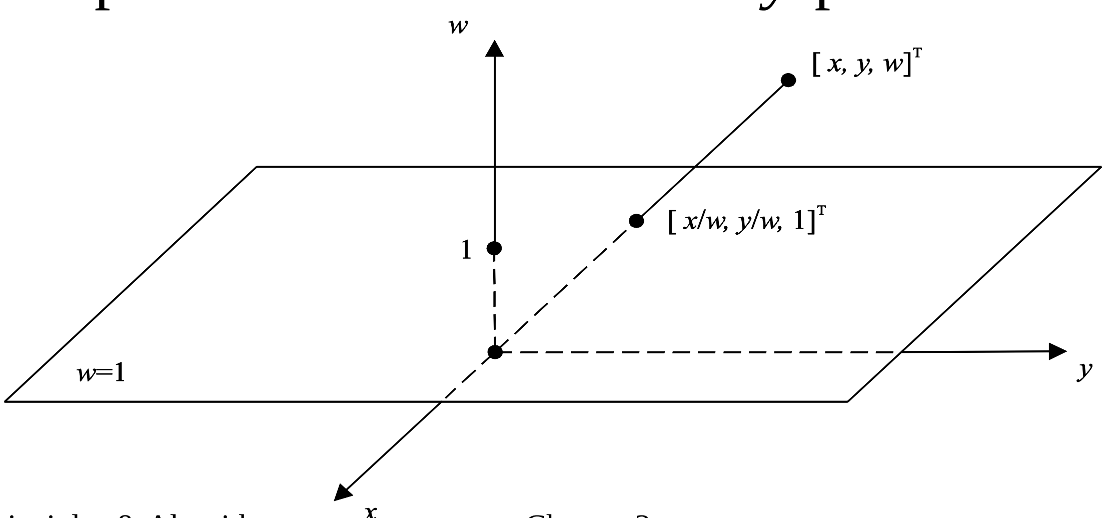
$$x' = ax + by$$

$$y' = cx + dy$$

- Thus translation cannot be included in a composite transformation
- Solution to the problem → *homogeneous coordinates*

Homogeneous Coordinates

- Homogeneous Coordinates use one additional dimension than the space we want to represent
- 2D space: $\begin{bmatrix} x \\ y \\ w \end{bmatrix}$, where w is the new coordinate that corresponds to the extra dimension; $w \neq 0$
- Fixing w=1 maintains our original dimensionality by taking slice w=1
- In 2D we use the plane w=1 instead of the xy-plane



Homogeneous Coordinates (2)

- Points whose homogeneous coordinates are multiples of each other are equivalent:

  e.g., $[1,2,3]^T$ and $[2,4,6]^T$ represent the same point

- The actual point that they represent is given by their unique *basic representation*, which has $w = 1$ and is obtained by dividing all coordinates by w:

  $[x/w, y/w, w/w]^T = [x/w, y/w, 1]^T$

- In general, we use the basic representation for points, and we ensure that our transformations preserve this property

Homogeneous Coordinates (3)

- How do homogeneous coordinates treat the translation problem?
- Exploit the fact that points have w = 1, in order to represent the translation of a point **p** = [x, y, w]^T by a vector $\vec{\mathbf{d}} = [d_x, d_y]$, as a linear transformation:

$$x' = 1x + 0y + d_x w = x + d_x$$

$$y' = 0x + 1y + d_y w = y + d_y$$

$$w' = 0x + 0y + 1w = 1$$

- Transformation on the w-coordinate ensures that the resulting point $\mathbf{p}' = [x', y', w']^T$ has $w' = 1$

Homogeneous Coordinates (4)

- The above linear expressions can be encapsulated in matrix form, thus treating translation in the same way as the other basic affine transformations
- In non-homogeneous transformations, the origin $[0, 0]^T$ is a *fixed point*:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- A positive effect of homogeneous coordinates is that there is no fixed point under homogeneous affine transformations

Homogeneous Coordinates (4)

- The 2D origin is now $[0, 0, 1]^T$ which is not a fixed point
- The point $[0, 0, 0]^T$ is outside w=1 plane → disallowed since it has w=0
- From here on points will be represented by their homogeneous coordinates:

$$2D \rightarrow [x, y, 1]^T$$

$$3D \rightarrow [x, y, z, 1]^T$$

2D Homogeneous Affine Transformations

- 2D homogeneous translation matrix:

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

- Translation is treated like the other basic affine transformations:

$$\mathbf{p}' = \mathbf{T}(\vec{\mathbf{d}}) \cdot \mathbf{p}$$

- The last row of a homogeneous transformation matrix is always [0, 0, 1] in order to preserve the unit value of the *w*-coordinate
- 2D homogeneous scaling matrix:

$$\mathbf{S}(s_x, s_y) = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Affine Transformations (2)

- 2D homogeneous rotation matrix:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- 2D homogeneous shear matrices:

shear on x axis →

$$\mathbf{SH_x}(a) = \begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

shear on y axis →

$$\mathbf{SH_y}(b) = \begin{bmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Inverse Transformations

- Often necessary to reverse a transformation

- 2D inverse homogeneous translation matrix:

$$\mathbf{T}^{\textbf{-1}}(\vec{\mathbf{d}}) = \mathbf{T}(\textbf{-}\vec{\mathbf{d}}) = \begin{bmatrix} 1 & 0 & -d_x \\ 0 & 1 & -d_y \\ 0 & 0 & 1 \end{bmatrix}$$

- 2D inverse homogeneous scaling matrix:

$$\mathbf{S}^{\textbf{-1}}(s_x, s_y) = \mathbf{S}(\frac{1}{s_x}, \frac{1}{s_y}) = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Inverse Transformations (2)

- 2D inverse homogeneous rotation matrix:

$$\mathbf{R}^{\mathbf{-1}}(\theta) = \mathbf{R}(-\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- 2D inverse homogeneous shear matrix:

shear on x axis → $\mathbf{SH}_{\mathbf{x}}^{\mathbf{-1}}(a) = \mathbf{SH}_{\mathbf{x}}(-a) = \begin{bmatrix} 1 & -a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

shear on y axis → $\mathbf{SH}_{\mathbf{y}}^{\mathbf{-1}}(b) = \mathbf{SH}_{\mathbf{y}}(-b) = \begin{bmatrix} 1 & 0 & 0 \\ -b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

2D Homogeneous Inverse Transformations

- Applying an object transformation on an object is equivalent to the application of the inverse transformation on the coordinate system (*axis transformation*)

EXAMPLE:

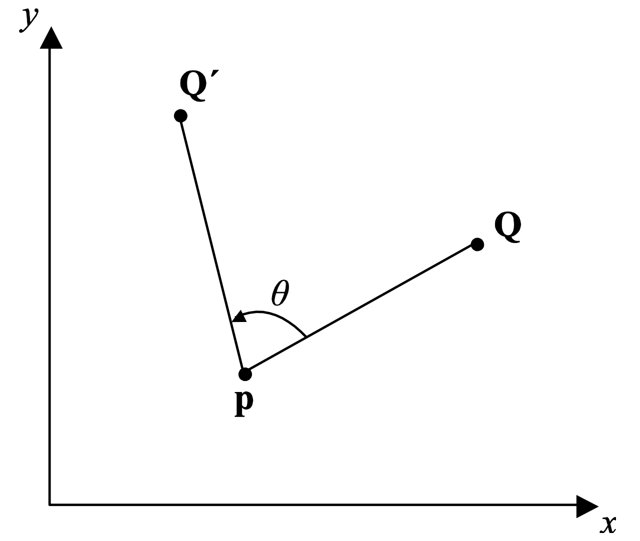
- Isotropically scaling an object by a factor of 2 is equivalent to isotropically scaling the coordinate system axis by a factor of 1/2 (*shrinking*)

Properties of Homogeneous Matrices

Some properties of homogeneous affine transformation matrices:

- → $\mathbf{T}(\overrightarrow{\mathbf{d1}})\cdot\mathbf{T}(\overrightarrow{\mathbf{d2}}) = \mathbf{T}(\overrightarrow{\mathbf{d2}})\cdot\mathbf{T}(\overrightarrow{\mathbf{d1}}) = \mathbf{T}(\overrightarrow{\mathbf{d1}} + \overrightarrow{\mathbf{d2}})$
- → $\mathbf{S}(s_{x1}, s_{y1})\cdot\mathbf{S}(s_{x2}, s_{y2}) = \mathbf{S}(s_{x2}, s_{y2})\cdot\mathbf{S}(s_{x1}, s_{y1}) = \mathbf{S}(s_{x1}\cdot s_{x2}, s_{y1}\cdot s_{y2})$
- → $\mathbf{R}(\theta 1)\cdot\mathbf{R}(\theta 2) = \mathbf{R}(\theta 2)\cdot\mathbf{R}(\theta 1) = \mathbf{R}(\theta 1 + \theta 2)$
- → $\mathbf{S}(s_x, s_y)\cdot\mathbf{R}(\theta) = \mathbf{R}(\theta)\cdot\mathbf{S}(s_x, s_y)$ for isotropic scaling only ($s_x = s_y$)

2D Transformation Example 1



EXAMPLE 1: Rotation about an arbitrary point

Determine the transformation matrix R(θ, p) required to perform rotation about an arbitrary point p by an angle θ

SOLUTION

- Step 1: Translate by $-\vec{\mathbf{p}}, \mathbf{T}(-\vec{\mathbf{p}})$
- Step 2: Rotate by θ, **R**(θ)
- Translate by $\vec{\mathbf{p}}, \mathbf{T}(\vec{\mathbf{p}})$

$$\mathbf{R}(\theta, p) = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -p_x \\ 0 & 1 & -p_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & p_x - p_x\cos\theta + p_y\sin\theta \\ \sin\theta & \cos\theta & p_y - p_x\sin\theta - p_y\cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$

2D Transformation Example 2

EXAMPLE 2: Rotation of a triangle about a point

Rotate the triangle $\triangle abc$ by 45° about the point **p**=[-1,-1]^T, where **a**=[0,0]T, **b**=[1,1]^T and **c**=[5,2]^T

SOLUTION

- The triangle can by represented by the matrix $\mathbf{T} = \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$
- We shall apply **R**(θ, **p**) [Ex. 1] to the triangle:

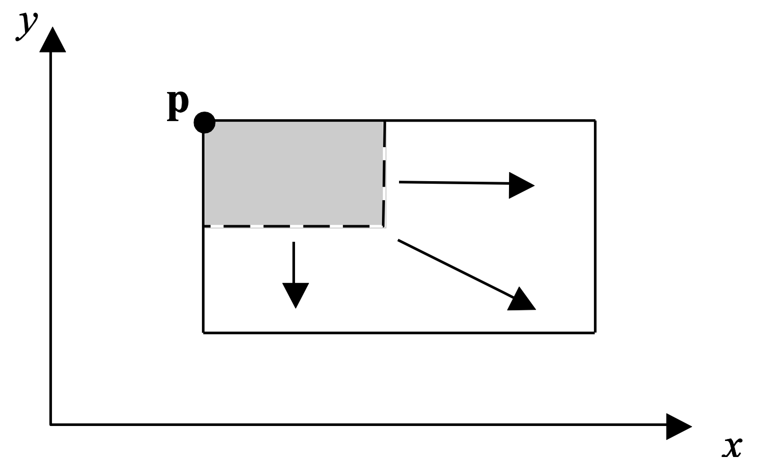
$$\mathbf{R}(45^\circ, [-1,-1]^T)\cdot \mathbf{T} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & -1 \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \sqrt{2}-1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -1 & \frac{3}{2}\sqrt{2}-1 \\ \sqrt{2}-1 & 2\sqrt{2}-1 & \frac{9}{2}\sqrt{2}-1 \\ 1 & 1 & 1 \end{bmatrix}$$

- The rotated triangle is $\triangle a'b'c'$ where **a'**=[-1, $\sqrt{2}-1$]^T, **b'**=[-1, $2\sqrt{2}-1$]^T and **c'**=[$\frac{3}{2}\sqrt{2}-1, \frac{9}{2}\sqrt{2}-1$]^T

2D Transformation Example 3

EXAMPLE 3: Scaling about an arbitrary point

Determine the transformation matrix $\mathbf{S}(s_x, s_y, \mathbf{p})$ required to perform scaling by s_x and s_y about an arbitrary point p



SOLUTION

- Step 1: Translate by $-\vec{\mathbf{p}}, \mathbf{T}(-\vec{\mathbf{p}})$
- Step 2: Scale by s_x and s_y , $\mathbf{S}(s_x, s_y)$
- Step 3: Translate by $\vec{\mathbf{p}}, \mathbf{T}(\vec{\mathbf{p}})$ to undo the initial translation

$$\mathbf{S}(s_x, s_y, p) = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -p_x \\ 0 & 1 & -p_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & p_x - p_x s_x \\ 0 & s_y & p_y - p_y s_y \\ 0 & 0 & 1 \end{bmatrix}$$

2D Transformation Example 4

EXAMPLE 4: Scaling of a triangle about a point

Double the lengths of the sides of triangle $\triangle abc$ keeping its vertex **c** fixed . The coordinates of its vertices are **a**=[0,0]T, **b**=$[1,1]^T$, **c**=$[5,2]^T$

SOLUTION

- The triangle can by represented by the matrix **T** [Ex. 2]
- We shall apply the matrix **S**(s_x,s_y,**p**) [Ex. 3] to the triangle, setting the scaling factor equal to 2 and **p=c**

$$\mathbf{S}(2,2,[5,2,1]^T)\cdot\mathbf{T} = \begin{bmatrix} 2 & 0 & -5 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -5 & -3 & 5 \\ -2 & 0 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

- The scaled triangle is $\triangle a'b'c'$ where **a'**=$[-5,-2]^T$, **b'**=$[-3,0]^T$ and **c'**=$[5,2]^T$

2D Transformation Example 5

EXAMPLE 5: Axis transformation

Suppose that the coordinate system is translated by the vector $\vec{\mathbf{v}} = [v_x, v_y]^T$.

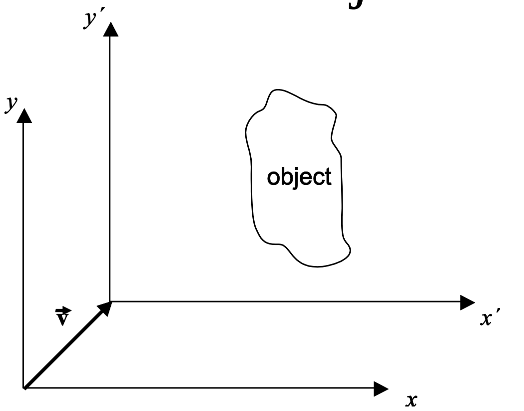
Determine the matrix that describes this effect

SOLUTION

- The required transformation matrix must produce the coordinates of the objects with respect to the new coordinate system. This is achieved by applying the inverse translation to the objects:

$$\mathbf{T}(-\vec{\mathbf{v}}) = \begin{bmatrix} 1 & 0 & -v_x \\ 0 & 1 & -v_y \\ 0 & 0 & 1 \end{bmatrix}$$

⇨



- Similar argument holds for any other axis transformation. Its effect is encapsulated by applying the inverse transformation to the objects

2D Transformation Example 6

<u>EXAMPLE 6: Mirroring about an arbitrary axis</u>

Determine the transformation matrix required to perform mirroring about an axis specified by a point $\mathbf{p}=[p_x,p_y]^T$ and a direction vector $\vec{\mathbf{v}}=[v_x,v_y]^T$

<u>SOLUTION</u>

- Step 1: Translate by $-\vec{\mathbf{p}}, \mathbf{T}(-\vec{\mathbf{p}})$
- Step 2: Rotate by –θ (negative as it is clockwise), **R**(-θ)

θ forms between x-axis and vector $\vec{\mathbf{v}}$ and:

$$\sin\theta = \frac{v_y}{\sqrt{v_x^2+v_y^2}} \qquad \cos\theta = \frac{v_x}{\sqrt{v_x^2+v_y^2}}$$

The two previous steps make the general axis coincide with the x-axis

- Step 3: Perform mirroring about the x-axis, **S**(1, -1)
- Step 4: Rotate by θ, **R**(θ)

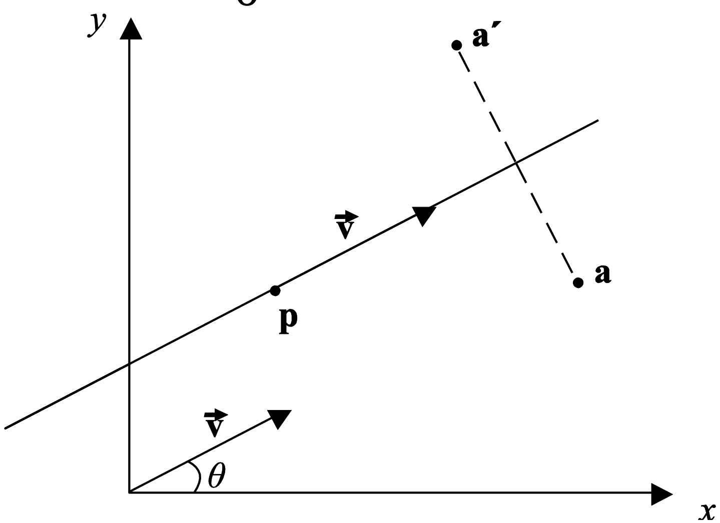
2D Transformation Example 6 (2)

- Step 5: Translate by $\vec{\mathbf{p}}, \mathbf{T}(\vec{\mathbf{p}})$

Summarizing the previous steps we have:

$$\mathbf{M}_{\mathbf{SYM}} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -p_x \\ 0 & 1 & -p_y \\ 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} \cos^2\theta - \sin^2\theta & 2\sin\theta\cos\theta & p_x - p_x(\cos^2\theta - \sin^2\theta) - 2p_y\sin\theta\cos\theta \\ 2\sin\theta\cos\theta & \sin^2\theta - \cos^2\theta & p_y - p_y(\sin^2\theta - \cos^2\theta) - 2p_x\sin\theta\cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$



2D Transformation Example 7

EXAMPLE 7: Mirror polygon

Given a polygon, determine its mirror polygon with respect to (a) the line y=2 and (b) the axis specified by the point **p**=$[0,2]^T$ and the vector $\vec{\mathbf{v}}=[1,1]^T$ The polygon is given by its vertices **a**=$[-1,0]^T$, **b**=$[0,-2]^T$, **c**=$[1,0]^T$ and **d**=$[0,2]^T$

SOLUTION

- The polygon can be represented by matrix $\mathbf{\Pi}=\begin{bmatrix}-1 & 0 & 1 & 0\\ 0 & -2 & 0 & 2\\ 1 & 1 & 1 & 1\end{bmatrix}$
- In case (a) **p**=$[0,2]^T$ and $\vec{\mathbf{v}}=[1,0]^T$ thus $\theta=0^o$, $\sin\theta=0$, $\cos\theta=1$ and we have:

$$\Pi' = \mathbf{M}_{\mathbf{SYM}}\cdot\mathbf{\Pi} = \begin{bmatrix}1 & 0 & 0\\ 0 & -1 & 4\\ 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}-1 & 0 & 1 & 0\\ 0 & -2 & 0 & 2\\ 1 & 1 & 1 & 1\end{bmatrix} = \begin{bmatrix}-1 & 0 & 1 & 0\\ 4 & 6 & 4 & 2\\ 1 & 1 & 1 & 1\end{bmatrix}$$

2D Transformation Example 7 (2)

- In case (b) $\mathbf{p}=[0,2]^T$ and $\vec{\mathbf{v}}=[1,1]^T$ thus $\sin\theta = \cos\theta = \dfrac{1}{\sqrt{2}}$

  and we have:

$$\Pi' = \mathbf{M}_{\mathbf{SYM}} \cdot \mathbf{\Pi} = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -2 & 0 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -2 & -4 & -2 & 0 \\ 1 & 2 & 3 & 2 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

where $\mathbf{M}_{\mathbf{SYM}}$ is the matrix of [Ex. 6]

2D Transformation Example 8

EXAMPLE 8: Window-to-Viewport transformation

The contents of a 2D window must be transferred to a 2D "viewport". Both the window and "viewport" are rectangular parallelograms with sides parallel to the x- and y-axis. Determine the window to viewport transformation matrix. Also determine how objects are deformed by this transformation.

SOLUTION

Suppose that the window and the viewport are defined by two opposite vertices $[w_{xmin}, w_{ymin}]^T, [w_{xmax}, w_{ymax}]^T$ and $[v_{xmin}, v_{ymin}]^T, [v_{xmax}, v_{ymax}]^T$

- Step 1: Translate $[w_{xmin}, w_{ymin}]^T$ to the origin, using $\mathbf{T}(-\vec{\mathbf{w}}_{\mathbf{min}})$ where

$$\vec{\mathbf{w}}_{\mathbf{min}} = [w_{xmin}, w_{ymin}]^T$$

2D Transformation Example 8 (2)

- Step 2: Scale the window to the size of the viewport, using $\mathbf{S}(s_x, s_y)$ where

$$s_x = \frac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}} \qquad s_y = \frac{v_{ymax} - v_{ymin}}{w_{ymax} - w_{ymin}}$$

- Step 3: Translate to the minimum viewport vertex $[v_{xmin}, v_{ymin}]^T$, using $\mathbf{T}(\vec{\mathbf{v}}_{\mathbf{min}})$ where $\vec{\mathbf{v}}_{\mathbf{min}} = [v_{xmin}, v_{ymin}]^T$

  Summarizing the previous steps we have:

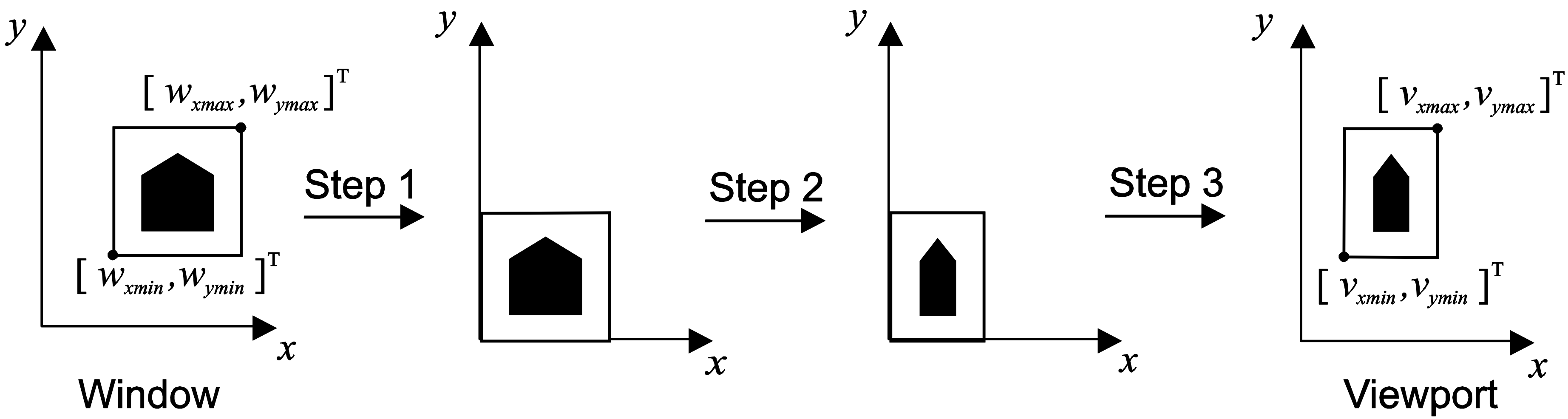
$$\mathbf{M}_{\mathbf{WV}} = \mathbf{T}(\vec{\mathbf{v}}_{\mathbf{min}}) \cdot \mathbf{S}(s_x, s_y) \cdot \mathbf{T}(-\vec{\mathbf{w}}_{\mathbf{min}})$$

$$= \begin{bmatrix} 1 & 0 & v_{xmin} \\ 0 & 1 & v_{ymin} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}} & 0 & 0 \\ 0 & \frac{v_{ymax} - v_{ymin}}{w_{ymax} - w_{ymin}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -w_{xmin} \\ 0 & 1 & -w_{ymin} \\ 0 & 0 & 1 \end{bmatrix}$$

2D Transformation Example 8 (3)

Finally :

$$\mathbf{M}_{\mathbf{WV}} = \begin{bmatrix} \dfrac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}} & 0 & v_{xmin} - w_{xmin} \dfrac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}} \\ 0 & \dfrac{v_{ymax} - v_{ymin}}{w_{ymax} - w_{ymin}} & v_{ymin} - w_{ymin} \dfrac{v_{ymax} - v_{ymin}}{w_{ymax} - w_{ymin}} \\ 0 & 0 & 1 \end{bmatrix}$$



2D Transformation Example 8 (4)

- $\mathbf{M_{WV}}$ contains non-isotropic scaling ($s_x \neq s_y$) → objects will be deformed. A circle will become an ellipse and a square will become a rectangular parallelogram.
- The *aspect ratios* of the window and the viewport are defined as the ratios of their x- to their y-sizes:

$$a_w = \frac{w_{xmax} - w_{xmin}}{w_{ymax} - w_{ymin}}, a_v = \frac{v_{xmax} - v_{xmin}}{v_{ymax} - v_{ymin}}$$

- If $a_w \neq a_v$ then objects are deformed. To avoid this, is to use the largest part of the viewport with the same aspect ratio as the window. For example we can change the v_{xmax} or the v_{ymax} boundary of the viewport in the following manner:

if ($a_v > a_w$) then $v_{xmax} = v_{xmin} + a_w * (v_{ymax} - v_{ymin})$

else if ($a_v < a_w$) then $v_{ymax} = v_{ymin} + \frac{(v_{xmax} - v_{xmin})}{a_w}$

2D Transformation Example 9

EXAMPLE 9: Window-to-Viewport transformation instances

Determine the window-to-viewport transformation from the window:

$[w_{xmin}, w_{ymin}]^T = [1,1]^T, [w_{xmax}, w_{ymax}]^T = [3,5]^T$

to the viewport : $[v_{xmin}, v_{ymin}]^T = [0,0]^T, [v_{xmax}, v_{ymax}]^T = [1,1]^T$

If there is deformation, how can it be corrected?

SOLUTION

- Direct application of the $\mathbf{M_{WV}}$ [Ex. 8]

  For the window and viewport pair gives $\mathbf{M_{WV}} = \begin{bmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & \frac{1}{4} & -\frac{1}{4} \\ 0 & 0 & 1 \end{bmatrix}$

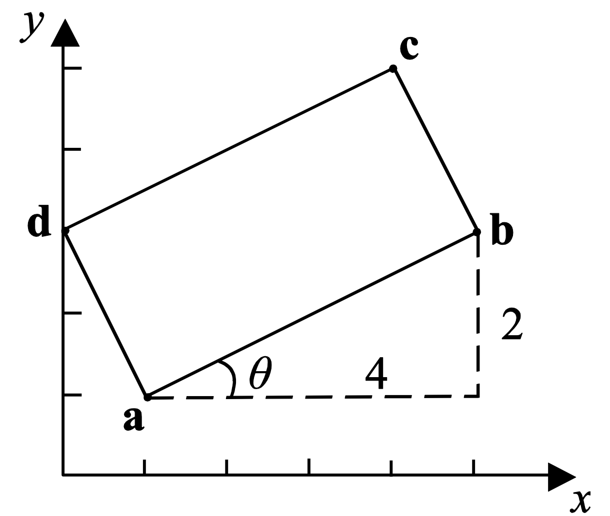
- Now $a_w = \frac{1}{2}$ and $a_v = \frac{1}{1}$, so there is distortion since $(a_v > a_w)$. It can be corrected by reducing the size of the viewport by setting:

$$v_{xmax} = v_{xmin} + a_w * (v_{ymax} - v_{ymin}) = \frac{1}{2}$$

2D Transformation Example 10

EXAMPLE 10: Tilted window-to-viewport transformation

Suppose that the window is tilted and given by its four vertices **a**=$[1,1]^T$, **b**=$[5,3]^T$, **c**=$[4,5]^T$ and **d**=$[0,3]^T$. Determine the transformation $\mathbf{M}_{\mathbf{WV}}^{\mathbf{TILT}}$ that maps it to the viewport $[v_{xmin}, v_{ymin}]^T = [0,0]^T, [v_{xmax}, v_{ymax}]^T = [1,1]^T$



SOLUTION

- Step 1: Rotate the window by angle –θ about a point **a**. For this we shall use matrix **R**(θ, **p**) [Ex. 1], instantiating it as **R**(-θ, **a**) where

$$\sin\theta = \frac{1}{\sqrt{5}} \qquad \cos\theta = \frac{2}{\sqrt{5}}$$

- Step 2: Apply the window to viewport transformation $\mathbf{M}_{\mathbf{WV}}$ [Ex. 8]

2D Transformation Example 10 (2)

Before Step 2 we must determine the maximum x- and y- coordinates of the rotated window by computing:

$$\mathbf{c}' = \mathbf{R}(-\theta, a)\cdot\mathbf{c} = \begin{bmatrix} 1+2\sqrt{5} \\ 1+\sqrt{5} \\ 1 \end{bmatrix}$$

Thus $[w_{xmin}, w_{ymin}]^T = \mathbf{a}, [w_{xmax}, w_{ymax}]^T = \mathbf{c}'$, and we have:

$$\mathbf{M}_{\mathbf{WV}}^{\mathbf{TILT}} = \mathbf{M}_{\mathbf{WV}}\cdot\mathbf{R}(-\theta, a) = \begin{bmatrix} \frac{1}{2\sqrt{5}} & 0 & -\frac{1}{2\sqrt{5}} \\ 0 & \frac{1}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 1-\frac{3}{\sqrt{5}} \\ -\frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} & 1-\frac{1}{\sqrt{5}} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{5} & \frac{1}{10} & -\frac{3}{10} \\ -\frac{1}{5} & \frac{2}{5} & -\frac{1}{5} \\ 0 & 0 & 1 \end{bmatrix}$$

3D Homogeneous Affine Transformations

- 3D homogeneous coordinates are similar to 2D
- Add an extra coordinate to create $[x, y, z, w]^T$ where w corresponds to the extra dimension
- Points whose homogeneous coordinates are multiples are equivalent e.g. $[1, 2, 3, 2]^T$ and $[2, 4, 6, 4]^T$
- Basic representation of a point
  - is unique
  - has w=1
  - is obtained by dividing by w : $[x/w, y/w, z/w, w/w]^T = [x/w, y/w, z/w, 1]^T$, $w \neq 0$

  Example: $$[\frac{1}{2},\frac{2}{2},\frac{3}{2},\frac{2}{2}]^T = [\frac{2}{4},\frac{4}{4},\frac{6}{4},\frac{4}{4}]^T = [\frac{1}{2},1,\frac{3}{2},1]^T$$
- Obtain a 3D projection of 4D space by setting w=1
- Points: 4×1 vectors. Transformations: 4×4 matrices

3D Homogeneous Translation

- Specified by a 3-dimensional vector $\vec{\mathbf{d}} = [d_x, d_y, d_z]^{\mathrm{T}}$
- Matrix form:

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

- $\mathbf{T}(\vec{\mathbf{d}})$ can be combined with other affine transformation matrices by matrix multiplication
- Inverse translation: $\mathbf{T}^{-1}(\vec{\mathbf{d}}) = \mathbf{T}(-\vec{\mathbf{d}})$

3D Homogeneous Scaling

- Three scaling factors: s_x, s_y, s_z
- If scaling factor < 1 → the object's size is <u>reduced</u> in the respective dimension
  scaling factor > 1 → the object's size is <u>increased</u>
- Matrix form:

$$\mathbf{S}(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Scaling has a translation side-effect, proportional to the scaling factor

3D Homogeneous Scaling (2)

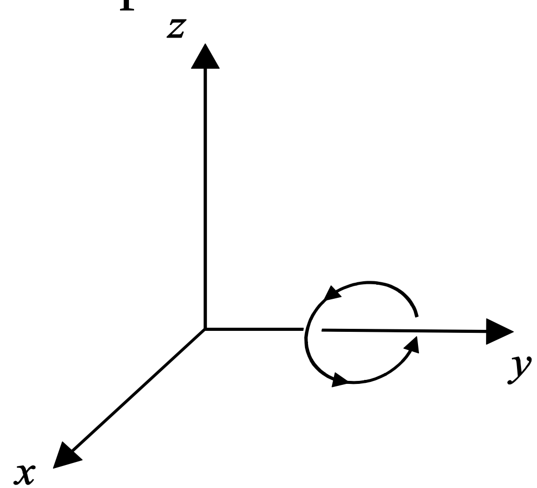
- Isotropic scaling:
  - if $s_x = s_y = s_z$
  - preserves the similarity of objects (angles)
- Mirroring:
  - about one of the major planes (xy, xz, yz)
  - using a -1 scaling factor
  - e.g. mirroring about the xy-plane is **S**(1, 1, -1)

- Inverse scaling: $\mathbf{S}^{-1}(s_x, s_y, s_z) = \mathbf{S}(\frac{1}{s_x}, \frac{1}{s_y}, \frac{1}{s_z})$

3D Homogeneous Rotation

- Different from 2D rotation: rotate about an **axis**, not a point
- Basic rotation transformations: rotate about the 3 main axes x, y, z
- Rotation about an arbitrary axis: by combining basic rotations
- Positive rotation about an axis α:

  in *right-handed* coordinate system is the one in the *counterclockwise* direction when looking from the positive part of an axis toward the origin



Positive rotation about y-axis

3D Homogeneous Rotation(2)

- The distance from the axis of rotation does not change
- Rotation does not affect the coordinate that corresponds to the axis of rotation
- Rotation matrices about the main axes:

$$\mathbf{R_x}(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R_y}(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R_z}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Inverse rotation: $\mathbf{R_x^{-1}}(\theta) = \mathbf{R_x}\ (-\theta),\ \mathbf{R_y^{-1}}(\theta) = \mathbf{R_y}(-\theta)\ \textit{and}\ \mathbf{R_z^{-1}}(\theta) = \mathbf{R_z}(-\theta),$
- Rotations can also be expressed using quaternions

3D Homogeneous Shear

- "Shears" object along one of the major planes
- Increases 2 coordinates by an amount equal to the 3rd coordinate times the respective shearing factors
- 3 cases in 3D shear: xy, xz, yz
- xy shear: increases x by $a \times z-\text{coordinate}$
  increases y by $b \times z-\text{coordinate}$

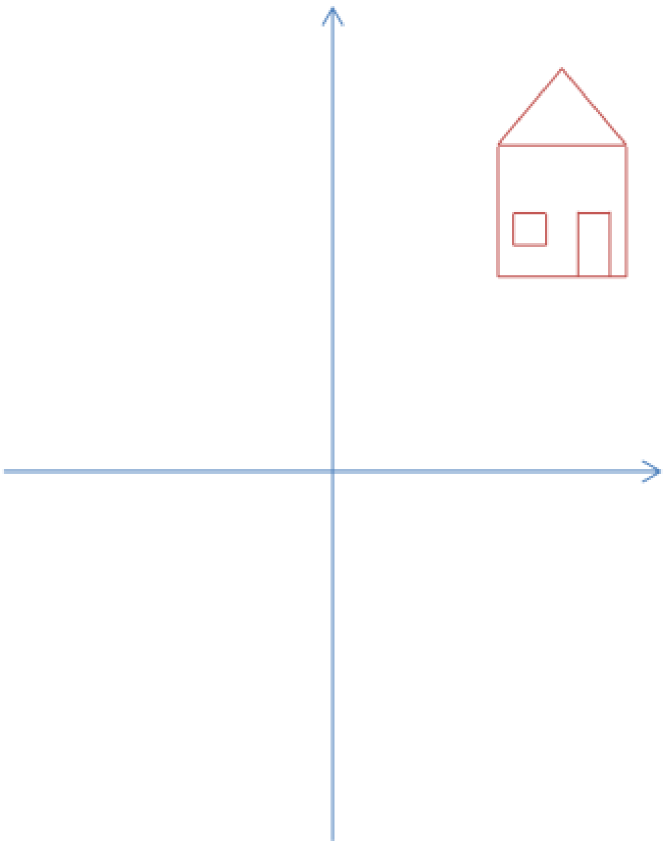
Similar for xz & yz :

$$\mathbf{SH}_{\mathbf{xy}}(a,b)=\begin{bmatrix}1 & 0 & a & 0\\ 0 & 1 & b & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \quad \mathbf{SH}_{\mathbf{xz}}(a,b)=\begin{bmatrix}1 & a & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & b & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \quad \mathbf{SH}_{\mathbf{yz}}(a,b)=\begin{bmatrix}1 & 0 & 0 & 0\\ a & 1 & 0 & 0\\ b & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

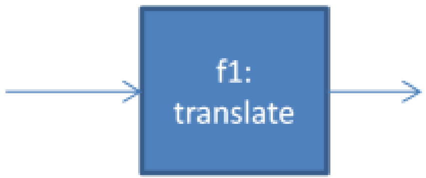
- Inverse shear:

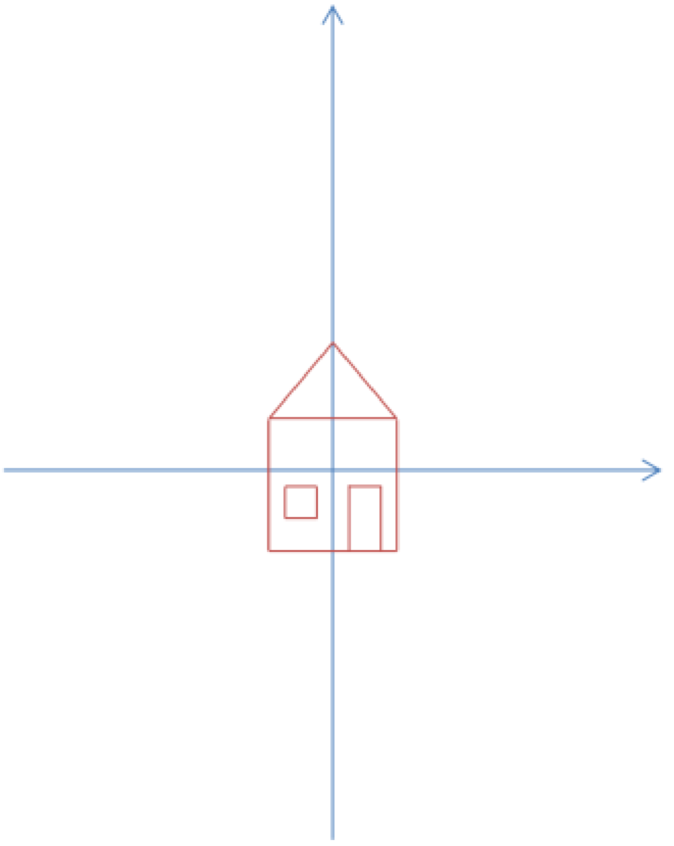
$$\mathbf{SH}^{-1}_{\mathbf{xy}}(a,b)=\mathbf{SH}_{\mathbf{xy}}(-a,-b),$$
$$\mathbf{SH}^{-1}_{\mathbf{xz}}(a,b)=\mathbf{SH}_{\mathbf{xz}}(-a,-b),$$
$$\mathbf{SH}^{-1}_{\mathbf{yz}}(a,b)=\mathbf{SH}_{\mathbf{yz}}(-a,-b)$$

Example Composite Transformation

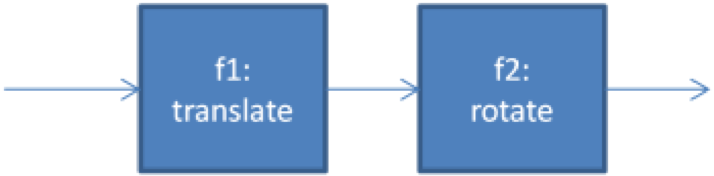


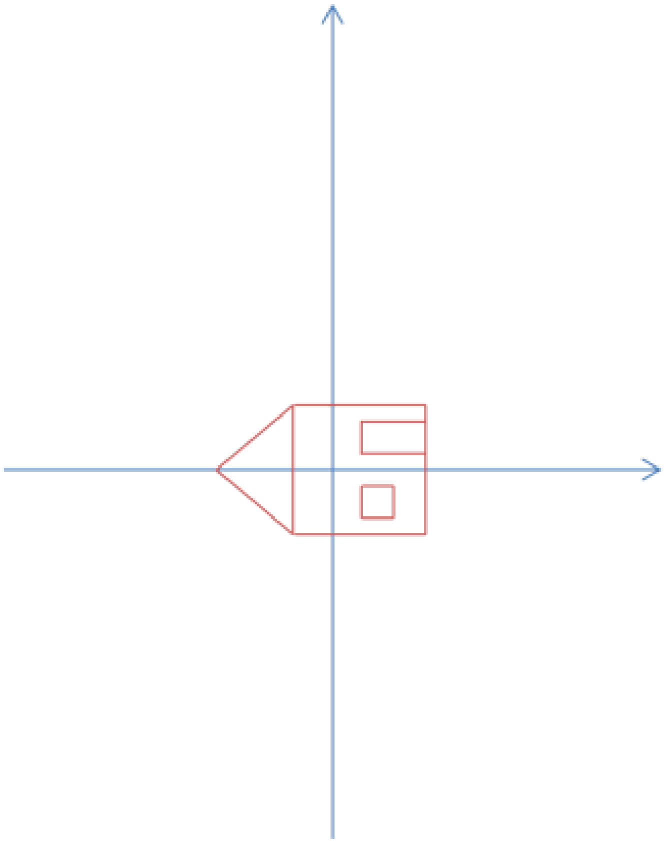
Example Composite Transformation (2)



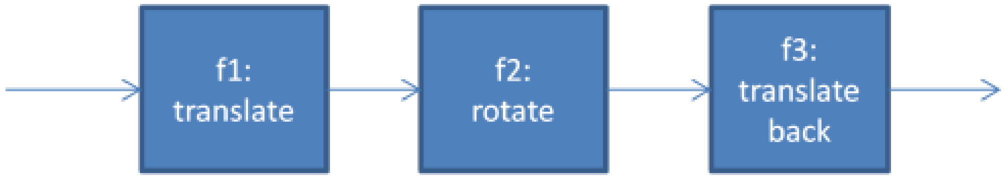


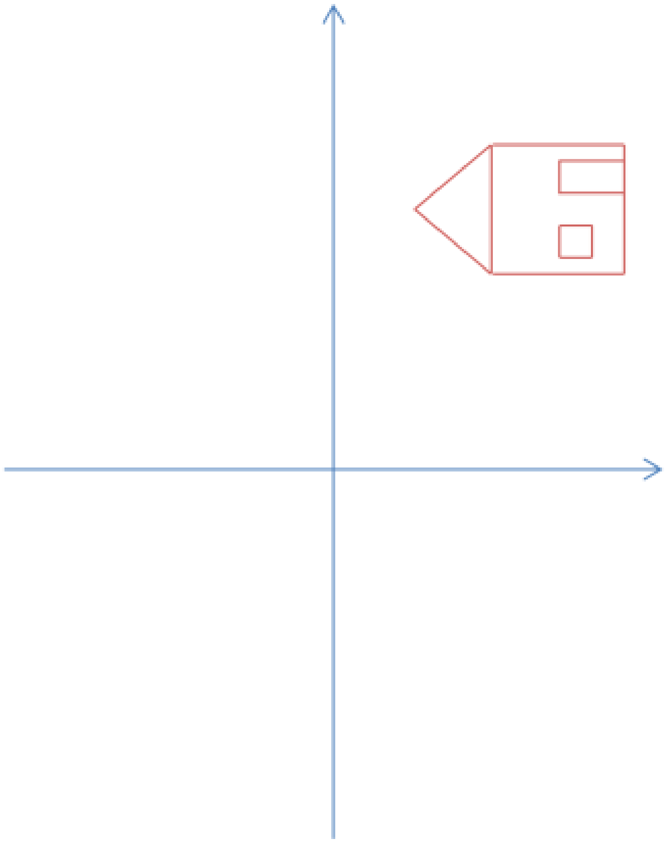
Example Composite Transformation (3)



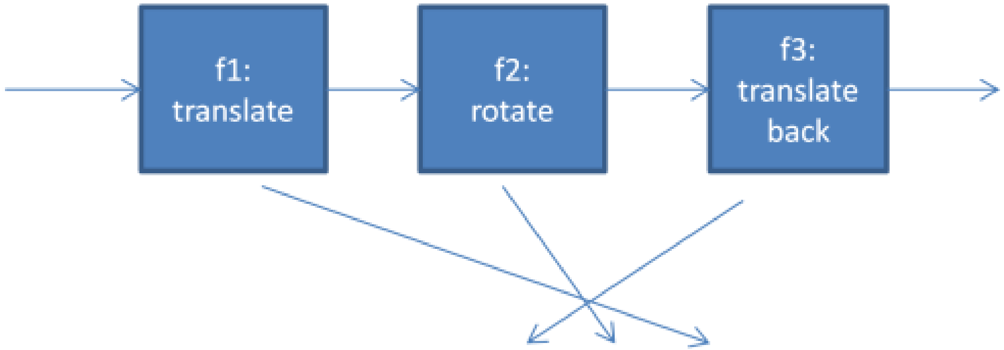


Example Composite Transformation (4)

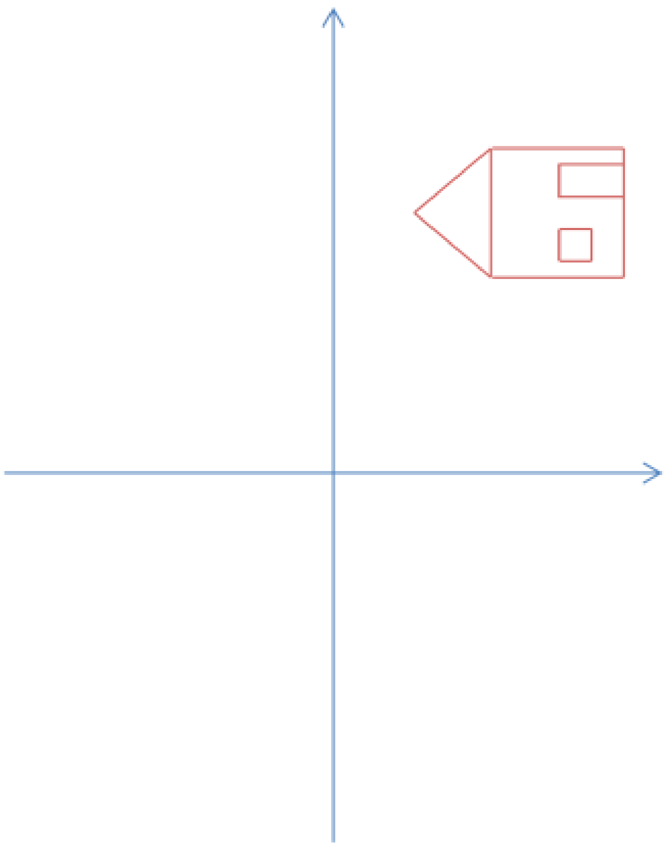




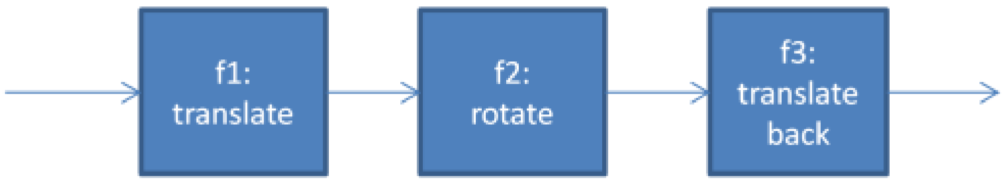
Example Composite Transformation (5)

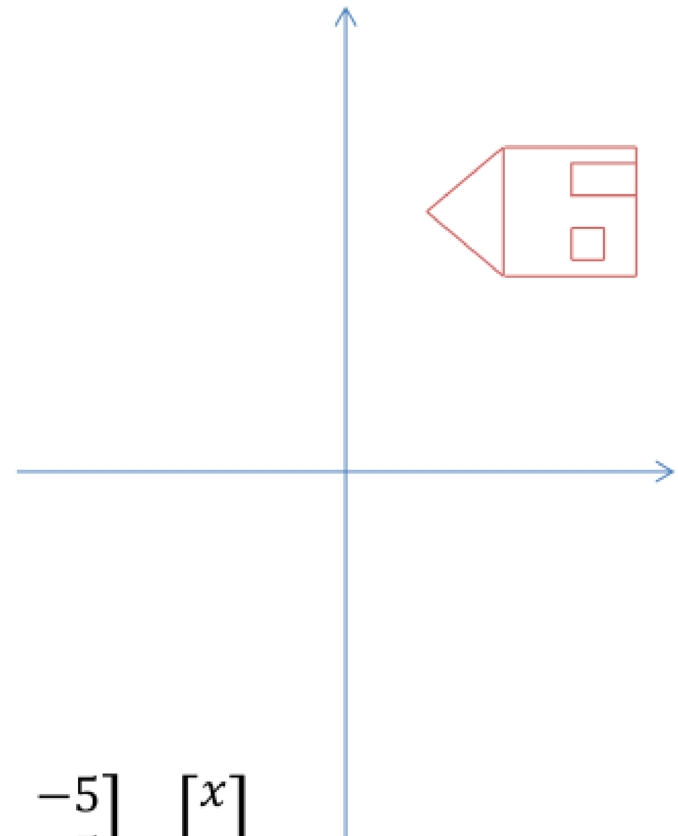


$$\vec{x}' = M_3 M_2 M_1 \vec{x}$$



Example Composite Transformation (6)

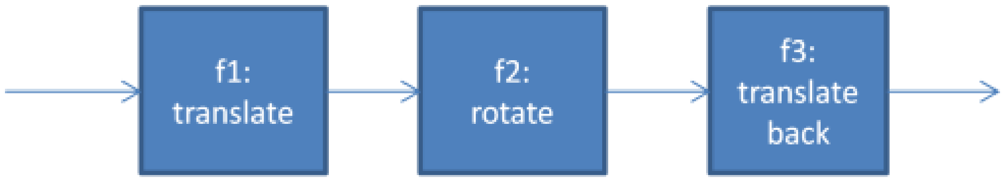


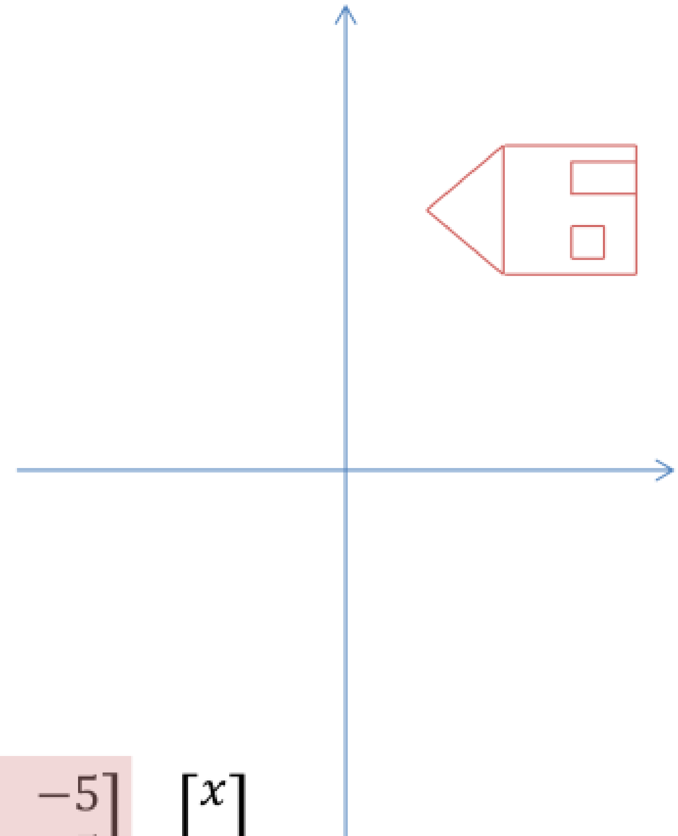


$$\vec{x}' = M_3 M_2 M_1 \vec{x}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & -5 \\ 0 & 1 & 0 & -5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Example Composite Transformation (7)

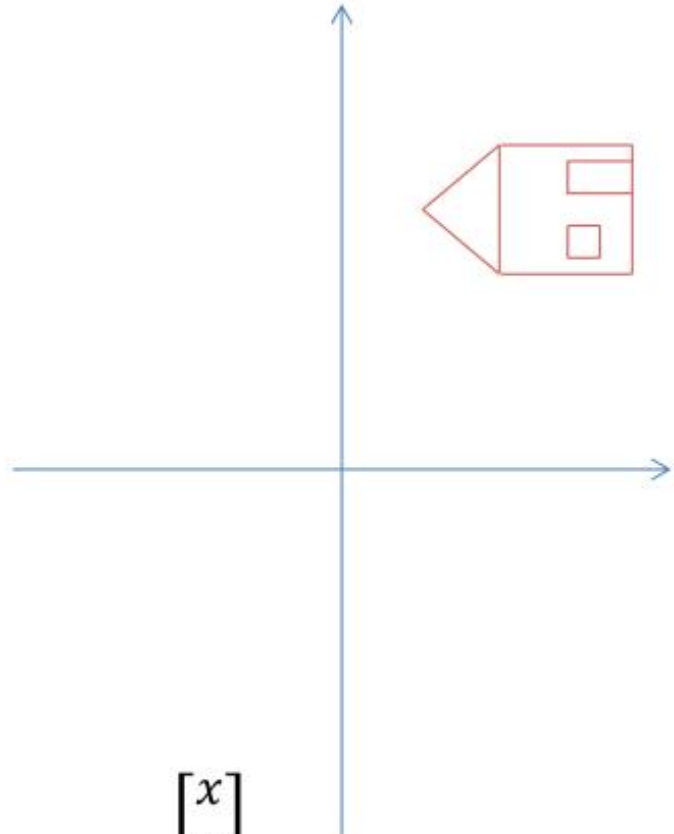




$$\vec{x}' = M_3 M_2 M_1 \vec{x}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & -5 \\ 0 & 1 & 0 & -5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

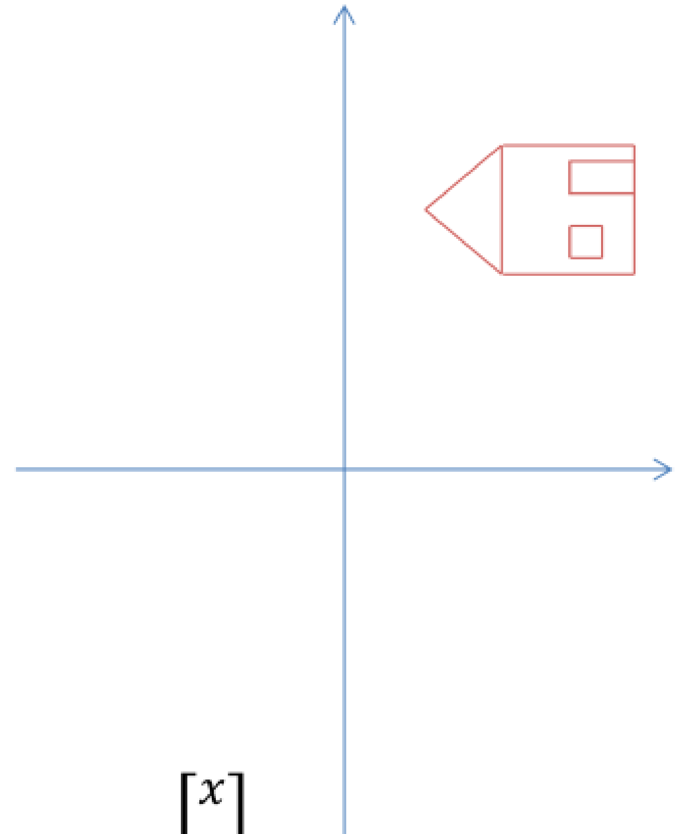
Example Composite Transformation (8)



$$\vec{x}' = M_3 M_{21} \vec{x}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & -5 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

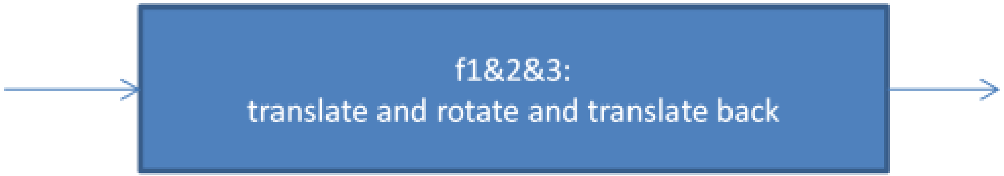
Example Composite Transformation (9)

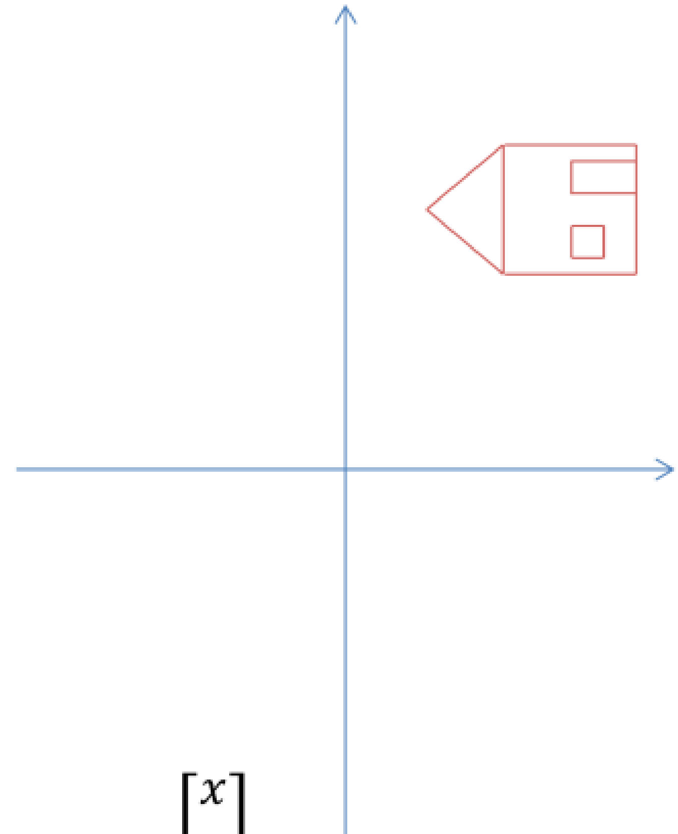


$$\vec{x}' = \boxed{M_3 M_{21}}\vec{x}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 & -5 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Example Composite Transformation (10)





$$\vec{x}' = M_{321}\vec{x}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 10 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D Transformation Example 11

EXAMPLE 11: Composite Rotation - Bending

Compute the bending matrix:

rotation about the x-axis by θ_x → rotation about the y-axis by θ_y

Does the order of the rotations matter?

SOLUTION

1. $$\mathbf{M}_{\mathbf{BEND}} = \mathbf{R}_{\mathbf{y}}(\theta_y)\cdot\mathbf{R}_{\mathbf{x}}(\theta_x) = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ 0 & \sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_y & \sin\theta_x\sin\theta_y & \cos\theta_x\sin\theta_y & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ -\sin\theta_y & \sin\theta_x\cos\theta_y & \cos\theta_x\cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. Compute the reverse order

$$\mathbf{M}'_{\mathbf{BEND}} = \mathbf{R}_{\mathbf{x}}(\theta_x)\cdot\mathbf{R}_{\mathbf{y}}(\theta_y) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ 0 & \sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ \sin\theta_x\sin\theta_y & \cos\theta_x & -\sin\theta_x\cos\theta_y & 0 \\ -\cos\theta_x\sin\theta_y & \sin\theta_x & \cos\theta_x\cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

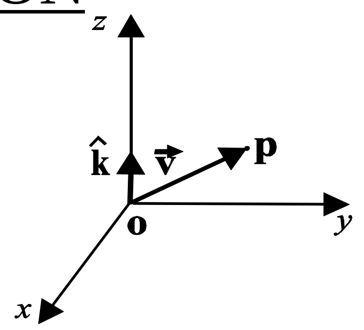
$\mathbf{M}_{BEND} \neq \mathbf{M}'_{BEND}$ so the order of the rotations matters

3D Transformation Example 12

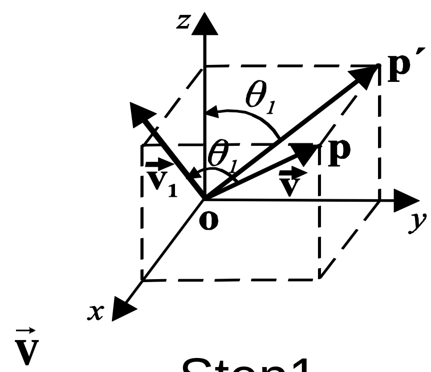
EXAMPLE 12: Alignment of Vector with Axis

Determine the transformation $\mathbf{A}(\vec{\mathbf{v}})$ that aligns a given vector $\vec{\mathbf{v}} = [a, b, c]^T$ with the unit vector $\hat{\mathbf{k}}$ along the positive z-axis.

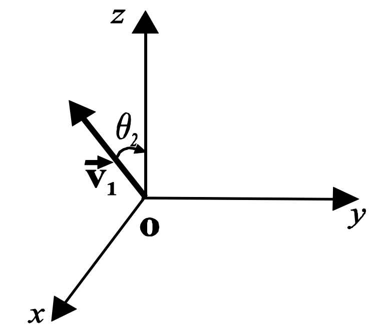
SOLUTION



Initial



Step1



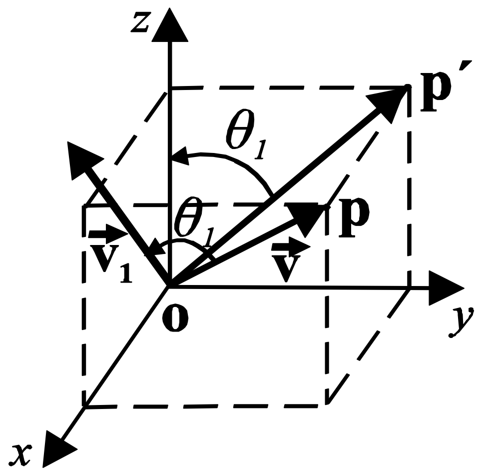
Step 2

Using two rotations:

- Step 1: Rotate about x by θ_1 so that $\vec{\mathbf{v}}$ is mapped onto $\vec{\mathbf{v}}_1$ which lies on the xz-plane, $\mathbf{R_x}(\theta_1)$
- Step 2: Rotate $\vec{\mathbf{v}}_1$ about y by θ_2 so that it coincides with $\hat{\mathbf{k}}$, $\mathbf{R_y}(\theta_2)$

3D Transformation Example 12 (2)

- Alignment matrix $\mathbf{A}(\vec{\mathbf{v}})$: $\mathbf{A}(\vec{\mathbf{v}}) = \mathbf{R_y}(\theta_2) \cdot \mathbf{R_x}(\theta_1)$
- Compute angle θ_1:
  - θ_1 is equal to the angle formed between the projection of $\vec{\mathbf{v}}$ onto the yz-plane and the z-axis
  - For the tip **p** of $\vec{\mathbf{v}}$ we have **p**=[a, b, c]^T
  - The tip of its projection on yz is **p**′=[0, b, c]^T
  - Assuming that b, c are not both 0 we get:



$$\sin\theta_1 = \frac{b}{\sqrt{b^2+c^2}} \quad , \quad \cos\theta_1 = \frac{c}{\sqrt{b^2+c^2}}$$

Thus,

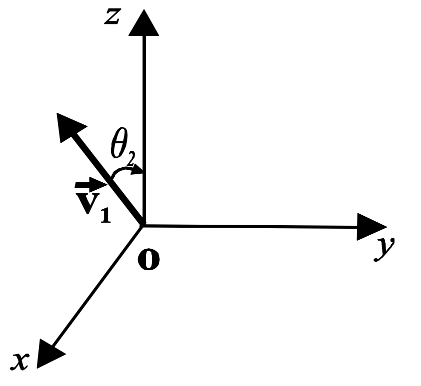
$$\mathbf{R_x}(\theta_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_1 & -\sin\theta_1 & 0 \\ 0 & \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{c}{\sqrt{b^2+c^2}} & -\frac{b}{\sqrt{b^2+c^2}} & 0 \\ 0 & \frac{b}{\sqrt{b^2+c^2}} & \frac{c}{\sqrt{b^2+c^2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 12 (3)

- Apply $R_x(\theta_1)$ to $\vec{v}$, to get its xz projection $\vec{v}_1$:

$$\overrightarrow{\mathbf{v}_1} = \mathbf{R_x}(\theta_1)\cdot\vec{\mathbf{v}} = \mathbf{R_x}(\theta_1)\cdot\begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ \sqrt{b^2+c^2} \\ 1 \end{bmatrix}$$

- Note that: $|\overrightarrow{\mathbf{v}_1}| = |\vec{\mathbf{v}}| = \sqrt{a^2+b^2+c^2}$
- Compute θ_2:



$$\sin\theta_2 = \frac{a}{\sqrt{a^2+b^2+c^2}} \;, \qquad \cos\theta_2 = \frac{\sqrt{b^2+c^2}}{\sqrt{a^2+b^2+c^2}}$$

- Thus

$$\mathbf{R_y}(\theta_2) = \begin{bmatrix} \cos\theta_2 & 0 & \sin\theta_2 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_2 & 0 & \cos\theta_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{b^2+c^2}}{\sqrt{a^2+b^2+c^2}} & 0 & \frac{a}{\sqrt{a^2+b^2+c^2}} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{a}{\sqrt{a^2+b^2+c^2}} & 0 & \frac{\sqrt{b^2+c^2}}{\sqrt{a^2+b^2+c^2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 12 (4)

- Compute matrix $\mathbf{A}(\vec{\mathbf{v}})$:

$$\mathbf{A}(\vec{\mathbf{v}}) = \mathbf{R}_{\mathbf{y}}(\theta_2)\cdot\mathbf{R}_{\mathbf{x}}(\theta_1) = \begin{bmatrix} \frac{\lambda}{|\vec{\mathbf{v}}|} & -\frac{ab}{\lambda|\vec{\mathbf{v}}|} & -\frac{ac}{\lambda|\vec{\mathbf{v}}|} & 0 \\ 0 & \frac{c}{\lambda} & -\frac{b}{\lambda} & 0 \\ \frac{a}{|\vec{\mathbf{v}}|} & \frac{b}{|\vec{\mathbf{v}}|} & \frac{c}{|\vec{\mathbf{v}}|} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$where \ \ |\vec{\mathbf{v}}| = \sqrt{a^2+b^2+c^2} \text{ and } \lambda = \sqrt{b^2+c^2}$$

- Compute the inverse matrix $\mathbf{A}(\vec{\mathbf{v}})^{-1}$ (useful in next example):

$$\mathbf{A}^{-1}(\vec{\mathbf{v}}) = (\mathbf{R}_{\mathbf{y}}(\theta_2)\cdot\mathbf{R}_{\mathbf{x}}(\theta_1))^{-1} = \mathbf{R}_{\mathbf{x}}(\theta_1)^{-1}\cdot\mathbf{R}_{\mathbf{y}}(\theta_2)^{-1} = \mathbf{R}_{\mathbf{x}}(-\theta_1)\cdot\mathbf{R}_{\mathbf{y}}(-\theta_2) = \begin{bmatrix} \frac{\lambda}{|\vec{\mathbf{v}}|} & 0 & \frac{a}{|\vec{\mathbf{v}}|} & 0 \\ -\frac{ab}{\lambda|\vec{\mathbf{v}}|} & \frac{c}{\lambda} & \frac{b}{|\vec{\mathbf{v}}|} & 0 \\ -\frac{ac}{\lambda|\vec{\mathbf{v}}|} & -\frac{b}{\lambda} & \frac{c}{|\vec{\mathbf{v}}|} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

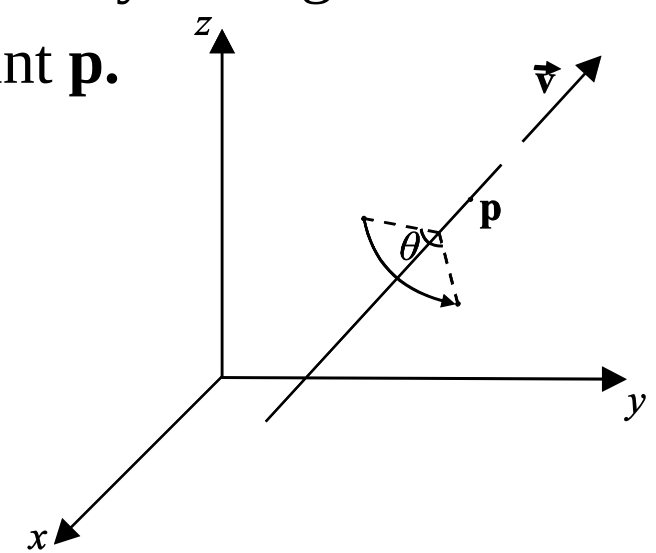
- If b=c=0 then $\vec{\mathbf{v}}$ coincides with the x-axis $\rightarrow$
  rotate about y by 90° or -90°
  depending on the sign of α

$$\mathbf{A}(\vec{\mathbf{v}}) = \mathbf{R}_{\mathbf{y}}(-\theta_2) = \begin{bmatrix} 0 & 0 & -\frac{a}{|a|} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{a}{|a|} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 13

EXAMPLE 13: Rotation about an Arbitrary Axis using 2 Translations & 5 Rotations

Find the transformation which performs a rotation by an angle θ about an arbitrary axis specified by a vector $\vec{\mathbf{v}}$ and a point **p.**



SOLUTION

$\mathbf{A}(\vec{\mathbf{v}})$ transformation [Ex. 12] :

- aligns an arbitrary vector with z-axis
- use it to reduce the problem to rotation around z

- Step 1: Translate **p** to the origin, $\mathbf{T}(-\vec{\mathbf{p}})$
- Step 2: Align $\vec{\mathbf{v}}$ with the z-axis using $\mathbf{A}(\vec{\mathbf{v}})$ matrix
- Step 3: Rotate about the z-axis by the desired angle θ, $\mathbf{R_z}(\theta)$
- Step 4: Undo the alignment, $\mathbf{A}^{-1}(\vec{\mathbf{v}})$
- Step 5: Undo the translation, $\mathbf{T}(\vec{\mathbf{p}})$

$$\mathbf{M_{ROT\text{-}AXIS}} = \mathbf{T}(\vec{\mathbf{p}}) \cdot \mathbf{A}^{-1}(\vec{\mathbf{v}}) \cdot \mathbf{R_z}(\theta) \cdot \mathbf{A}(\vec{\mathbf{v}}) \cdot \mathbf{T}(-\vec{\mathbf{p}})$$

3D Transformation Example 14

EXAMPLE 14: Coordinate System Transformation using 1 Translation&3 Rotations

Determine the transformation $\mathbf{M}_{\mathbf{ALIGN}}$ required to align a given 3D coordinate system with basis vectors $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$ with the xyz coordinate system with basis vectors $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$.

The origin of the 1st coordinate system relative to xyz is $\mathbf{O}_{\mathbf{lmn}}$.

SOLUTION

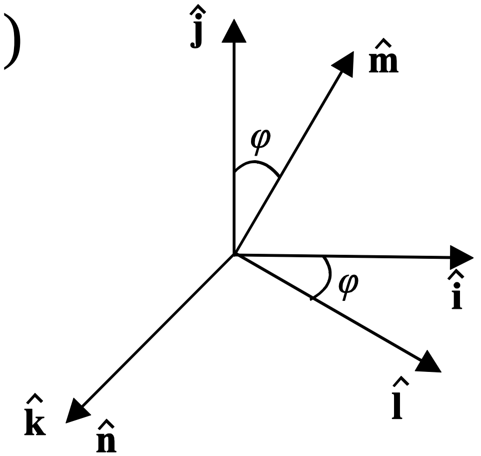
Axis transformation:

- aligning the $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$ basis to $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$ the basis $\Longrightarrow$
- changing an object's system from $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$ to $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$

The solution is an extension of $\mathbf{A}(\vec{\mathbf{v}})$ [Ex. 12]

3D Transformation Example 14 (2)

- Step 1: Translate by **-O$_{lmn}$** to make the 2 origins coincide, $\mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}})$
- Step 2: Align the $\hat{\mathbf{n}}$ basis vector with the $\hat{\mathbf{k}}$ basis vector, using $\mathbf{A}(\vec{\mathbf{v}})$ of [Ex. 12], $\mathbf{A}(\hat{\mathbf{n}})$



- Step 3: Rotate by φ around the z-axis to align the other 2 axes, $\mathbf{R}_{\mathbf{z}}(\varphi)$

$$\mathbf{M}_{\mathbf{ALIGN}} = \mathbf{R}_{\mathbf{z}}(\varphi)\cdot \mathbf{A}(\hat{\mathbf{n}})\cdot\mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}})$$

Necessary to transform the $\hat{\mathbf{l}}$ or the $\hat{\mathbf{m}}$ vector by $\mathbf{A}(\hat{\mathbf{n}})$ to estimate φ.

e.g. $\mathbf{m}' = \mathbf{A}(\hat{\mathbf{n}})\cdot\hat{\mathbf{m}}$: the *sinφ* and *cosφ* values required for the rotation are then the *x* and *y* components of $\mathbf{m}'$

3D Transformation Example 14 (3)

CONCRETE EXAMPLE :

The orthonormal basis vectors of the 2 coordinate systems are:

$$\hat{\mathbf{i}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \qquad \hat{\mathbf{j}} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \qquad \hat{\mathbf{k}} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix};$$

$$\hat{\mathbf{l}} = \begin{bmatrix} \frac{3}{\sqrt{29}} \\ \frac{4}{\sqrt{29}} \\ \frac{2}{\sqrt{29}} \end{bmatrix}, \quad \hat{\mathbf{m}} = \begin{bmatrix} -\frac{32}{\sqrt{1653}} \\ \frac{25}{\sqrt{1653}} \\ -\frac{2}{\sqrt{1653}} \end{bmatrix}, \quad \hat{\mathbf{n}} = \begin{bmatrix} -\frac{2}{\sqrt{57}} \\ -\frac{2}{\sqrt{57}} \\ \frac{7}{\sqrt{57}} \end{bmatrix}$$

and the origins of the 2 coordinate systems coincide ($\mathbf{O_{lmn}} = [0.0.0]^T$).

Find the transformation $\mathbf{M}_{\mathbf{ALIGN}}$.

- The basis vectors of the 2nd system are expressed in terms of the 1st
- From the coordinates of $\hat{\mathbf{n}}$ [Ex. 12]:

$$a = -\frac{2}{\sqrt{57}}, b = -\frac{2}{\sqrt{57}}, c = \frac{7}{\sqrt{57}} \text{ and } \lambda = \sqrt{b^2 + c^2} = \sqrt{(-\frac{2}{\sqrt{57}})^2 + (\frac{7}{\sqrt{57}})^2}$$

3D Transformation Example 14 (4)

- Compute $\mathbf{A}(\hat{\mathbf{n}})$: $\mathbf{A}(\hat{\mathbf{n}}) = \begin{bmatrix} \sqrt{\frac{53}{57}} & -\frac{4}{\sqrt{3021}} & \frac{14}{\sqrt{3021}} & 0 \\ 0 & \frac{7}{\sqrt{53}} & \frac{2}{\sqrt{53}} & 0 \\ -\frac{2}{\sqrt{57}} & -\frac{2}{\sqrt{57}} & \frac{7}{\sqrt{57}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

- Compute $\mathbf{m}'$:

$$\hat{\mathbf{m}}' = \mathbf{A}(\hat{\mathbf{n}}) \cdot \hat{\mathbf{m}} = \mathbf{A}(\hat{\mathbf{n}}) \cdot \begin{bmatrix} -\frac{32}{\sqrt{1653}} \\ \frac{25}{\sqrt{1653}} \\ -\frac{2}{\sqrt{1653}} \\ 1 \end{bmatrix} = \begin{bmatrix} -\frac{32}{\sqrt{1537}} \\ 3\sqrt{\frac{57}{1537}} \\ 0 \\ 1 \end{bmatrix}$$

- So $\sin\varphi = -\frac{32}{\sqrt{1537}}$ and $\cos\varphi = 3\sqrt{\frac{57}{1537}}$

3D Transformation Example 14 (5)

- Compute $\mathbf{R_z}(\varphi)$:

$$\mathbf{R_z}(\varphi)=\begin{bmatrix} 3\sqrt{\frac{57}{1537}} & \frac{32}{\sqrt{1537}} & 0 & 0 \\ -\frac{32}{\sqrt{1537}} & 3\sqrt{\frac{57}{1537}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Compute $\mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}})$: the origins of the 2 coordinate systems coincide so $\mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}})=\mathbf{ID}$
- So, $\mathbf{M}_{\mathbf{ALIGN}}=\mathbf{R_z}(\varphi)\cdot\mathbf{A}(\hat{\mathbf{n}})\cdot\mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}})$

$$\mathbf{M}_{\mathbf{ALIGN}}=\mathbf{R_z}(\varphi)\cdot\mathbf{A}(\hat{\mathbf{n}})\cdot\mathbf{ID}=\begin{bmatrix} 3\sqrt{\frac{57}{1537}} & \frac{32}{\sqrt{1537}} & 0 & 0 \\ -\frac{32}{\sqrt{1537}} & 3\sqrt{\frac{57}{1537}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} \sqrt{\frac{53}{57}} & -\frac{4}{\sqrt{3021}} & \frac{14}{\sqrt{3021}} & 0 \\ 0 & \frac{7}{\sqrt{53}} & \frac{2}{\sqrt{53}} & 0 \\ -\frac{2}{\sqrt{57}} & -\frac{2}{\sqrt{57}} & \frac{7}{\sqrt{57}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\% = \begin{bmatrix} \frac{3}{\sqrt{29}} & \frac{4}{\sqrt{29}} & \frac{2}{\sqrt{29}} & 0 \\ -\frac{32}{\sqrt{1653}} & \frac{25}{\sqrt{1653}} & -\frac{2}{\sqrt{1653}} & 0 \\ -\frac{2}{\sqrt{57}} & -\frac{2}{\sqrt{57}} & \frac{7}{\sqrt{57}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 15

EXAMPLE 15: Change of Basis

Determine the transformation $\mathbf{M}_{\mathbf{BASIS}}$ required to change the orthonormal basis of a coordinate system with from $B1 = (\hat{\mathbf{i}}_1, \hat{\mathbf{j}}_1, \hat{\mathbf{k}}_1)$ to $B2 = (\hat{\mathbf{i}}_2, \hat{\mathbf{j}}_2, \hat{\mathbf{k}}_2)$ and vice versa.

SOLUTION

Let the coordinates of the same vector in the 2 bases be $\vec{\mathbf{v}}_{B1}$ and $\vec{\mathbf{v}}_{B2}$.

If the coordinates of the $\hat{\mathbf{i}}_2, \hat{\mathbf{j}}_2, \hat{\mathbf{k}}_2$ basis vectors in *B1* are:

$$\hat{\mathbf{i}}_{2,B1} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \hat{\mathbf{j}}_{2,B1} = \begin{bmatrix} d \\ e \\ f \end{bmatrix} \quad \hat{\mathbf{k}}_{2,B1} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

then (from linear algebra):

$$\vec{\mathbf{v}}_{B1} = \begin{bmatrix} a & d & p \\ b & e & q \\ c & f & r \end{bmatrix} \cdot \vec{\mathbf{v}}_{B2}$$

3D Transformation Example 15 (2)

Thus ,

$$\mathbf{M}_{\mathbf{BASIS}}^{-1} = \begin{bmatrix} a & d & p \\ b & e & q \\ c & f & r \end{bmatrix}$$

B2 is an orthonormal basis $\Longrightarrow \mathbf{M}_{\mathbf{BASIS}}^{-1}$ is an orthonormal matrix $\Longrightarrow$

$$\mathbf{M}_{\mathbf{BASIS}} = (\mathbf{M}_{\mathbf{BASIS}}^{-1})^{\mathrm{T}} = \begin{bmatrix} a & b & c \\ d & e & f \\ p & q & r \end{bmatrix}$$

Homogeneous form:

$$\mathbf{M}_{\mathbf{BASIS}} = \begin{bmatrix} a & b & c & 0 \\ d & e & f & 0 \\ p & q & r & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 16

EXAMPLE 16: Coordinate System Transformation using Change of Basis

Use the change-of-basis result of Ex. 15 to align a given 3D coordinate system with basis vectors $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$ with the xyz-coordinate system with basis vectors $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$.

The origin of the 1st coordinate system relative to xyz is $\mathbf{O}_{\mathbf{lmn}}$.

SOLUTION

This is an axis transformation:

- changing an object's coordinate system from $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$ to $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$
- Change-of-basis replaces the 3 rotational transformations in Ex. 14

- Step 1: Translate by $-\mathbf{O}_{\mathbf{lmn}}$ to make the 2 origins coincide, $\mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}})$
- Step 2: Use $\mathbf{M}_{\mathbf{BASIS}}$ to change the basis from $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$ to $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$

3D Transformation Example 16 (2)

$$\mathbf{M}_{\mathbf{ALIGN2}} = \mathbf{M}_{\mathbf{BASIS}} \cdot \mathbf{T}(-\vec{\mathbf{O}}_{\mathbf{lmn}}) = \begin{bmatrix} a & b & c & -(a\,o_x + b\,o_y + c\,o_z) \\ d & e & f & -(d\,o_x + e\,o_y + f\,o_z) \\ p & q & r & -(p\,o_x + q\,o_y + r\,o_z) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where the basis vectors $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$ expressed in the basis $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$ are:

$$\hat{\mathbf{l}} = [a, b, c]^T, \hat{\mathbf{m}} = [d, e, f]^T, \hat{\mathbf{n}} = [p, q, r]^T \text{ and } \mathbf{O}_{\mathbf{lmn}} = [o_x, o_y, o_z]$$

<u>CONCRETE EXAMPLE [of Ex 14]:</u>

No translation because the 2 origins coincide .

$$\mathbf{M}_{\mathbf{BASIS}} = \begin{bmatrix} \frac{3}{\sqrt{29}} & \frac{4}{\sqrt{29}} & \frac{2}{\sqrt{29}} \\ -\frac{32}{\sqrt{1653}} & \frac{25}{\sqrt{1653}} & -\frac{2}{\sqrt{1653}} \\ -\frac{2}{\sqrt{57}} & -\frac{2}{\sqrt{57}} & \frac{7}{\sqrt{57}} \end{bmatrix}$$

with homogeneous form

$$\mathbf{M}_{\mathbf{BASIS}} = \begin{bmatrix} \frac{3}{\sqrt{29}} & \frac{4}{\sqrt{29}} & \frac{2}{\sqrt{29}} & 0 \\ -\frac{32}{\sqrt{1653}} & \frac{25}{\sqrt{1653}} & -\frac{2}{\sqrt{1653}} & 0 \\ -\frac{2}{\sqrt{57}} & -\frac{2}{\sqrt{57}} & \frac{7}{\sqrt{57}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 17

EXAMPLE 17: Rotation about an Arbitrary Axis using Change of Basis

Use the change-of-basis result of Ex. 15 to find an alternative transformation which performs a rotation by an angle θ about an arbitrary axis specified by a vector $\vec{\mathbf{v}}$ and a point **p**.

SOLUTION

- Let $$\vec{\mathbf{v}} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ and } \mathbf{p} = \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}$$
- Equation of plane perpendicular to $\vec{\mathbf{v}}$ through **p:** $\alpha(x\text{-}x_p)+b(y\text{-}y_p)+c(z\text{-}z_p)=0$
- Let: **q** a point on that plane, such that $\mathbf{q} \neq \mathbf{p}$ and $\vec{\mathbf{m}} = \mathbf{q} - \mathbf{p}$ and $\vec{\mathbf{l}} = \vec{\mathbf{m}} \times \vec{\mathbf{v}}$
- Normalize the vectors $\vec{\mathbf{l}}, \vec{\mathbf{m}}, \vec{\mathbf{v}}$ to define a coordinate system basis $(\hat{\mathbf{l}}, \hat{\mathbf{m}}, \hat{\mathbf{n}})$ with one axis being $\vec{\mathbf{v}}$ and the other 2 axes on the given plane
- Use $\mathbf{M}_{\mathbf{BASIS}}$ transformation to align it with the xyz-coordinate system
- Perform the desired rotation by θ around the z-axis

3D Transformation Example 17 (2)

- Step 1: Translate **p** to the origin, $\mathbf{T}(-\vec{\mathbf{p}})$
- Step 2: Align the $(\hat{\mathbf{l}},\hat{\mathbf{m}},\hat{\mathbf{v}})$ basis with the $(\hat{\mathbf{i}},\hat{\mathbf{j}},\hat{\mathbf{k}})$ basis, $\mathbf{M}_{\mathbf{BASIS}}$
- Step 3: Rotate about the z-axis by desired angle θ, $\mathbf{R}_{\mathbf{z}}(\theta)$
- Step 4: Undo alignment, $\mathbf{M}^{-1}_{\mathbf{BASIS}}$
- Step 5: Undo the translation, $\mathbf{T}(\vec{\mathbf{p}})$

$$\mathbf{M}_{\mathbf{ROT-AXIS2}} = \mathbf{T}(\vec{\mathbf{p}})\cdot\mathbf{M}^{-1}_{\mathbf{BASIS}}\cdot\mathbf{R}_{\mathbf{z}}(\theta)\cdot\mathbf{M}_{\mathbf{BASIS}}\cdot\mathbf{T}(-\vec{\mathbf{p}})$$

The algebraic derivation of the $\mathbf{M}_{\mathbf{ROT-AXIS2}}$ matrix is simpler than $\mathbf{M}_{\mathbf{ROT-AXIS}}$

3D Transformation Example 18

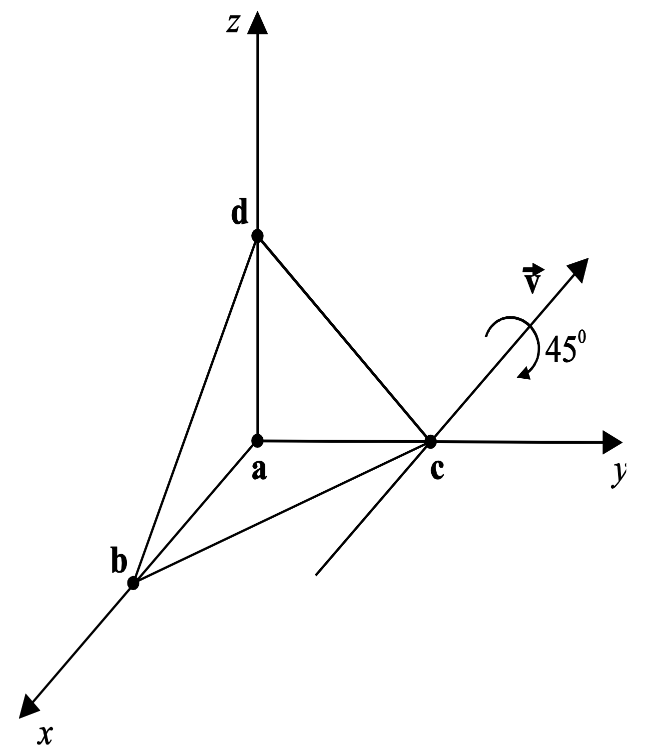
EXAMPLE 18: Rotation of a Pyramid

Rotate the pyramid defined by the vertices $\mathbf{a} = [0,0,0]^T, \mathbf{b} = [1,0,0]^T, \mathbf{c} = [0,1,0]^T$ and $\mathbf{d} = [0,0,1]^T$ by 45° about the axis defined by **c** and the vector $\vec{\mathbf{v}} = [0,1,1]^T$.

SOLUTION

The pyramid is represented by a matrix **P** :

$$\mathbf{P} = \begin{bmatrix} \mathbf{a} & \mathbf{b} & \mathbf{c} & \mathbf{d} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$



3D Transformation Example 18 (2)

- Rotate the pyramid : use the $\mathbf{M}_{\text{ROT-AXIS}}$ matrix.

$$\mathbf{M}_{\text{ROT-AXIS}} = \mathbf{T}(\vec{\mathbf{p}})\cdot \mathbf{A}^{-1}(\vec{\mathbf{v}})\cdot\mathbf{R}_{\mathbf{z}}(\theta)\cdot \mathbf{A}(\vec{\mathbf{v}})\cdot \mathbf{T}(-\vec{\mathbf{p}})$$

- The submatrices:

$$\mathbf{T}(-\vec{\mathbf{c}}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{A}(\vec{\mathbf{v}}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_{\mathbf{z}}(45^{\circ}) = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{A}^{-1}(\vec{\mathbf{v}}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}(\vec{\mathbf{c}}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Transformation Example 18 (3)

- Combine the submatrices to get:

$$\mathbf{M}_{\mathrm{ROT-AXIS}} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{2+\sqrt{2}}{4} & \frac{2-\sqrt{2}}{4} & \frac{2-\sqrt{2}}{4} \\ -\frac{1}{2} & \frac{2-\sqrt{2}}{4} & \frac{2+\sqrt{2}}{4} & \frac{\sqrt{2}-2}{4} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Compute the rotated pyramid:

$$\mathbf{P}' = \mathbf{M}_{\mathrm{ROT-AXIS}} \cdot \mathbf{P} = \begin{bmatrix} \frac{1}{2} & \frac{1+\sqrt{2}}{2} & 0 & 1 \\ \frac{2-\sqrt{2}}{4} & \frac{4-\sqrt{2}}{4} & 1 & \frac{2-\sqrt{2}}{2} \\ \frac{\sqrt{2}-2}{4} & \frac{\sqrt{2}-4}{4} & 0 & \frac{\sqrt{2}}{2} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

- The vertices of the rotated pyramid are:

$$\mathbf{a}' = [\frac{1}{2}, \frac{2-\sqrt{2}}{4}, \frac{\sqrt{2}-2}{4}]^T, \mathbf{b}' = [\frac{1+\sqrt{2}}{2}, \frac{4-\sqrt{2}}{4}, \frac{\sqrt{2}-4}{4}]^T, \mathbf{c}' = [0,1,0]^T \text{ and } \mathbf{d}' = [1, \frac{2-\sqrt{2}}{2}, \frac{\sqrt{2}}{2}]^T$$

Quaternions

- Used as an alternative way to express rotation
- Useful for animating rotations
- A quaternion consists of 4 real numbers: q = (s, x, y, z)
  - s → scalar part of quaternion q
  - $\vec{\mathbf{v}} = (x, y, z)$ → vector part of quaternion q
- Thus an alternative representation of quaternion q is: $q = (s, \vec{\mathbf{v}})$
- Can be viewed as an extension of complex numbers in 4D:
  - Using "imaginary units" i, j and k such that: $i^2=j^2=k^2=-1$ & ij=k, ji=-k and so on by cyclic permutation, quaternion q may be written as: q = s+ xi+ yj+ zk
- A real number u corresponds to the quaternion: $q = (u, \vec{\mathbf{0}})$
- An ordinary vector $\vec{\mathbf{v}}$ corresponds to the quaternion: q = $(0, \vec{\mathbf{v}})$
- A point **p** corresponds to the quaternion: q = (0, **p**)

Quaternions (2)

- Addition between quaternions:

$$q_1 + q_2 = (s_1, \vec{v}_1) + (s_2, \vec{v}_2) = (s_1 + s_2,\ \vec{v}_1 + \vec{v}_2)$$

- Multiplication between quaternions:

$$q_1 \cdot q_2 = (s_1 s_2 - \vec{\mathbf{v}}_1 \cdot \vec{\mathbf{v}}_2, s_1 \vec{\mathbf{v}}_2 + s_2 \vec{\mathbf{v}}_1 + \vec{\mathbf{v}}_1 \times \vec{\mathbf{v}}_2) =$$

$$(s_1 s_2 - x_1 x_2 - y_1 y_2 - z_1 z_2, s_1 x_2 + x_1 s_2 + y_1 z_2 - z_1 y_2,$$

$$s_1 y_2 + y_1 s_2 + z_1 x_2 - x_1 z_2, s_1 z_2 + z_1 s_2 + x_1 y_2 - y_1 x_2)$$

- Multiplication is associative
- Multiplication is **not** commutative
- The *conjugate quaternion* of q is defined as: $\overline{q} = (s, -\vec{\mathbf{v}})$
- It holds that: $\overline{q_1 \cdot q_2} = \overline{q}_2 \cdot \overline{q}_1$
- The norm of q is defined as:

$$|q|^2 = q \cdot \overline{q} = \overline{q} \cdot q = s^2 + |\vec{\mathbf{v}}|^2 = s^2 + x^2 + y^2 + z^2$$

Quaternions (3)

- It holds that: $\left|q_1 \cdot q_2\right| = \left|q_1\right| \cdot \left|q_2\right|$
- A *unit quaternion* is one whose norm: $\left|q\right| = 1$
- The *inverse quaternion* of q is defined as: $q^{-1} = \frac{1}{\left|q\right|^2}\bar{q}$
- It holds that: $q \cdot q^{-1} = q^{-1} \cdot q = 1$
- If $\left|q\right| = 1$ then $q^{-1} = \bar{q}$

Expressing rotation using quaternions

- Consider a rotation by an angle θ about an axis through the origin whose direction is specified by a unit vector $\hat{\mathbf{n}}$. The rotation can be expressed by the unit quaternion:

$$q = (\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\hat{\mathbf{n}})$$

- The above unit quaternion can be applied to a point **p** represented by the quaternion **p** = (0, **p**) as: $p' = q \cdot \mathbf{p} \cdot q^{-1} = q \cdot \mathbf{p} \cdot \bar{q}$
- Thus: $p' = \left(0, (s^2 - \vec{\mathbf{v}} \cdot \vec{\mathbf{v}})\mathbf{p} + 2\vec{\mathbf{v}}(\vec{\mathbf{v}} \cdot \mathbf{p}) + 2s(\vec{\mathbf{v}} \times \mathbf{p})\right)$.

  where

$$s = \cos\frac{\theta}{2} \quad \text{and} \quad \vec{\mathbf{v}} = \sin\frac{\theta}{2}\hat{\mathbf{n}}$$

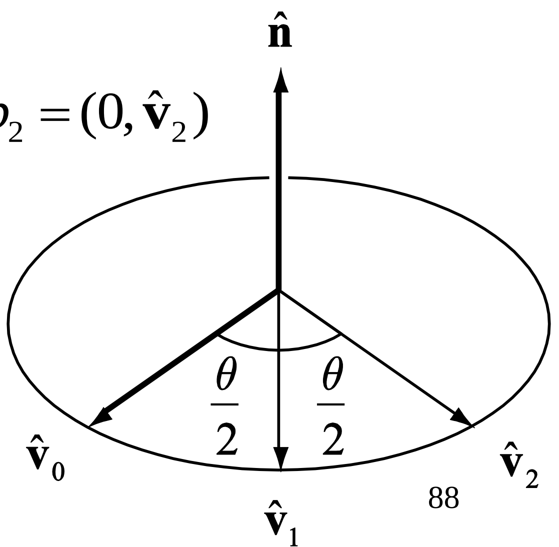
- Notice that quaternion p' represents a point **p'** since its scalar part is 0
- Point **p'** is exactly the image of the original point **p** after rotation by angle θ about the given axis

Expressing rotation using quaternions (2)

- Expressing 2 consecutive rotations:

$$q_2 \cdot (q_1 \cdot \mathbf{p} \cdot \overline{q_1}) \cdot \overline{q_2} = (q_2 \cdot q_1) \cdot \mathbf{p} \cdot (\overline{q_1} \cdot \overline{q_2}) = (q_2 \cdot q_1) \cdot \mathbf{p} \cdot \overline{(q_2 \cdot q_1)}$$

- The composite rotation is represented by the unit quaternion: $q = q_2 q_1$
- Quaternion multiplication is simpler, requires fewer operations and is numerically more stable than rotation matrix multiplication
- Proof of quaternion rotation:
  - Consider a unit vector $\hat{\mathbf{v}}_0$, a rotation axis $\hat{\mathbf{n}}$ and the images $\hat{\mathbf{v}}_1$, $\hat{\mathbf{v}}_2$ of $\hat{\mathbf{v}}_0$ after 2 consecutive rotations by θ/2 around $\hat{\mathbf{n}}$
  - The respective quaternions are: $p_0 = (0, \hat{\mathbf{v}}_0),\ p_1 = (0, \hat{\mathbf{v}}_1),\ p_2 = (0, \hat{\mathbf{v}}_2)$
  - We observe : $\cos\frac{\theta}{2} = \hat{\mathbf{v}}_0 \cdot \hat{\mathbf{v}}_1$ and $\sin\frac{\theta}{2}\ \hat{\mathbf{n}} = \hat{\mathbf{v}}_0 \times \hat{\mathbf{v}}_1$
  - Thus we write: $q = (\hat{\mathbf{v}}_0 \cdot \hat{\mathbf{v}}_1,\ \hat{\mathbf{v}}_0 \times \hat{\mathbf{v}}_1) = p_1 \cdot \overline{p_0}$
  - Similarly: $q = p_2 \cdot \overline{p_1}$



Expressing rotation using quaternions (3)

- Then: $q \cdot p_0 \cdot \overline{q} = (p_1 \cdot \overline{p_0}) \cdot p_0 \cdot \overline{(p_2 \cdot \overline{p_1})} = (p_1 \cdot \overline{p_0}) \cdot p_0 \cdot p_1 \cdot \overline{p_2} = p_1 \cdot p_1 \cdot \overline{p_2} = p_2$
  since $p_1 \cdot p_1 = (-1, \vec{0}) = -1$ because $|\mathbf{v}_1| = 1$ and also $(-1) \cdot \overline{p_2} = -(0, -\hat{\mathbf{v}}_2) = (0, \hat{\mathbf{v}}_2) = p_2$

- Thus $q \cdot p_0 \cdot \overline{q}$ results in the rotation of $\hat{\mathbf{v}}_0$ by angle θ about $\hat{\mathbf{n}}$

- Using similar arguments, $q \cdot p_1 \cdot \overline{q}$ results in the same rotation for $\hat{\mathbf{v}}_1$, whereas $q \cdot (0, \hat{\mathbf{n}}) \cdot \overline{q}$ yields $\hat{\mathbf{n}}$, which agrees with the fact that $\hat{\mathbf{n}}$ is the axis of rotation

Expressing rotation using quaternions (4)

- Generalizing the above for an arbitrary vector:

  $\hat{\mathbf{v}}_0, \hat{\mathbf{v}}_1, \hat{\mathbf{n}}$ are linearly independent. Therefore a vector $\vec{\mathbf{p}}$ may be written as a linear combination $\vec{p} = \lambda_0 \hat{\mathbf{v}}_0 + \lambda_1 \hat{\mathbf{v}}_1 + \lambda \hat{\mathbf{n}}$
- Then:

$$
\begin{aligned}
q \cdot (0, \vec{\mathbf{p}}) \cdot \overline{q} &= q \cdot (0, \lambda_0 \hat{\mathbf{v}}_0 + \lambda_1 \hat{\mathbf{v}}_1 + \lambda \hat{\mathbf{n}}) \cdot \overline{q} \\
&= q \cdot (0, \lambda_0 \hat{\mathbf{v}}_0) \cdot \overline{q} + q \cdot (0, \lambda_1 \hat{\mathbf{v}}_1) \cdot \overline{q} + q \cdot (0, \lambda \hat{\mathbf{n}}) \cdot \overline{q} \\
&= \lambda_0 (q \cdot (0, \hat{\mathbf{v}}_0) \cdot \overline{q}) + \lambda_1 (q \cdot (0, \hat{\mathbf{v}}_1) \cdot \overline{q}) + \lambda (q \cdot (0, \hat{\mathbf{n}}) \cdot \overline{q})
\end{aligned}
$$

which is exactly a quaternion with 0 scalar part and a vector part made up of the rotated components of $\vec{\mathbf{p}}$

Conversion between Quaternions and Rotation Matrices

- The rotation matrix corresponding to a rotation represented by a unit quaternion q = (s, x, y, z) is :

$$\mathbf{R}_q = \begin{bmatrix} 1-2y^2-2z^2 & 2xy-2sz & 2xz+2sy & 0 \\ 2xy+2sz & 1-2x^2-2z^2 & 2yz-2sx & 0 \\ 2xz-2sy & 2yz+2sx & 1-2x^2-2y^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- If the following matrix

$$\mathbf{R} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & 0 \\ m_{10} & m_{11} & m_{12} & 0 \\ m_{20} & m_{21} & m_{22} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

represents a rotation then the corresponding quaternion q = (s, x, y, z) may be computed as follows

Conversion between Quaternions and Rotation Matrices (2)

- Step 1:

$$s = \frac{1}{2}\sqrt{m_{00} + m_{11} + m_{22} + 1}$$

- Step 2:

$$x = \frac{m_{21} - m_{12}}{4s}, \quad y = \frac{m_{02} - m_{20}}{4s}, \quad z = \frac{m_{10} - m_{01}}{4s}$$

- If s = 0 (or is a number near 0), use a different set of relations:

then

$$x = \frac{1}{2}\sqrt{m_{00} - m_{11} - m_{22} + 1}, \quad y = \frac{m_{01} + m_{10}}{4x},$$

$$z = \frac{m_{02} + m_{20}}{4x}, \quad s = \frac{m_{21} - m_{12}}{4x}$$

An Example

<u>EXAMPLE 19: Rotation of a pyramid</u>

Re- work example 18, using quaternions.

<u>SOLUTION</u>

- Step 1: Translation by $-\vec{\mathbf{c}}, \mathbf{T}(-\vec{\mathbf{c}})$ so that the axis passes through the origin
- Step 2: perform the rotation using $\mathbf{R}_q$. The quaternion that expresses the rotation by 45^o about an axis with direction $\hat{\mathbf{v}}$ is:

$$q = \left(\cos\frac{45^\circ}{2}, \sin\frac{45^\circ}{2}\,\hat{\mathbf{v}}\right) = (\cos 22.5^\circ, 0, \frac{\sin 22.5^\circ}{\sqrt{2}}, \frac{\sin 22.5^\circ}{\sqrt{2}})$$

where

$$\cos^2 22.5^\circ = \frac{1+\cos 45^\circ}{2} = \frac{2+\sqrt{2}}{4}, \sin^2 22.5^\circ = \frac{1-\cos 45^\circ}{2} = \frac{2-\sqrt{2}}{4}$$

An Example (2)

Therefore:

$$\mathbf{R}_q = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{2+\sqrt{2}}{4} & \frac{2-\sqrt{2}}{4} & 0 \\ -\frac{1}{2} & \frac{2-\sqrt{2}}{4} & \frac{2+\sqrt{2}}{4} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Step 3: Translate by $\vec{\mathbf{c}}, \mathbf{T}(\vec{\mathbf{c}})$

The final transformation is:

$$\mathbf{M}_{ROT-AXIS3} = \mathbf{T}(\vec{\mathbf{c}}) \cdot \mathbf{R}_q \cdot \mathbf{T}(-\vec{\mathbf{c}}) = \mathbf{M}_{ROT-AXIS} \quad \text{[Ex. 18]}$$

Geometric Properties

- Affine transformations preserve important geometric features of objects. That's why they are used in Computer Graphics and Visualization
- For example let Φ be an affine transformation and **p**, **q** points, then:

$$\Phi(\lambda\mathbf{p}+(1-\lambda)\mathbf{q})=\lambda\Phi(\mathbf{p})+(1-\lambda)\Phi(\mathbf{q}),\ \lambda\in[0,1]$$

  states that the affine transformation of a line segment under Φ is another line segment
- Ratios of distances on the line segment (λ / (1-λ)) are preserved
- Affine transformation subclasses: *linear*, *similitudes*, *rigid*

Property preserved	Affine	Linear	Similitude	Rigid
Angles	No	No	Yes	Yes
Distances	No	No	No	Yes
Ratios of distances	Yes	Yes	Yes	Yes
Parallel lines	Yes	Yes	Yes	Yes
Affine combinations	Yes	Yes	Yes	Yes
Straight lines	Yes	Yes	Yes	Yes
Cross ratios	Yes	Yes	Yes	Yes

Geometric Properties (2)

- Linear transformation can be presented by a matrix **A**
  - Linear: all homogeneous affine transformations
- Similitudes preserve the similarity of objects. The result is identical to the initial object, except for its size
  - Similitudes: rotation, translation, isotropic scaling
- Rigid transformations preserve all the geometric features of objects
  - Rigid: rotations and translations