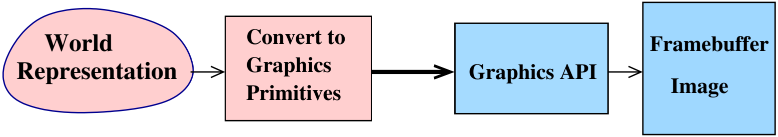
CSE251
Basics of Computer Graphics
Module: Geometry

Avinash Sharma

Spring 2019

Overview

Graphics Process



- **Model** the desired world in your head.
- **Represent** it using natural structures in the program. Convert to standard primitives supported by the API
- **Processing** is done by the API. Converts the primitives in stages and forms an image in the framebuffer
- The image is displayed automatically on the device

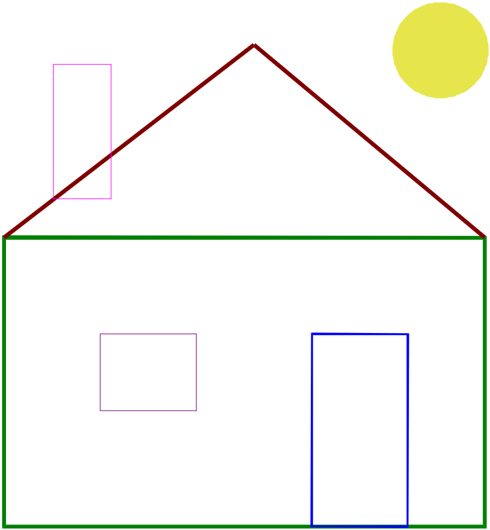
How to Draw A House?

- Compose out of basic shapes

```
drawRectangle(v1, v2, v3, v4);      // Main part
drawTriangle(v2, v3, v5);           // Roof
drawRectangle( ... );               // Door
drawRectangle( ... );               // Window
drawRectangle( ... );               // Chimney
drawCircle( ... );                  // Sun
```

- That's all, really!

Resulting House



Graphics Primitives

- Points: 2D or 3D. (x, y) or (x, y, z).
- Lines: specified using end-points
- Triangles/Polygons: specified using vertices
- Why not **circles, ellipses, hyperbolas**?

Graphics Attributes

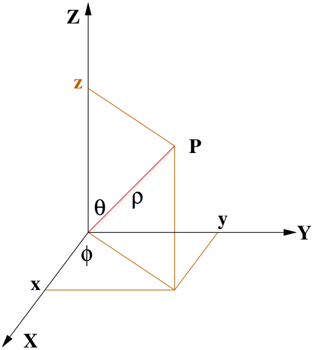
- Colour, Point width.
- Line width, Line style.
- Fill, Fill Pattern.

Point Representation

- A point is represented using 2 or 3 numbers $(x, y, [z])$ that are the projections on to the respective coordinate axes.
- Fundamental shape-defining primitive in most Graphics APIs. Everything else is built from it!
- Represented using byte, short, int, float, double, etc.
- The scale and unit are application dependent. Could be angstroms or lightyears!
- Points undergo transformations:
  Translations, Rotations, Scaling, Shearing.

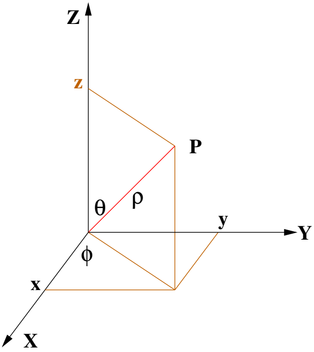
3D Coordinates

- Cartesian: (x, y, z).
- Polar: (ρ, θ, ϕ)
- $z =$
  $y =$
  $x =$
- $\rho =$
  $\phi =$
  $\theta =$
- Elevation: θ, Azimuthal: ϕ



3D Coordinates

- Cartesian: (x, y, z).
- Polar: (ρ, θ, ϕ)
- $z = \rho \cos \theta$
  $y = \rho \sin \theta \sin \phi$
  $x = \rho \sin \theta \cos \phi$
- $\rho^2 = x^2 + y^2 + z^2$
  $\phi = \tan^{-1}(y/x)$
  $\theta = \tan^{-1}(\sqrt{x^2 + y^2}/z)$
- Elevation: θ, Azimuthal: ϕ



Translation

- Translate a point $P = (x, y, [z])$ by $(a, b, [c])$.
- Points coordinates become $P' = (?, ?, ?)$.
- In vector form, $P' = ?$.

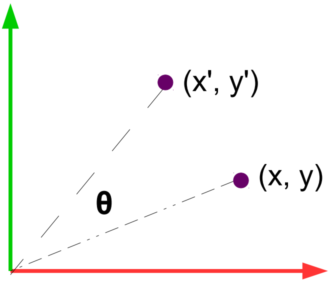
Translation

- Translate a point $P = (x, y, [z])$ by $(a, b, [c])$.
- Points coordinates become $P' = (x + a, y + b, [z + c])$.
- In vector form, $P' = P + T$, where $T = (a, b, [c])$.
- Distances, angles, parallelism are all maintained.

2D Rotation

- Rotate about origin CCW by θ.
- $x' = ?, \quad y' = ?$
- Matrix notation: $P' = R\,P$

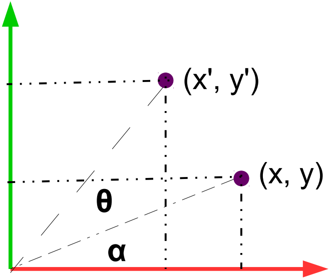
$$\begin{bmatrix} x \\ y \end{bmatrix}' = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



2D Rotation

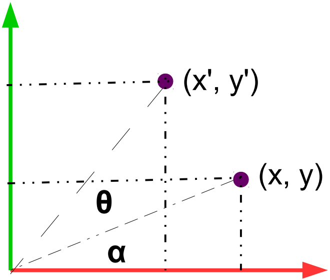
- Rotate about origin CCW by θ.
- $x' = ?, \;\; y' = ?$
- Matrix notation: $P' = R\,P$

$$\begin{bmatrix} x \\ y \end{bmatrix}' = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



2D Rotation

- Rotate about origin CCW by θ.
- $x' = x\cos\theta - y\sin\theta$,
  $y' = x\sin\theta + y\cos\theta$.



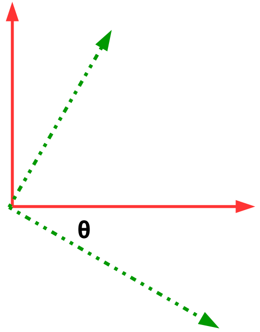
- Matrix notation: $P' = R\ P$

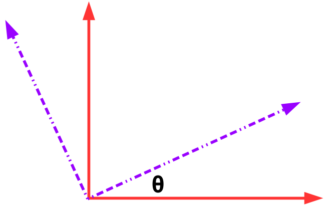
$$\begin{bmatrix} x \\ y \end{bmatrix}' = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D Rotation: Observations

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

- Orthonormal: $R^{-1} = R^T$
- Rows: vectors that **rotate to** coordinate axes
- Cols: vectors coordinate axes **rotate to**
- Invariants: distances, angles, parallelism.





3D Rotations

- Rotation could be about any axis in 3D!
- About Z-axis: Just like 2D rotation case. Z-coordinates don't change anyway.
- X-Y coordinates change exactly the same way as in 2D.
- CCW +ve, when looking into the **arrowhead**

$$R_z(\theta) = ??$$

3D Rotations

- Rotation could be about any axis in 3D!
- About Z-axis: Z-coordinates don't change anyway

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

- CCW +ve; orthonormal; length preserving
- Rows: vectors that rotate onto axes; columns: vectors that axes rotate into....

3D Rotations

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

- CCW +ve; orthonormal
- Rows: vectors that rotate onto axes; columns: vectors that axes rotate into....
- Rotation about an arbitrary axis, for example, $[1, 1, 1]^\intercal$??
  Coming soon ....

Non-uniform Scaling

- Scale along X, Y, Z directions by s, u, and t.
- $x' = s\,x, \;\; y' = u\,y, \;\; z' = t\,z$.
- We are more comfortable with $P' = S\,P$ or

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}' = \begin{bmatrix} s & 0 & 0 \\ 0 & u & 0 \\ 0 & 0 & t \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Invariants: parallelism, ratios of lengths in any direction (Angles also for uniform scaling.)

Shearing

- Add a little bit of x to y or other combinations

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}' = \begin{bmatrix} 1 & x_y & x_z \\ y_x & 1 & y_z \\ z_x & z_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- One of $x_y, x_z, y_x, y_z, z_x, z_y \neq 0$. Rectangles can become parallelograms.
- Invariants: parallelism, ratios of lengths in any direction.

Reflection

- Negative entries in a matrix indicate reflection.

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

- Reflection needn't be about a coordinate axis alone

General Transformation

- Rotation, scaling, shearing, and reflection: **Matrix-vector** product. Vectors get tranformed into other vectors
- Translation alone is a **vector-vector** addition
- Sequence of: Translation, rotation, scaling, translation and rotation: $\mathbf{P}' = \mathbf{R_2}\ [\mathbf{S}\ \mathbf{R_1}\ (\mathbf{P} + \mathbf{t_1}) + \mathbf{t_2}]$
- If translation is also a matrix-vector product, we can combine above transformations into a single matrix: $\mathbf{P}' = \mathbf{R_2}\ \mathbf{T_2}\ \mathbf{S}\ \mathbf{R_1}\ \mathbf{T_1}\ \mathbf{P} = \mathbf{M}\ \mathbf{P}$
- How?

General Transformation

- Rotation, scaling, shearing, and reflection: **Matrix-vector** product. Vectors get tranformed into other vectors
- Translation alone is a **vector-vector** addition
- Sequence of: Translation, rotation, scaling, translation and rotation: $\mathbf{P}' = \mathbf{R_2}\ [\mathbf{S}\ \mathbf{R_1}\ (\mathbf{P} + \mathbf{t_1}) + \mathbf{t_2}]$
- If translation is also a matrix-vector product, we can combine above transformations into a single matrix: $\mathbf{P}' = \mathbf{R_2}\ \mathbf{T_2}\ \mathbf{S}\ \mathbf{R_1}\ \mathbf{T_1}\ \mathbf{P} = \mathbf{M}\ \mathbf{P}$
- How? Answer: **homogeneous coordinates**!

Homogeneous Coordinates

- Add a *non-zero scale factor* w to each coordinate.
  A 2D point is represented by a vector $[x \;\; y \;\; w]^\mathsf{T}$
- $[x \;\; y \;\; w]^\mathsf{T} \equiv (x/w, \; y/w)$.
- Translate $[x \;\; y]^\mathsf{T}$ by $[a \;\; b]^\mathsf{T}$ to get $[x+a \;\; y+b]^\mathsf{T}$

$$\begin{bmatrix} x+a \\ y+b \\ 1 \end{bmatrix} = \begin{bmatrix} ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogeneous Coordinates

- Add a *non-zero scale factor* w to each coordinate.
  A 2D point is represented by a vector $[x \;\; y \;\; w]^\intercal$
- $[x \;\; y \;\; w]^\intercal \equiv (x/w,\; y/w)$
- Translate $[x \;\; y]^\intercal$ by $[a \;\; b]^\intercal$ to get $[x+a \;\; y+b]^\intercal$

$$\begin{bmatrix} x+a \\ y+b \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Now, translation is also: $\mathbf{P}' = \mathbf{T}\,\mathbf{P}$, a matrix-vector product and a linear operation.

Homogeneous Coordinates

- Add a *non-zero scale factor* w to each coordinate. A 2D point is represented by a vector $[x \;\; y \;\; w]^\intercal$
- $[x \;\; y \;\; w]^\intercal \equiv (x/w, \; y/w)$.
- Now, translation is also: $\mathbf{P}' = \mathbf{T} \; \mathbf{P}$
- For a point: Rotation followed by translation followed by scaling, followed by another rotation: $\mathbf{P}' = \mathbf{R_2 S T R_1} \; \mathbf{P}$.
- Similarly for 3D. Points represented by: $[x \; y \; z \; w]^T$.
- All matrices are 3×3 in 2D. Last row is $[0 \;\; 0 \;\; 1]$.
- All matrices are 4×4 in 3D. Last row is $[0 \;\; 0 \;\; 0 \;\; 1]$.

Homogeneous Representation

- Convert to a 4-vector with a scale factor as fourth.
  $(x, y, z) \equiv [kx \;\; ky \;\; kz \;\; k]^\mathsf{T}$ for any $k \neq 0$.
- Translation, rotation, scaling, shearing, etc. become linear operations represented by 4×4 matrices.
- What does $[x \; y \; z \; 0]^\mathsf{T}$ mean?
- $[a \; b \; c \; d]^\mathsf{T}$ could be a point or a plane. Lines are specified using two such vectors, either as join of two points or intersection of two planes!

Transformation Matrices: Rotations

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

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

- CCW +ve; orthonormal; length preserving; rows give direction vectors that rotate onto each axis; columns ....

Translation, Scaling, Composite

$$T(a,b,c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad S(a,b,c) = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- A sequence of transforms can be represented using a composite matrix: $\mathbf{M} = \mathbf{R_x T R_y S T} \cdots$
- Operations are not commutative, but are associative.
- $\mathbf{RT}$ and $\mathbf{TR}$??

Rotation and Translation

- $T_{4\times4} = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$ and $R_{4\times4} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$
- $T\ R = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \ ?$
- $R\ T = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \ ?$

Rotation and Translation

- $T_{4\times 4} = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$ and $R_{4\times 4} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$
- $T\ R = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$
- $R\ T = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{Rt} \\ \mathbf{0} & 1 \end{bmatrix}$
- $TR = R\ T$ if: (a) $\mathbf{R} = \mathbf{I}$ or (b) $\mathbf{t} = \mathbf{0}$ or (c) $\mathbf{Rt} = ?$

Rotation and Translation

- $T_{4\times4} = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$ and $R_{4\times4} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$
- $T\ R = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$
- $R\ T = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{Rt} \\ \mathbf{0} & 1 \end{bmatrix}$
- $TR = R\ T$ if: (a) $\mathbf{R} = \mathbf{I}$ or (b) $\mathbf{t} = \mathbf{0}$ or (c) $\mathbf{Rt} = \mathbf{t}$
- When is $\mathbf{Rt} = \mathbf{t}$? eigenvector of $\mathbf{R}$

Rotation and Translation

- $T_{4\times 4} = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$ and $R_{4\times 4} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$
- $T\ R = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$
- $R\ T = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{Rt} \\ \mathbf{0} & 1 \end{bmatrix}$
- $TR = R\ T$ if: (a) $\mathbf{R} = \mathbf{I}$ or (b) $\mathbf{t} = \mathbf{0}$ or (c) $\mathbf{Rt} = \mathbf{t}$
- When is $\mathbf{Rt} = \mathbf{t}$? when $\mathbf{t}$ is an eigenvector of $\mathbf{R}$

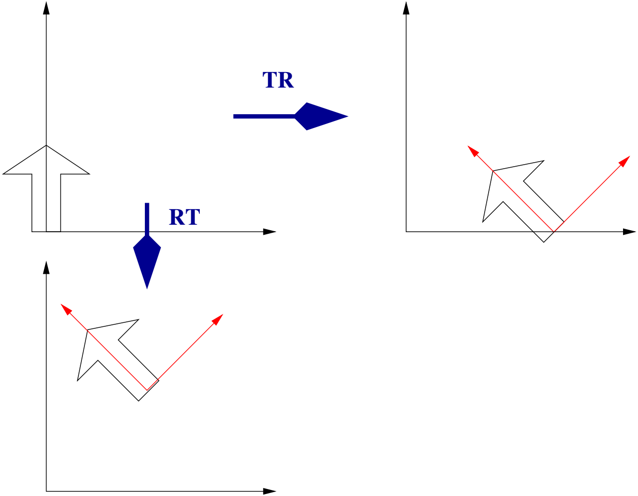
Commutativity

- Translations are commutative: $\mathbf{T_1 T_2 = T_2 T_1}$
- Scaling is commutative: $\mathbf{S_1 S_2 = S_2 S_1}$
- Are rotations commutative? $\mathbf{R_1 R_2 \stackrel{?}{=} R_2 R_1}$
- What would be an example?
  Consider the effect on Z-axis of:

Commutativity

- Translations are commutative: $\mathbf{T_1 T_2 = T_2 T_1}$
- Scaling is commutative: $\mathbf{S_1 S_2 = S_2 S_1}$
- Are rotations commutative? $\mathbf{R_1 R_2 \neq R_2 R_1}$
- Consider the effect on Z-axis of
  $\mathbf{R_x(90) R_y(90)}$ and $\mathbf{R_y(90) R_x(90)}$
- $\mathbf{RT \neq TR}$. (If translation is not parallel to rotation axis)
- Consider: $\mathbf{R}(\pi/\mathbf{4})$ and $T(5,0)$.
  Where does the origin $(0,0)$ go in $\mathbf{TR}$ and $\mathbf{RT}$?

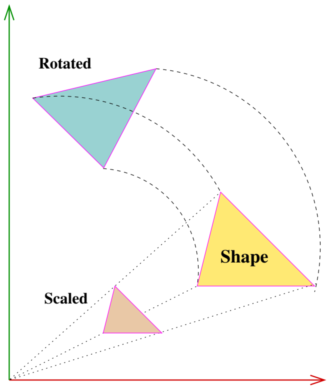
TR and RT



TR keeps it on X axis to $(5, 0)$. **RT** takes it to $(\frac{5}{\sqrt{2}}, \frac{5}{\sqrt{2}})$.

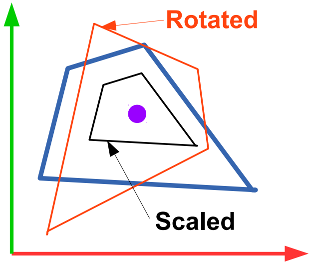
Objects Away from Origin

- Object "translates" when rotated or scaled!!
- Default: Perform these **about the origin**
- How do we transform points "in place"?
- Rotate or scale about the centroid of the object. Or about an arbitrary point
- How?



Transformations About A Point

- Rotating about point P
  – Align P with origin
  – Rotate/scale about origin
  – Translate back
- Rotation:
  $\mathbf{R_C}(\theta) = \mathbf{T(C)\ R\ T(-C)}$
- Scaling:
  $\mathbf{S_C}() = \mathbf{T(C)\ S()\ T(-C)}$
- A transformation $\mathbf{M}$:
  $\mathbf{M_C} = \mathbf{T(C)\ M\ T(-C)}$



R, T Operations on Points

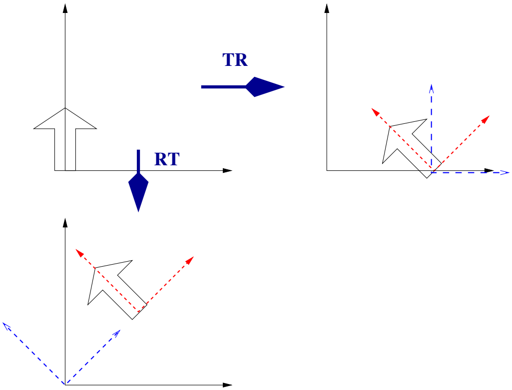
- **T(5,0) R($\pi/4$)**: Impact on a point:
  - R($\pi/4$): **(Point stays at $(\mathbf{0}, \mathbf{0})$)**
  - T$(5, 0)$: **(Point goes to $(\mathbf{5}, \mathbf{0})$)**
- **R($\pi/4$) T(5,0)**: Impact on the point:
  - T$(5, 0)$: **(Point moves to $(\mathbf{5}, \mathbf{0})$)**
  - R($\pi/4$). **(Point rotates about origin)**
- All points on the shape undergo the same motions and get new coordinates

R, T Operaions on Frames

- **T(5,0) R($\pi/4$)**: Impact on coordinate frame:
  - T$(5, 0)$: **(Origin shifted to $(\mathbf{5}, \mathbf{0})$)**
  - R$(\pi/4)$. **(Axes rotated at new origin)**
- **R($\pi/4$) T(5,0)**: Impact on coordinate frame:
  - R$(\pi/4)$: **(Axes rotate by 45 degrees))**
  - T$(5, 0)$. **(Point moves to $(\mathbf{5}, \mathbf{0})$ in new axes)**
- Frames move around, giving new coordinates to points on objects!!

Relating Coordinate Frames

- $T(5,0)$ and $R(\pi/4)$
- Start: Black axes
  Next: Blue axes
  Last: Red axes



Points and Frames in General

- Points go through changes in a common coordinate frame when a sequence of transformations is **viewed from right to left**
- Coordinate system goes through the same transformations when the sequence is **viewed from left to right**
- Composite transformations $P' = \mathbf{M_1 M_2 M_3}\ P$ relates the coordinates in successive coordinate frames as we go from left to right, starting with $\mathbf{X'Y'}$ coordinate frame to finally the $\mathbf{XY}$ frame.

Transforming the World Reference

- Consider $P_4 \;= \mathbf{M_4 M_3 M_2 M_1} \;\; P_0$
- Point P_0 undergoes 4 operations and get coordinates P_4
- Imagine the point having coordinates P_1, P_2, P_3 after operations $\mathbf{M_1}, \mathbf{M_2}, \mathbf{M_3}$
- We can also visualize coordinate frames $\mathbf{\Pi}_4, \mathbf{\Pi}_3, \mathbf{\Pi}_2, \mathbf{\Pi}_1, \mathbf{\Pi}_0$ in which a point has coordinates P_4 to P_0 respectively
- Operation $\mathbf{M}_i$ represents a change in coordinates from $\mathbf{\Pi}_i$ to $\mathbf{\Pi}_{i-1}$, resulting in new labels for the point.

Let us look at Ourselves

- Model IIIT Campus as a whole. Campus is our "world"
- Global coordinate frame $\mathbf{\Pi}_G$ for the campus: at the Gate
- Buildings: Himalaya, Vindhya, Bakul, Parul, ..., Palash. Each has a natural coordinate frame. $\mathbf{\Pi}_H$ is Himalaya's
- Himalaya has several rooms: H105, H204, H205, H304, etc., with own coordinate frames. $\mathbf{\Pi}_C$ is of H205 (our class)
- H205 has 55 desks, with coord frames $\mathbf{\Pi}_{Di}$ for desk i
- Desks are identical in geometry; the coord frame Π_{Di} places each in its location.

Consider a Desk

- Consider a corner point P of desk 37, with coordinates $(200, 30, 100)$ in $\mathbf{\Pi}_{D37}$. That is: $P_{D37} = (200, 30, 100)$
- Since our world is the campus, we have to ultimately describe everything in the coordinate frame $\mathbf{\Pi}_G$

$$P_G = \mathbf{M_{GH}}\,\mathbf{M_{HC}}\,\mathbf{M_{CD37}}\;P_{D37}$$

- $\mathbf{M_{GH}}$ aligns $\mathbf{\Pi}_G$ to $\mathbf{\Pi}_H$. $\mathbf{M_{HC}}$ aligns $\mathbf{\Pi}_H$ to $\mathbf{\Pi}_C$. $\mathbf{M_{CD37}}$ aligns $\mathbf{\Pi}_C$ to $\mathbf{\Pi}_{D37}$
- $P_G = \mathbf{M_{GH}} \overset{P_H}{|} \mathbf{M_{HC}} \overset{P_C}{|} \mathbf{M_{CD37}} \overset{P_{D37}}{|} P$ (for any point P on Desk37)
- We can place a given desk in any **building, room, place**!

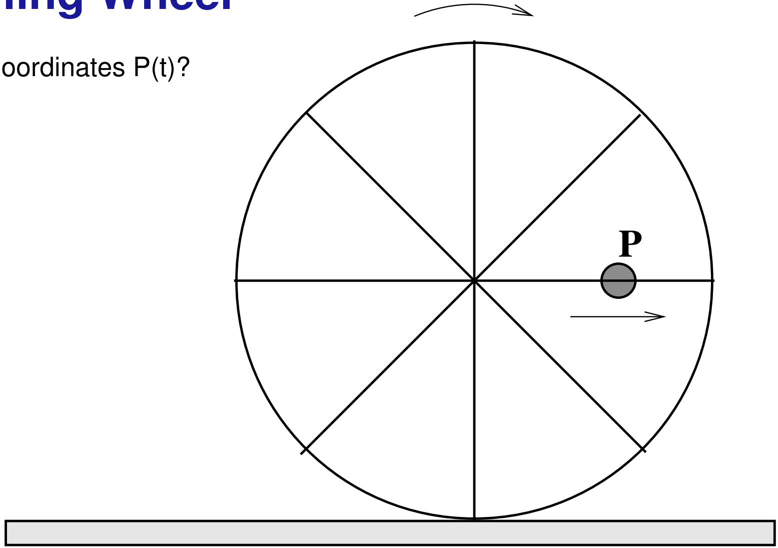
Walking on Stage

- Person walking horizontally on stage, with swinging arms
- How does the hand-tip move w.r.t each student? **How?**
- Student knows own position in room's reference frame
- Start at a student's eye. (That provides the viewpoint!)
- Align to room's reference frame using $\mathbf{M_1}$. Different matrix for each student, but everyone same now....
- Walk: pure translation. $\mathbf{M_2}$ aligns to person coord frame
- Arm swing: Simple harmonic motion with angle $\theta(t)$

Simpler viewpoints in newer coord frames.

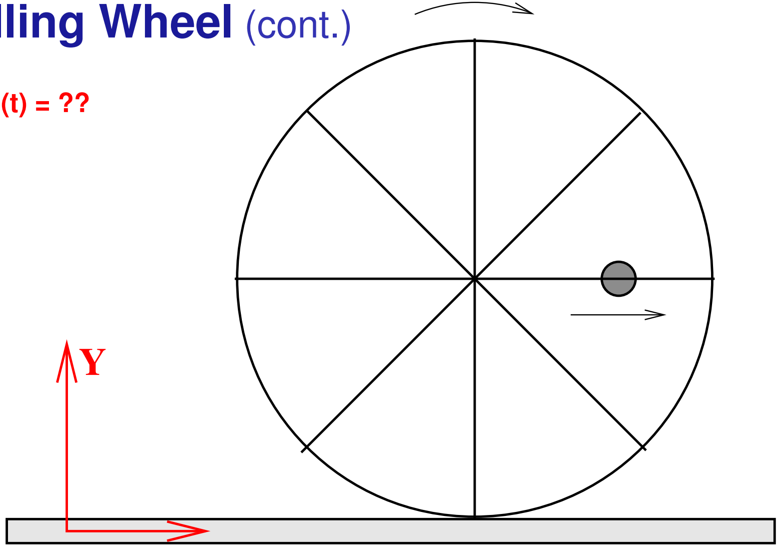
Rolling Wheel

Coordinates P(t)?



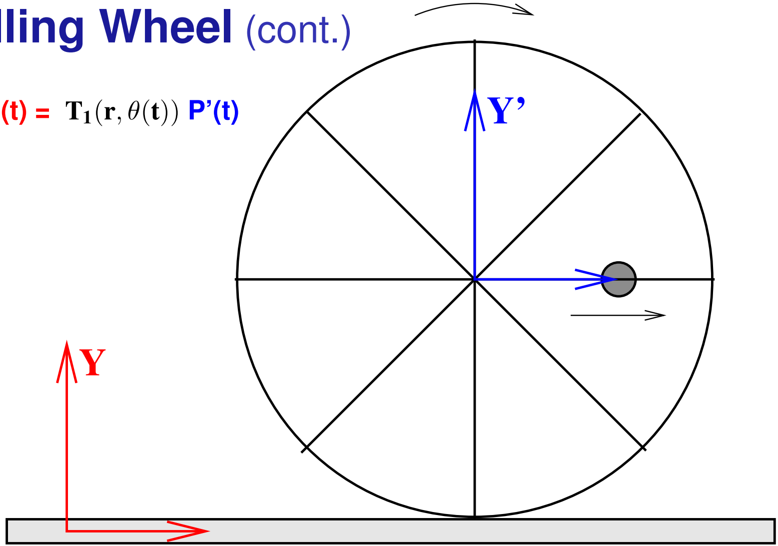
Rolling Wheel (cont.)

P(t) = ??



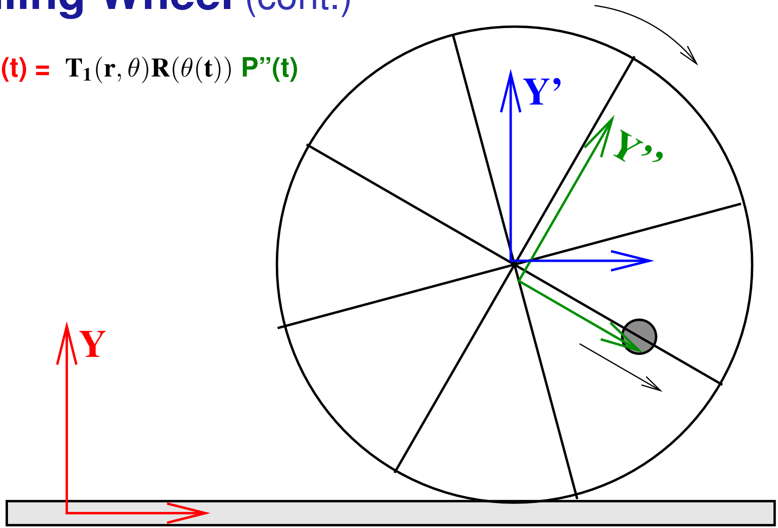
Rolling Wheel (cont.)

$\mathbf{P(t)} = \mathbf{T_1}(\mathbf{r}, \theta(\mathbf{t}))\ \mathbf{P'(t)}$

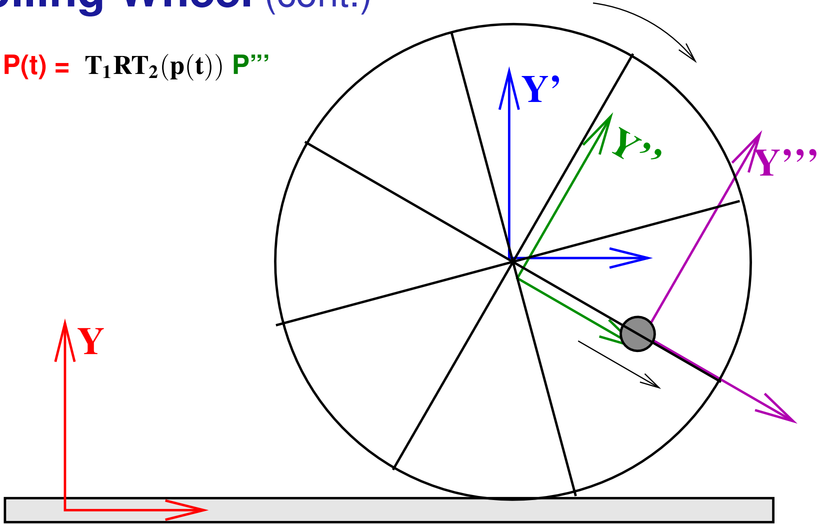


Rolling Wheel (cont.)

$\mathbf{P(t)} = \mathbf{T_1(r,\theta)R(\theta(t))}\ \mathbf{P''(t)}$



Rolling Wheel (cont.)

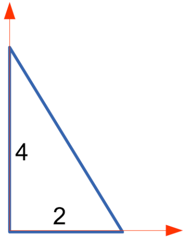


Final Transformation

- **P(t) =** $\mathbf{T_1(t)\, R(\theta(t))\, T_2(p(t))}$ **P'''**
- $\mathbf{T_1(t) = T(r\ \theta(t), r) = T(r\ \omega\ t, r)}$ (A translation matrix)
- $\mathbf{R(\theta(t)) = R_Z(\omega\ t)}$ (A normal rotation matrix)
- $\mathbf{T_2(t) = T(p(t), 0) = T(v\ t, 0)}$ (A translation matrix)
- **P'''** = $[0, 0, 1]^T$ (Origin of the bead)
- Lot simpler than thinking about it all together.
- What if we have a pendulum swinging freely on the bead?

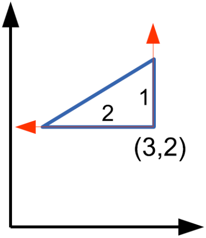
Given an object

- An object `traingleObj` is given. Can be drawn using `drawObject(triangleObj)`



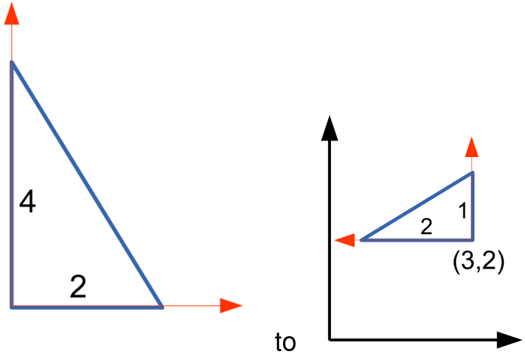
- `drawObject(triangleObj)` draws the object at (current) origin

Draw it in a different configuration



- Use `drawObject(triangleObj)`, with right transformations

Transformations

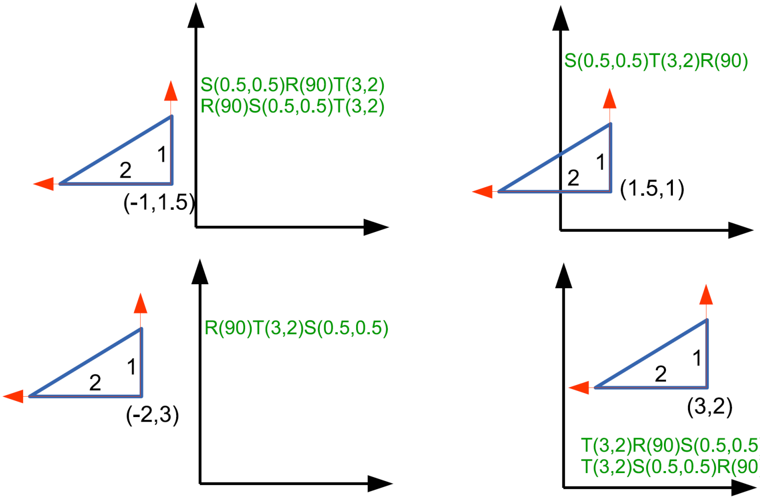


- What are the transformations? Combination of Translation, Rotation, Scaling!!
- Operations involved: $\mathbf{S(0.5, 0.5), T(3, 2), R(90)}$

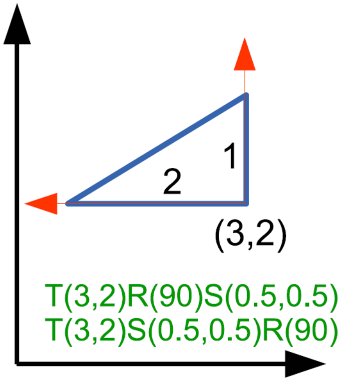
Which combination ?

1. $\mathbf{S(0.5, 0.5), R(90), T(3, 2)}$
2. $\mathbf{S(0.5, 0.5), T(3, 2), R(90)}$
3. $\mathbf{T(3, 2), R(90), S(0.5, 0.5)}$
4. $\mathbf{T(3, 2), S(0.5, 0.5), R(90)}$
5. $\mathbf{R(90), S(0.5, 0.5), T(3, 2)}$
6. $\mathbf{R(90), T(3, 2), S(0.5, 0.5)}$

Which combination ? (cont.)



Several Correct Situations

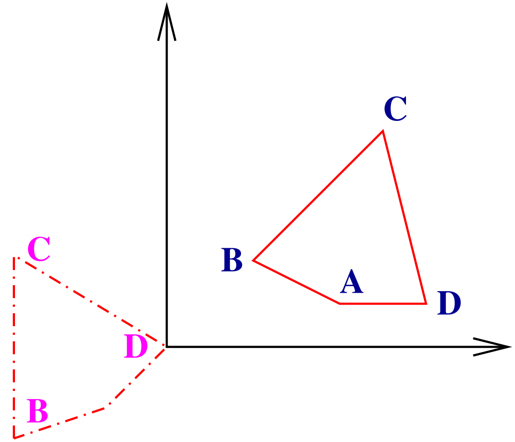


Another Similar Scenario

- A clock is hanging from a nail fixed to a flat plate. The plate is being translated with a velocity $\vec{\mathbf{v}}$ and acceleration $\vec{\mathbf{a}}$. The pendulum of the clock swings back and forth with a time period of 5 seconds and a max angle of $\pm\theta$. An ant travels from the bottom tip of the pendulum up to the centre.
- How do we compute the ant's position with respect to a fixed coordinate system coplanar with the plate?

  Please sketch the situation and work it out for yourself

A Transformation Problem

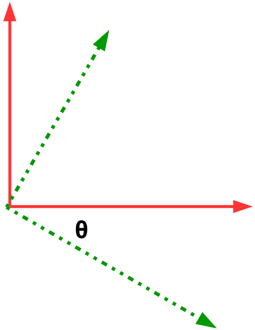


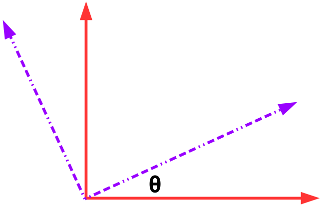
Bring **D** to origin and **BC** parallel to the Y axis as shown

2D Rotation: Observations

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

- Orthonormal: $R^{-1} = R^T$
- Rows: vectors that **rotate to** coordinate axes
- Cols: vectors coordinate axes **rotate to**
- Invariants: distances, angles, parallelism.





Transformation Computation

- Step 1: Translate by $-\mathbf{D}$. What is the orientation of BC?
- Step 2: Rotate to have unit vector $\vec{\mathbf{u}} = [u_x \;\; u_y]^\mathsf{T}$ from **B** to **C** on the Y axis. That is the second row of **R** matrix
- The matrix for the total operation: $\mathbf{M} = \mathbf{T}(-\mathbf{D})\mathbf{R}$
- Two options for first row. $[u_y \;\; -u_x]^\mathsf{T}$ and $[-u_y \; u_x]^\mathsf{T}$
- **R** matrix: (a) $\begin{bmatrix} u_y & -u_x \\ u_x & u_y \end{bmatrix}$ or (b) $\begin{bmatrix} -u_y & u_x \\ u_x & u_y \end{bmatrix}$?
- Difference? The direction aligned to the X-axis!
- Option (a) is correct. **Why?** Draw Option (b)!

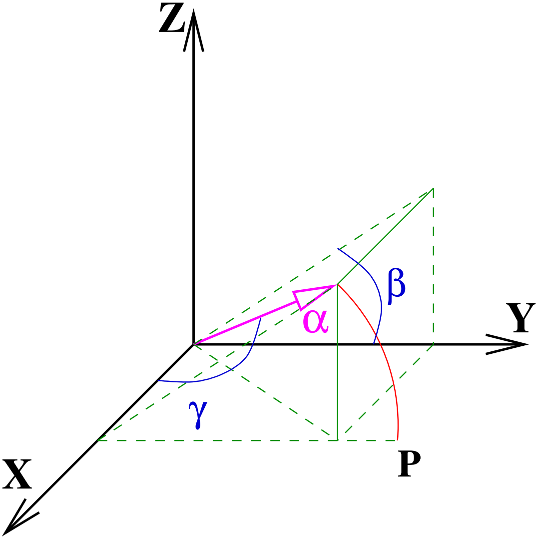
Rotation about an axis parallel to Z

- An axis parallel to Z axis, passing through point $(x, y, 0)$.
- Translate so that the axis passes through the origin: $\mathbf{T}(-x, -y, k)$ for any k!!
- Overall: $\mathbf{M} = \mathbf{T}(x, y, -k)\ \mathbf{R_Z}(\theta)\ \mathbf{T}(-x, -y, k)$
- Why shouldn't k matter? $\mathbf{R_Z}$ doesn't affect the z coordinate. So, whatever k is added first will be subtracted later

3D Rotation about an axis α

- What is $\mathbf{R}_{\alpha}(\theta)$?
- 3-step process:
  1. Apply $\mathbf{R}_{\alpha\mathbf{x}}$ to align α with the X axis.
  2. Rotate about X by angle θ.
  3. Undo the first rotation using $\mathbf{R}_{\alpha\mathbf{x}}^{-1}$
- Net result: $\mathbf{R}_{\alpha}(\theta) \quad = \quad \mathbf{R}_{\alpha\mathbf{x}}^{-1}\, \mathbf{R}_{\mathbf{x}}(\theta)\, \mathbf{R}_{\alpha\mathbf{x}}$
- Quite simple!? What is $\mathbf{R}_{\alpha\mathbf{x}}(\theta)$?
- **(We can align α with Y or Z axis also)**

3D Rotation about an axis α (cont.)



Computing $\mathbf{R}_\alpha$

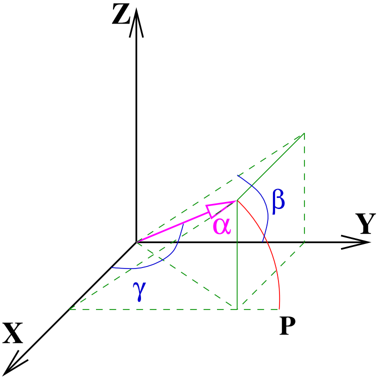
- First rotate by $-\beta$ about X axis. Vector α would lie in the XY plane, with tip at point **P**.
- $\beta = ?, \quad \tan\beta = ?$
- Next rotate by $-\gamma$ about Z axis. Vector α would coincide with the X axis.
- $\gamma = ?, \quad \tan\gamma = ?$

Computing $\mathbf{R}_\alpha$

- Rotate by $-\beta$ about X axis to bring α to XY plane
- $\tan\beta = \frac{\alpha_z}{\alpha_y}$
- Rotate by $-\gamma$ about Z axis to bring α to X axis
- $\tan\gamma = \frac{\sqrt{\alpha_y^2+\alpha_z^2}}{\alpha_x} = \frac{\sqrt{1-\alpha_x^2}}{\alpha_x}$ if $|\alpha| = 1$.
- $\mathbf{R}_{\alpha\mathbf{x}} = \mathbf{R}_\mathbf{z}(-\gamma)\mathbf{R}_\mathbf{x}(-\beta)$ and $\mathbf{R}_{\alpha\mathbf{x}}^{-1} = \mathbf{R}_\mathbf{x}(\beta)\mathbf{R}_\mathbf{z}(\gamma)$
- Alternative: Don't we know about **rotation matrices** and direction cosines that go **to/from coordinate axes**?

Final

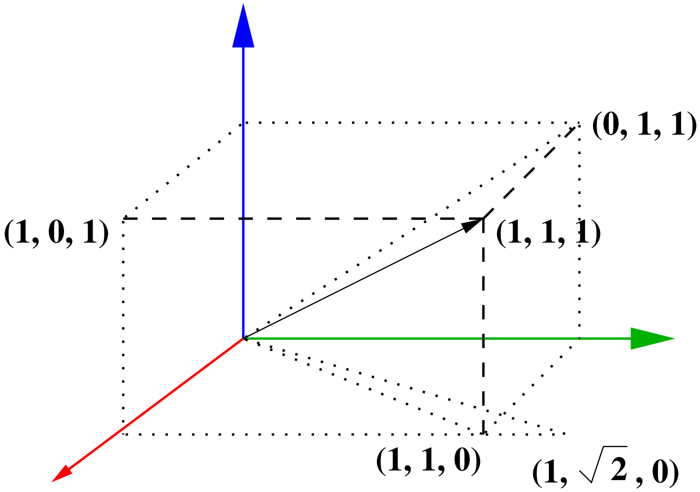
- $\mathbf{R}_\alpha(\theta) = \mathbf{R_x}(\beta)\mathbf{R_z}(\gamma) \quad \mathbf{R_x}(\theta) \quad \mathbf{R_z}(-\gamma)\mathbf{R_x}(-\beta)$



Alternate $\mathbf{R}_{\alpha\mathbf{x}}$

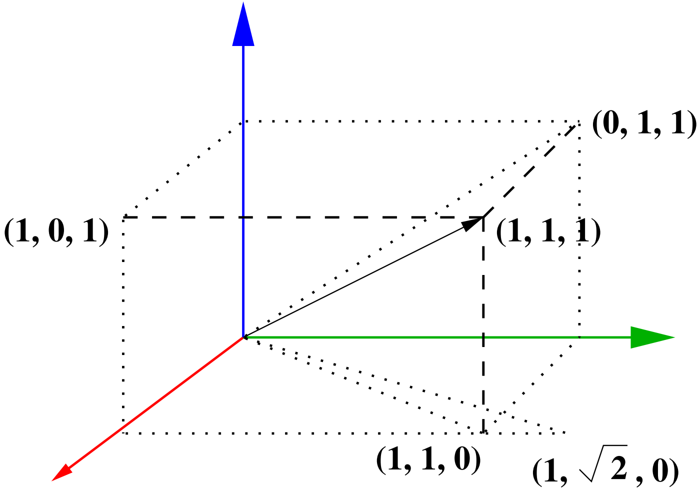
- After rotation, α will align with X-axis. Hence that is the first row $\mathbf{r_1}$ of the rotation matrix
- Find a direction orthogonal to α to be row $\mathbf{r_2}$. How?
- Take any vector $\mathbf{v}$ not parallel to α. $\mathbf{r_2} = \alpha \times \mathbf{v}$ will work!!
- Lastly, $\mathbf{r_3} = \mathbf{r_1} \times \mathbf{r_2}$ and $\mathbf{R}_{\alpha\mathbf{x}} = \begin{bmatrix} \alpha & 0 \\ \alpha \times \mathbf{v} & 0 \\ \mathbf{r_1} \times \mathbf{r_2} & 0 \\ \mathbf{0} & 1 \end{bmatrix}$
- Many possibilities, all with the same result (hopefully...)

Example: Rotation about $[1\ 1\ 1]^T$



$\beta = ?, \qquad \gamma = ?$

Example: Rotation about $[1\ 1\ 1]^T$



$\tan \beta = 1, \qquad \tan \gamma = \sqrt{2}$

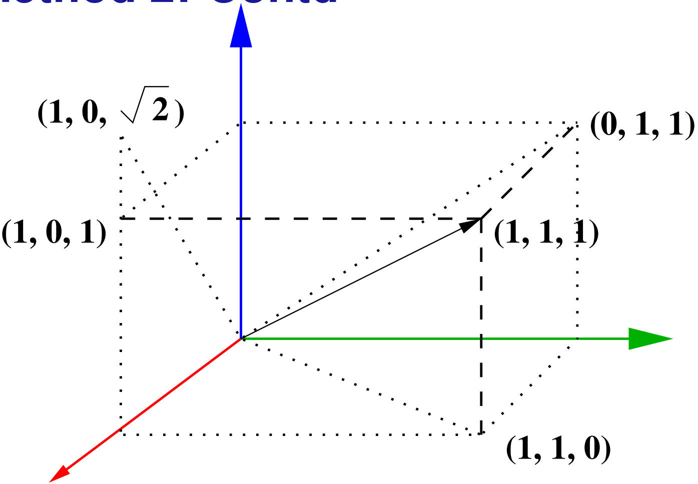
Computing $\mathbf{R}_{\alpha\mathbf{x}}$: Method 1

- Rotate by $-\pi/4$ about X. $\mathbf{R_X}(-\frac{\pi}{\mathbf{4}}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$
- $\mathbf{R_Z}(-\arctan(\sqrt{\mathbf{2}})) = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{6}} & 0 \\ \frac{-2}{\sqrt{6}} & \frac{1}{\sqrt{3}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- $\mathbf{R}^{\mathbf{I}}_{\alpha\mathbf{x}} = \mathbf{R_Z}(-\tan^{-\mathbf{1}}(\sqrt{\mathbf{2}}))\ \mathbf{R_X}(-\frac{\pi}{\mathbf{4}}) = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 \\ \frac{-2}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & 0 \\ 0 & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Computing $\mathbf{R}_{\alpha\mathbf{x}}$: Method 2

- $[\mathbf{1\ 1\ 1}]^{\mathbf{T}}$ will lie on X-axis. First row $\mathbf{r_1} = [\frac{\mathbf{1}}{\sqrt{\mathbf{3}}}\ \ \frac{\mathbf{1}}{\sqrt{\mathbf{3}}}\ \ \frac{\mathbf{1}}{\sqrt{\mathbf{3}}}]^{\mathbf{T}}$.
- Second row: $\mathbf{r_2} = \alpha \times [\mathbf{1\ 0\ 0}]^{\mathbf{T}} = [\mathbf{0}\ \ \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}\ \ \frac{\mathbf{-1}}{\sqrt{\mathbf{2}}}]^{\mathbf{T}}$
- Third row: $\mathbf{r_3} = \alpha \times [\mathbf{0}\ \ \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}\ \ \frac{\mathbf{-1}}{\sqrt{\mathbf{2}}}]^{\mathbf{T}} = [\frac{\mathbf{2}}{\sqrt{\mathbf{6}}}\ \ \frac{\mathbf{-1}}{\sqrt{\mathbf{6}}}\ \ \frac{\mathbf{-1}}{\sqrt{\mathbf{6}}}]^{\mathbf{T}}$
- $$\mathbf{R}^{\mathbf{II}}_{\alpha\mathbf{x}} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \\ \frac{2}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \mathbf{R_Y}(\tan^{-1}(\sqrt{\mathbf{2}}))\ \mathbf{R_X}(\tfrac{\pi}{\mathbf{4}})$$

$\mathbf{R}_{\alpha\mathbf{x}}$ Method 2: Contd



Question: Which vector v yields the matrix of Method 1?

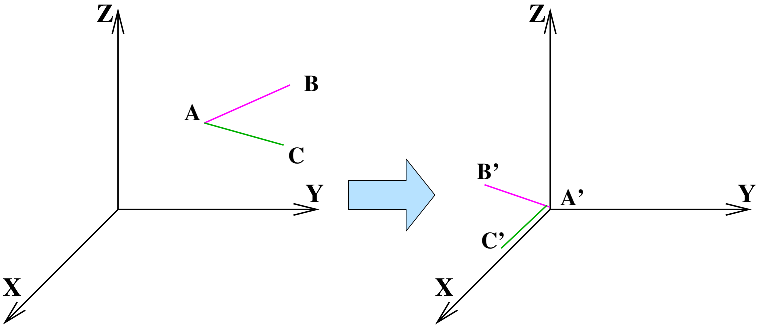
Rotation: Arbitrary Axis, Point

- An arbitrary axis may not pass through the origin.
- Translate by $\mathbf{T}$ so that it passes through the origin.
- Apply $\mathbf{R}_\alpha$.
- Translate back using $\mathbf{T}^{-1}$.
- Composite transformation: $\mathbf{T}^{-1}\ \mathbf{R}_\alpha\ \mathbf{T}$.

3D Transformations

- Many ways to *think about* a given transform.
- Ultimately, there is only one transform given the starting and ending configurations.
- Different methods let us analyze the problem from different perspectives.

Another Example



Transforming Lines

- A composite transformation can be seen as changing points in the coordinate system.
- These transformations preserve collinearity. Thus, points on a line remain on a (transformed) line.
- Take two points on the line, transform them, and compute the line between new points.
- Lines are defined as a join of 2 points or intersection of 2 planes in 3D. The same holds for transformed lines using transformed points or planes!

Transforming Planes

- A plane is defined by a 4-vector $\mathbf{n}$ (called the **normal** vector) in homogeneous coordinates.
- The plane consists of points $\mathbf{p}$ such that $\mathbf{n}^\intercal \mathbf{p} = \mathbf{0}$.
- Let $\mathbf{Q}$ transform $\mathbf{n}$ when points are transformed by $\mathbf{M}$.
- Coplanarity is preserved: $(\mathbf{Qn})^\intercal \mathbf{Mp} = \mathbf{0} = \mathbf{n}^\intercal \mathbf{Q}^\intercal \mathbf{Mp}$.
- True when $\mathbf{Q}^\intercal \mathbf{M} = \mathbf{I}$, or $\mathbf{Q} = \mathbf{M}^{-\intercal}$.
- $\mathbf{Q}$ is the Matrix of cofactors of $\mathbf{M}$ in the general case when $\mathbf{M}^{-1}$ doesn't exist.

Understanding Geometric Transformations

- Geometry transformation of objects is very important to compose graphics environments
- Understand what you want to be achieved, visualize it in your mind and compose the series of transformations
- Needs getting used to the ideas. Think about getting into a simpler situtation from the current one.