Representations and Transformations

Lydia E. Kavraki
http://www.cs.rice.edu/~kavraki
http://www.kavrakilab.org

Motion Planning: Representations and Transformations

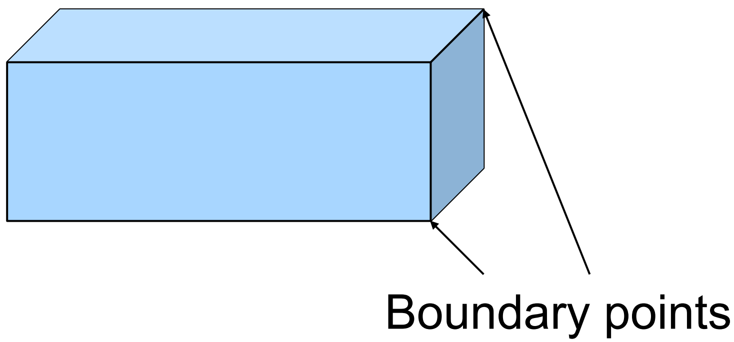
- **Objective:** Plan a path for a robot model, that takes it from some initial configuration to some desired goal configuration.
- **Constraints:** Avoid collisions in the physical environment at all times, respect physics of the problem.
- **Key questions:**
  1. How to represent the robot and the physical world?
  2. How to compute motion of the robot ?
  3. How to do all of this efficiently, and in a mathematically sound way ?

Representation of Physical Objects

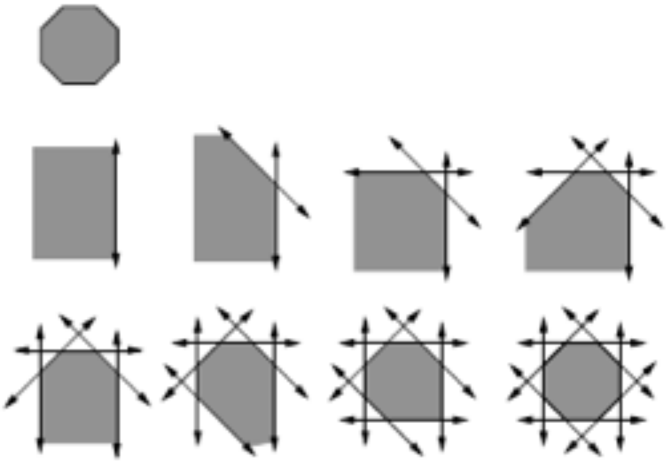
- World $\mathcal{W}$ is the physical space
- $\mathcal{W} = \mathbb{R}^2 \text{ or } \mathbb{R}^3$
- Many alternatives for representing physical objects:
  1. Boundary-point based
  2. Primitive based
  3. Polygon soups
  4. Point Clouds and others representations

Representation of Physical Objects

- **Boundary-point representation:** Represent the object by specifying details about points on the boundary

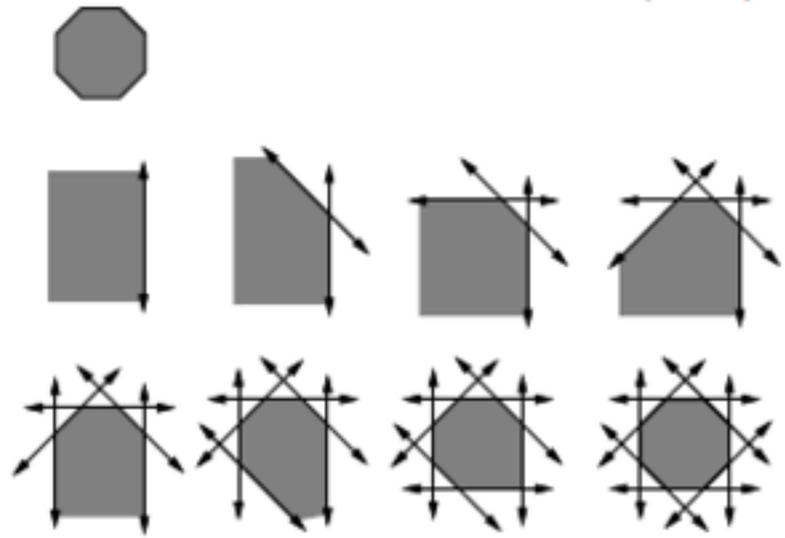


- **Primitive-based representation:** Use primitives for defining objects

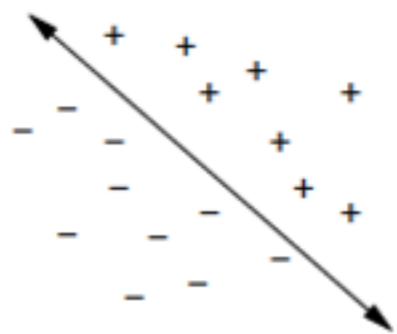


Linear Primitives

$f(x, y) = ax + by + c$ Inside: $f(x, y) \leq 0$



Intersections make convex polygons or polyhedra.



Notions of inside and outside are clear.

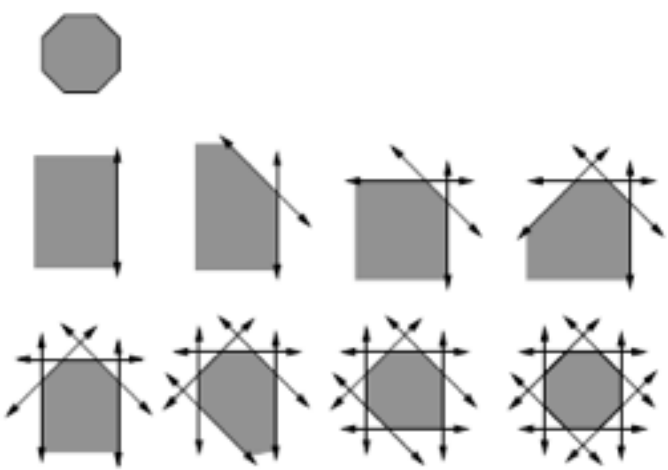
Convex vs. Nonconvex

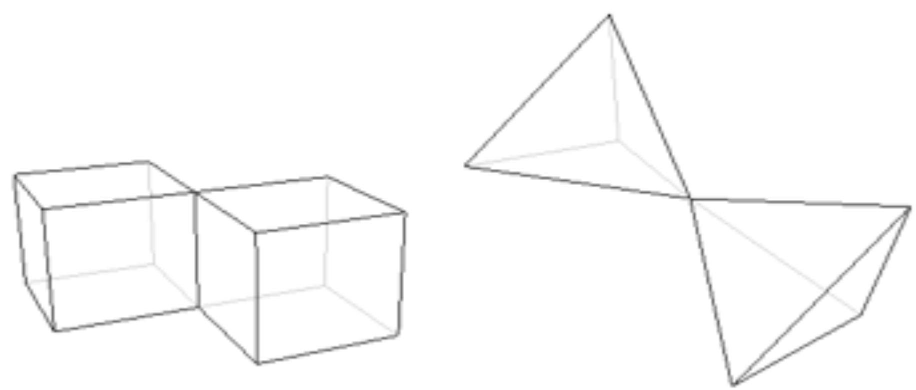
Convexity: A set S is convex if it satisfies the following:

$$x, y \in \mathcal{S} \Rightarrow \lambda x + (1 - \lambda) y \in \mathcal{S}$$

- Convex polygons (polyhedra) easy to describe using intersection of halfplanes (halfspaces)

- Nonconvex polygons are described as union of convex polygons (e.g, triangles)
- Nonconvex polyhedra are described as union of convex polyhedra (e.g., tetrahedra)

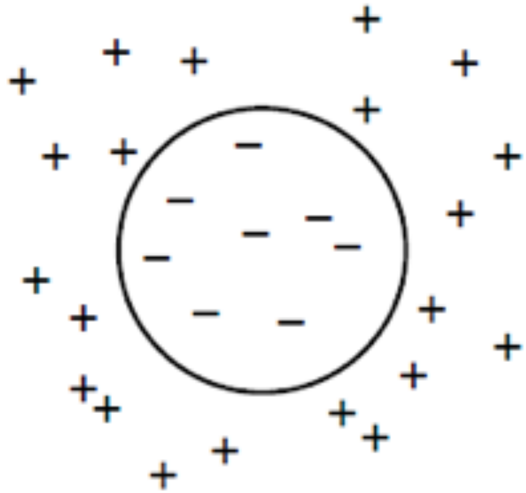




Note: The problem of decomposing a nonconvex set into convex sets is not easy in general

Semi Algebraic Sets

- Polygons and polyhedra use linear primitives (e.g., halfspaces)
- **Algebraic set:** A generalization of polygons and polyhedra that uses **nonlinear primitives**



$$H = \{(x, y) \in \mathcal{W} \mid f(x, y) \leq 0\}, \text{ where,}$$

$$f(x, y) = (x - 2)^2 + (y - 2)^2 - 4$$

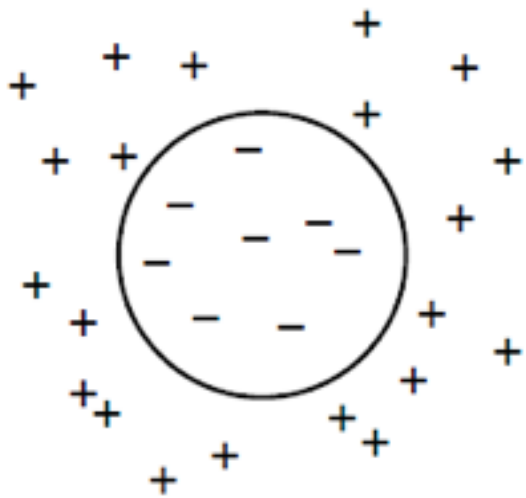
Semi Algebraic set: Sets constructed using finite intersection and unions of algebraic sets

?

Center of outer circle=(x1,y1); Radius = r1
Center of left eye: (x2,y2); Radius = r2
Center of right eye: (x3,y3); Radius = r3
Mouth is an ellipse: Center = (x4,y4); Major axis “a”, Minor axis “b”

Semi Algebraic Sets

- Polygons and polyhedra use linear primitives (e.g., halfspaces)
- **Algebraic set:** A generalization of polygons and polyhedra that uses **nonlinear primitives**



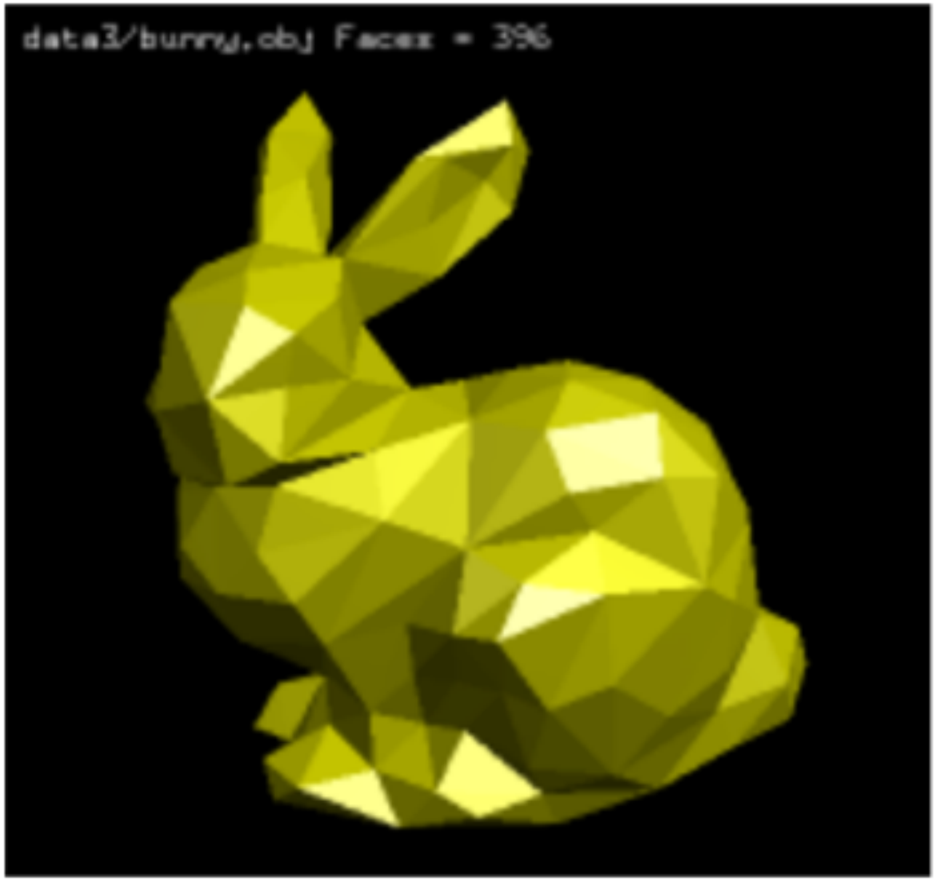
$$H = \{(x, y) \in \mathcal{W} \mid f(x, y) \leq 0\}, \text{ where,}$$

$$f(x, y) = (x-2)^2 + (y-2)^2 - 4$$

Semi Algebraic set: Sets constructed using finite intersection and unions of algebraic sets

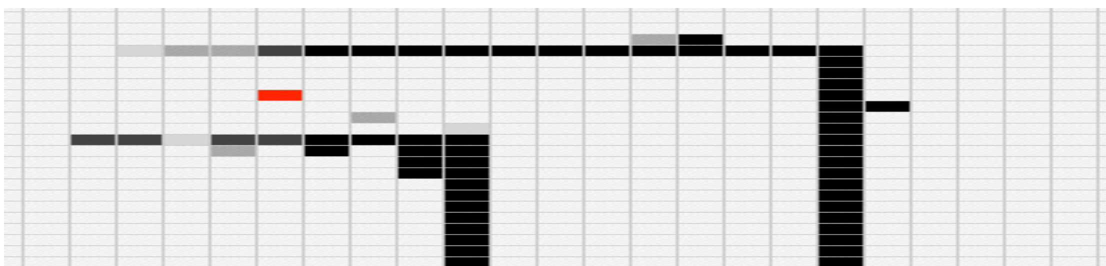
$$\begin{aligned}
f_1 &= x^2 + y^2 - r_1^2, \\
f_2 &= -\left((x - x_2)^2 + (y - y_2)^2 - r_2^2\right), \\
f_3 &= -\left((x - x_3)^2 + (y - y_3)^2 - r_3^2\right), \\
f_4 &= -\left(x^2/a^2 + (y - y_4)^2/b^2 - 1\right). \\
\mathcal{O} &= H_1 \cap H_2 \cap H_3 \cap H_4.
\end{aligned}$$

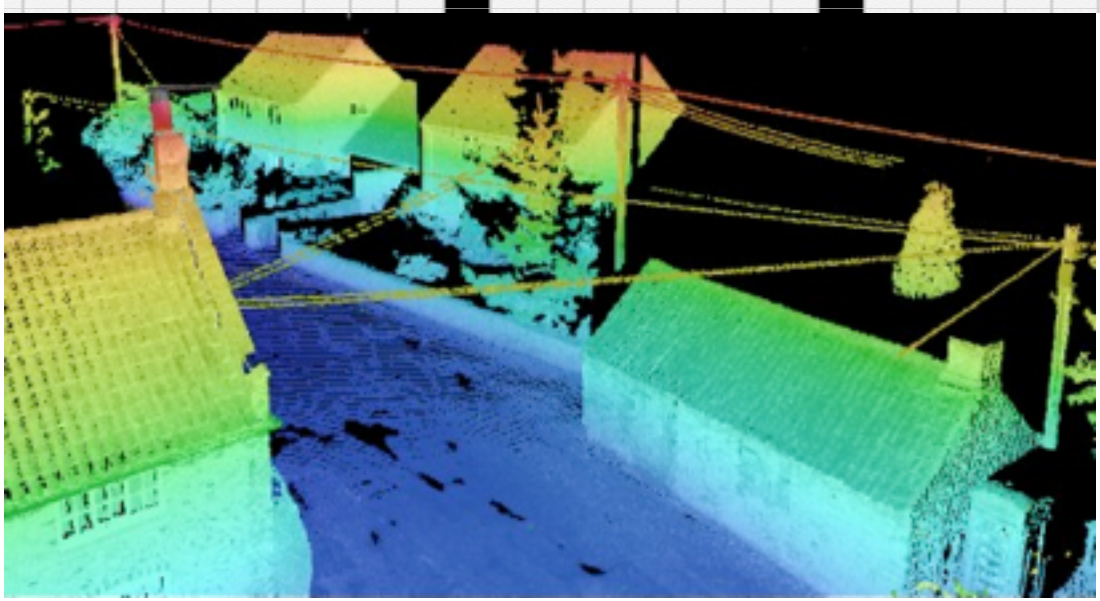
Polygon Soups



Objects identified using sensors

- Real world objects are usually detected using perception module of a robot
- Obstacles are usually detected using variety of sensors
- Obstacles are described using point cloud, occupancy grid, bitmaps



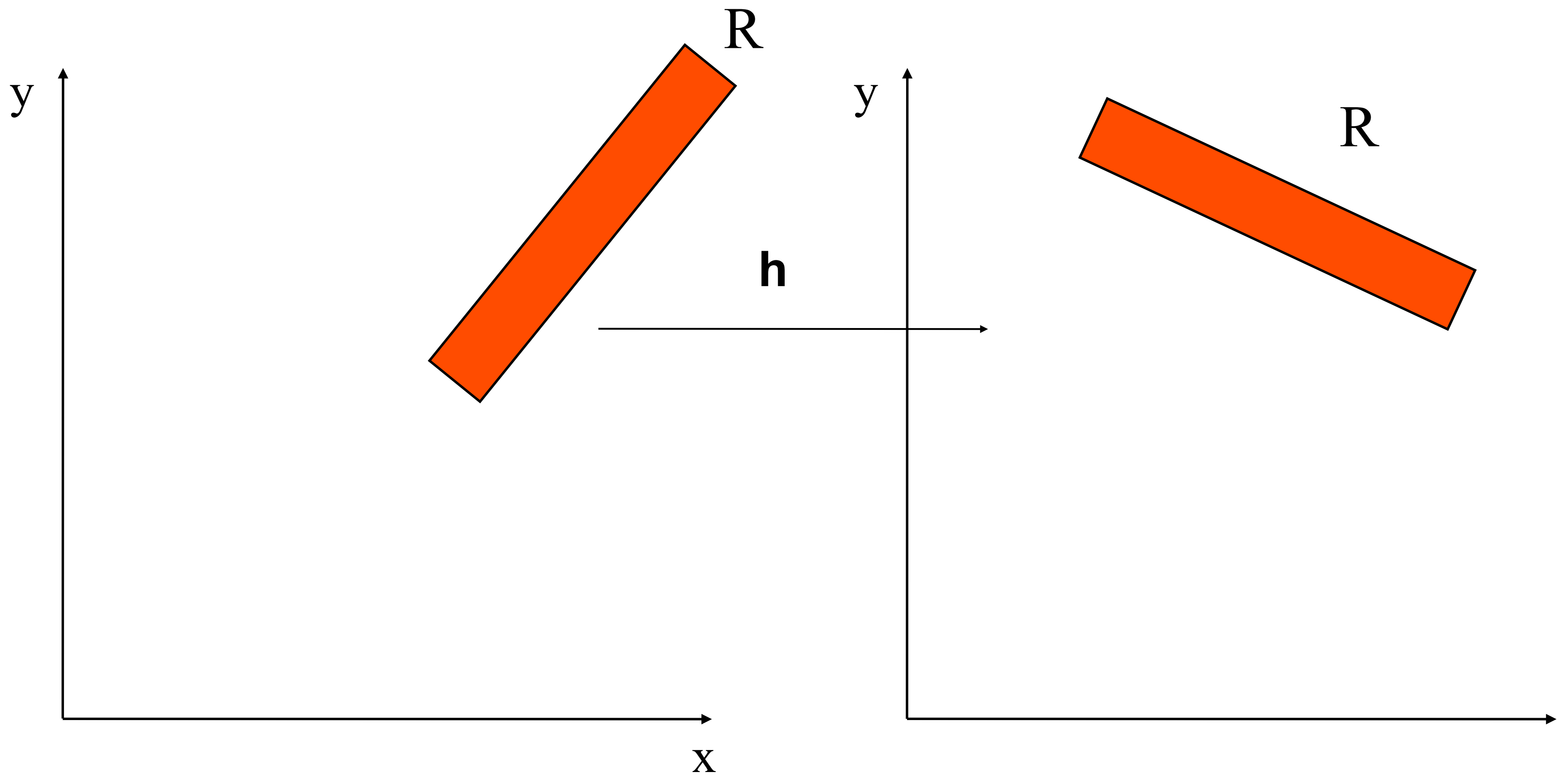


Rigid Body Transformations

- **Rigid body model:** Model of physical object with no deformation
- **Transformation of robot model:** $h(\mathcal{A}) = \{h(a) \in \mathcal{W} | a \in \mathcal{A}\}$

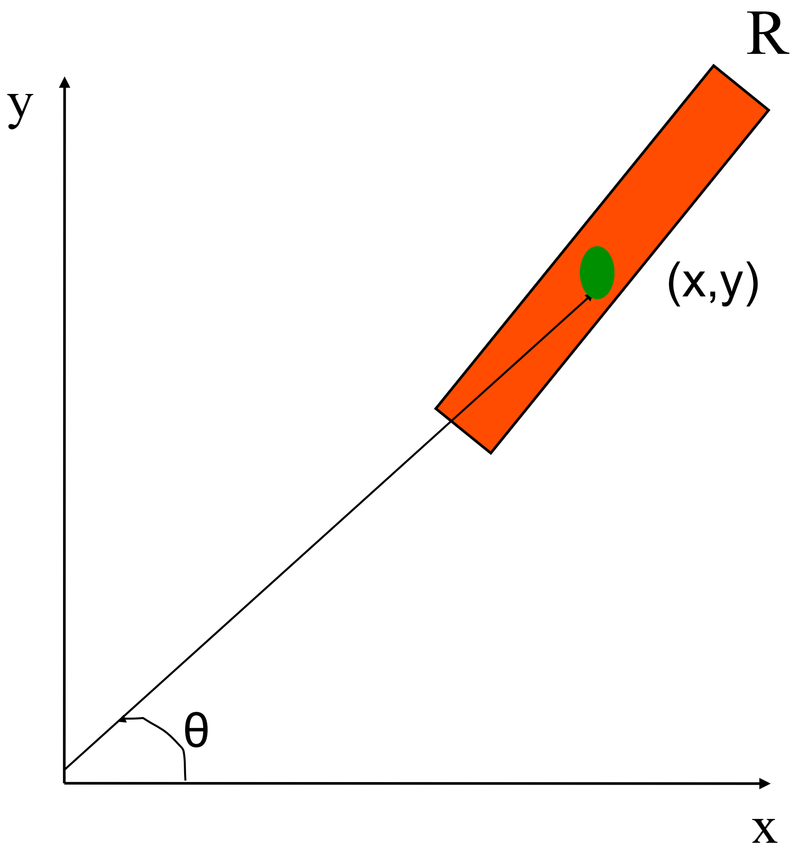
What is a rigid body transformation?

- A transformation ***h*** is a rigid body transformation, if it satisfies the following two conditions:
  1. Distance between points on the object is preserved
  2. Relative orientation does not change, i.e., no mirror images



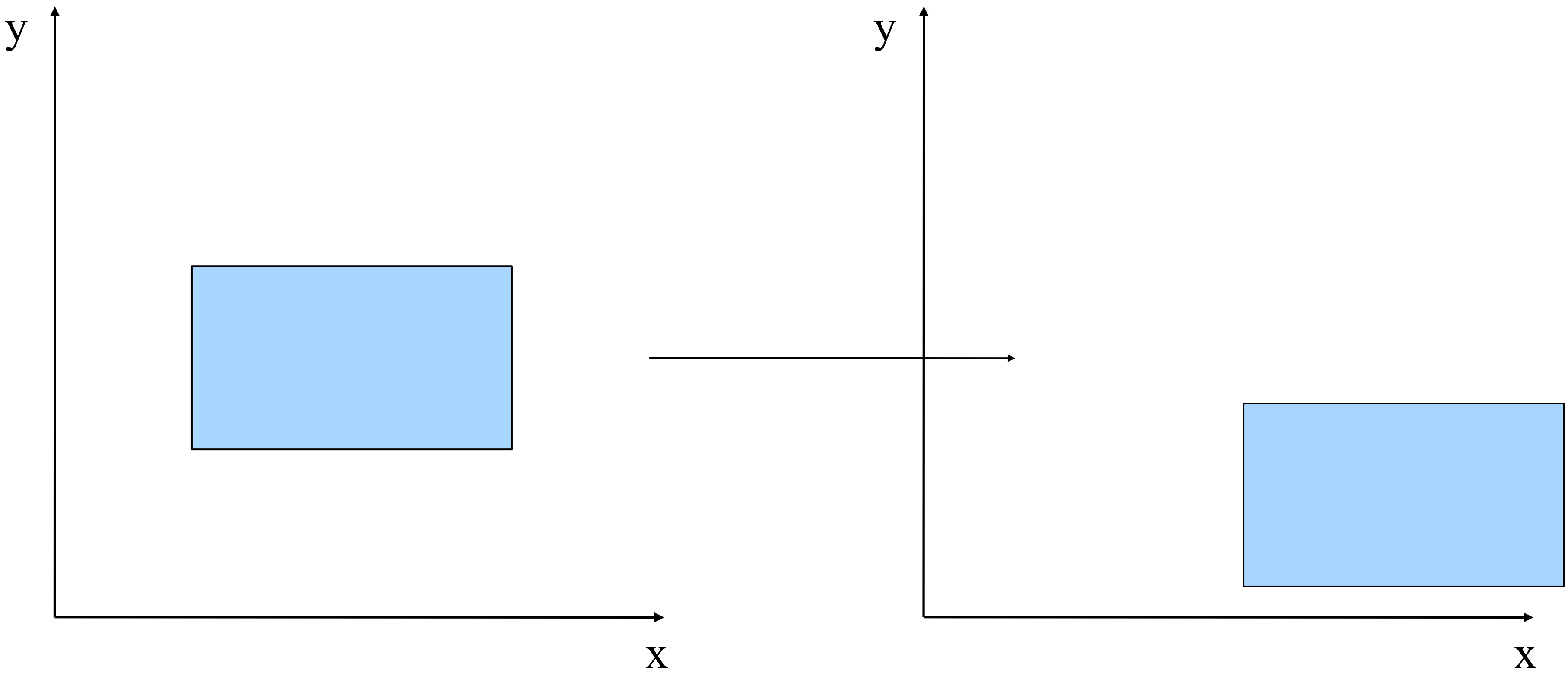
Rigid Body Transformations in 2D

- A rigid body in 2D is described completely by position and orientation of its reference frame **w.r.t a global coordinate frame**
- Two kinds of transformations possible: Translation and Rotation



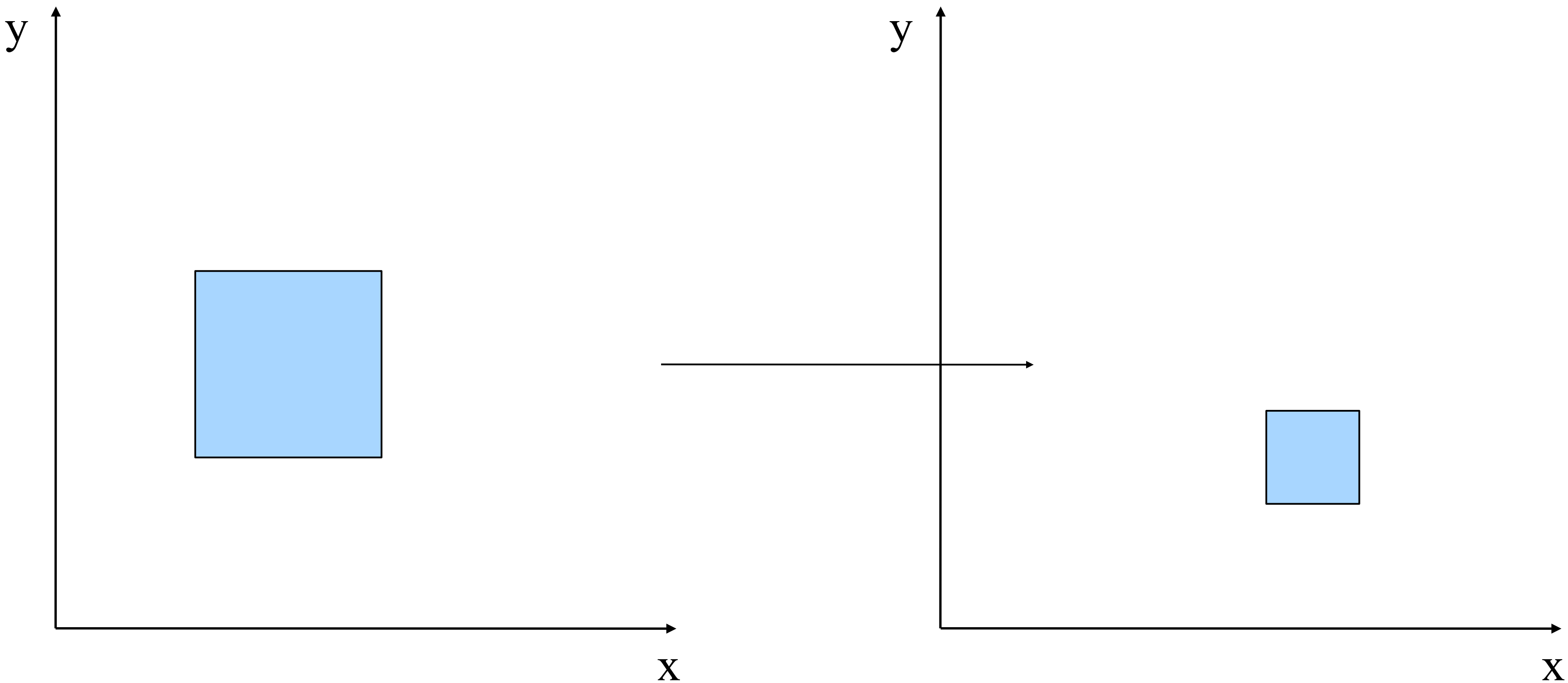
Rigid Body Translations in 2D

- **Rigid body translation** described by: $h(x, y) = (x + x_t, y + y_t)$
- **Boundary representation:** Apply ***h*** to each boundary point



Rigid Body Translations in 2D

- **Rigid body translation** described by: $h(x, y) = (x + x_t, y + y_t)$
- **Boundary representation:** Apply ***h*** to each boundary point

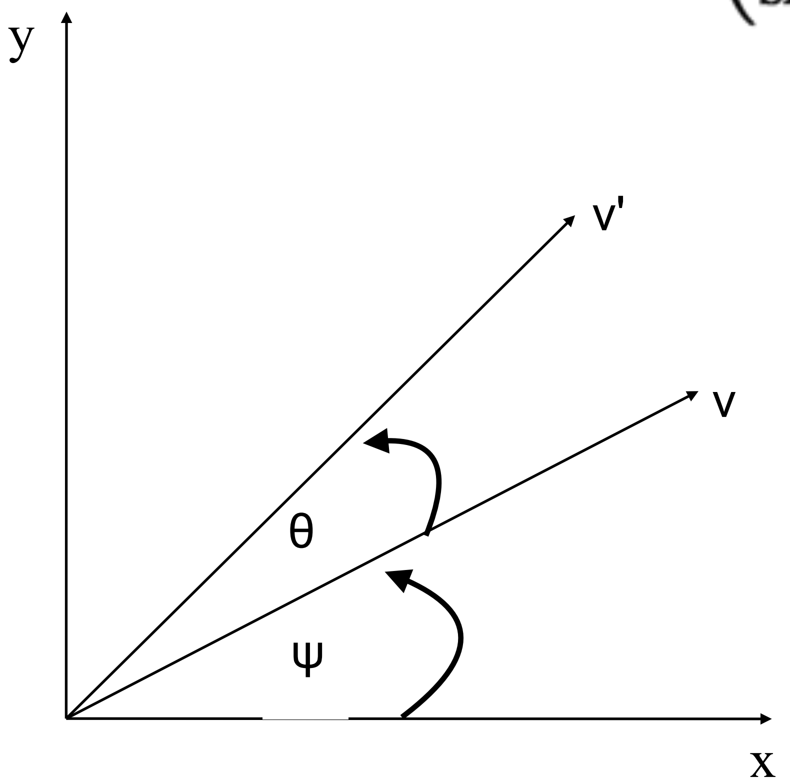


Question: Does the above picture indicate a rigid body translation?

Rigid body Rotations in 2D

- Rotation about origin is given by the transformation:

$$h(\theta) = R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$



$$\begin{aligned} v &= (r\cos(\psi), r\sin(\psi)) \\ v' &= (r\cos(\psi+\theta), r\sin(\psi+\theta)) \\ &= \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ &= R(\theta)v \end{aligned}$$

- Effect of successive rotations is given by multiplication:

$$v'' = R(\theta_2)v' = R(\theta_2)R(\theta_1)v \quad = R(\theta_2 + \theta_1)v$$

General Rigid Body Motions in 2D

- Given a vector v representing a point on rigid body,

$$(t, R(\theta)) : v \to R(\theta)v + t$$ **Rotate followed by translate !!**

- Same transformation in homogeneous coordinates:

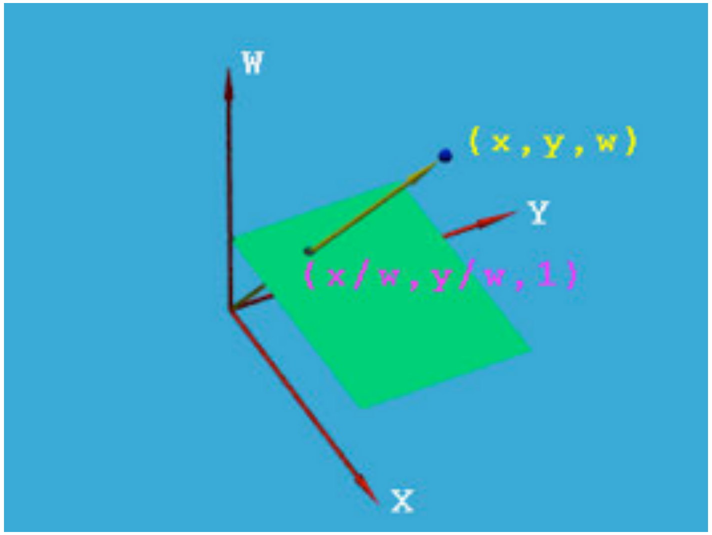
$$\begin{pmatrix} R(\theta) & t \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} v \\ 1 \end{pmatrix}$$

It is the same thing:

$$\left(\begin{array}{c|c} R(\theta) & t \\ \hline 0 & 1 \end{array} \right) \cdot \begin{pmatrix} v \\ 1 \end{pmatrix} = \begin{pmatrix} R(\theta) \cdot v + t \\ 1 \end{pmatrix}$$

Homogenous Coordinates

- **"∞" cannot be represented in Euclidean coordinate system**
- Lot of geometric concepts greatly simplified if ∞ can be represented
- Advantage that coordinates of points at infinity can also be represented using finite coordinates
- A triplet (x,y,w) represents homogeneous coordinates of a point (x/w,y/w) with w ≠ 0
- At least one of x,y,w is always nonzero
- (x,y,w) is same as (10x,10y, 10w): multiplying by a non-zero factor does not change the coordinate
- (x,y,0) represents ∞ in the direction of (x,y)



General Rigid Body Motions in 2D

- Given a vector v representing a point on rigid body,

$$(t, R(\theta)) : v \rightarrow R(\theta)v + t$$ **Rotate followed by translate !!**

- Same transformation in homogeneous coordinates:

$$\begin{pmatrix} R(\theta) & t \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} v \\ 1 \end{pmatrix}$$

- Successive transformations can be easily computed in homogeneous coordinates:

$$\text{Rotate by } \theta_1 \rightarrow \text{Translate by } t_1 \rightarrow \text{Rotate by } \theta_2 \rightarrow \text{Translate by } t_2$$

$$= \begin{pmatrix} R(\theta_2) & t_2 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} R(\theta_1) & t_1 \\ 0 & 1 \end{pmatrix}$$

Question: How to compute rotations about arbitrary point?

Homogeneous coordinates are convenient

Rotate by θ_1
Translate by t_1
Rotate by θ_2
Translate by t_2

Previously: $$R(\theta_2)\Big(R(\theta_1)\,v + t_1\Big) + t_2 =$$

$$R(\theta_2)R(\theta_1)\,v + R(\theta_2)t_1 + t_2$$

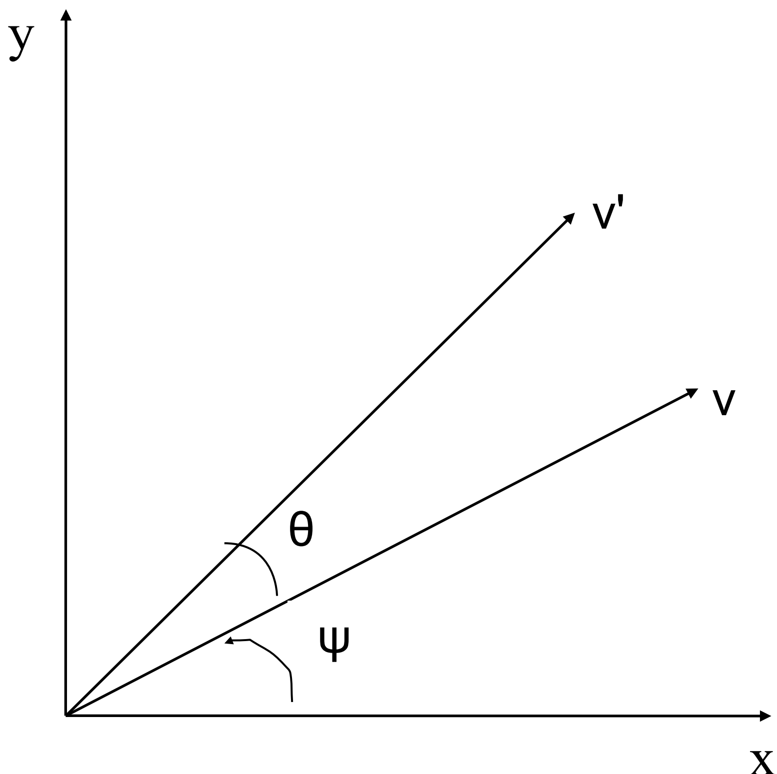
Now:

$$\left(\begin{array}{c|c} R(\theta_2) & t_2 \\ \hline 0 & 1 \end{array}\right)\left(\begin{array}{c|c} R(\theta_1) & t_1 \\ \hline 0 & 1 \end{array}\right) =$$

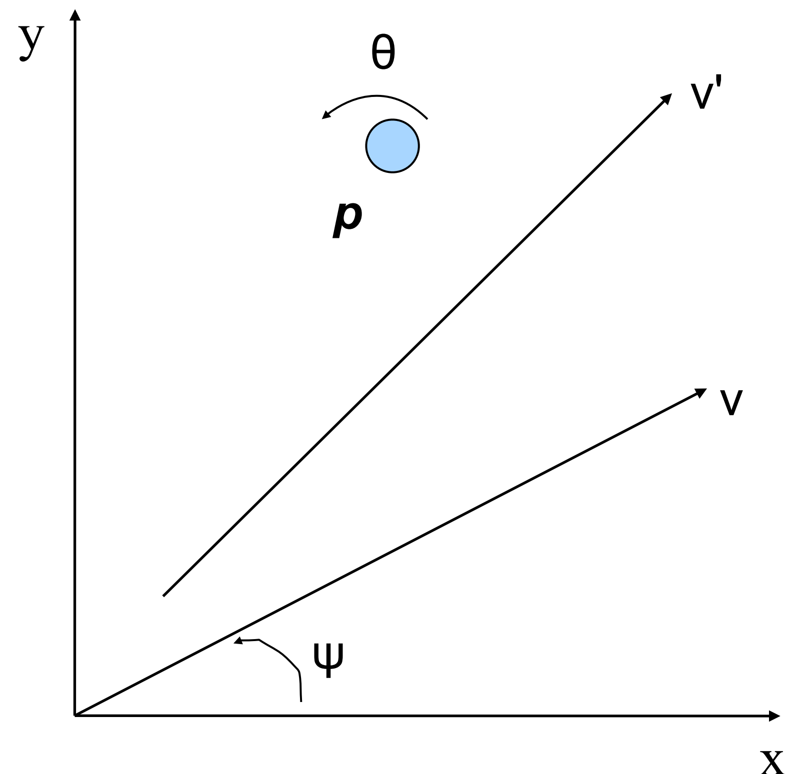
$$\left(\begin{array}{c|c} R(\theta_2)R(\theta_1) & R(\theta_2)t_1 + t_2 \\ \hline 0 & 1 \end{array}\right)$$

2D Rotations about an arbitrary point

- **Problem:** Find rotation matrix corresponding to rotation about an arbitrary point ***p***
- ***Solution hint:*** Use homogeneous coordinates and framework developed so far.



Rotation about origin



Rotation about ***p***

2D Rotations about an arbitrary point

- **Problem:** Find rotation matrix corresponding to rotation about an arbitrary point ***p***
- ***Solution hint:*** Use homogeneous coordinates and framework developed so far.
- **Solution:** Can be computed by using the following sequence of transformations:

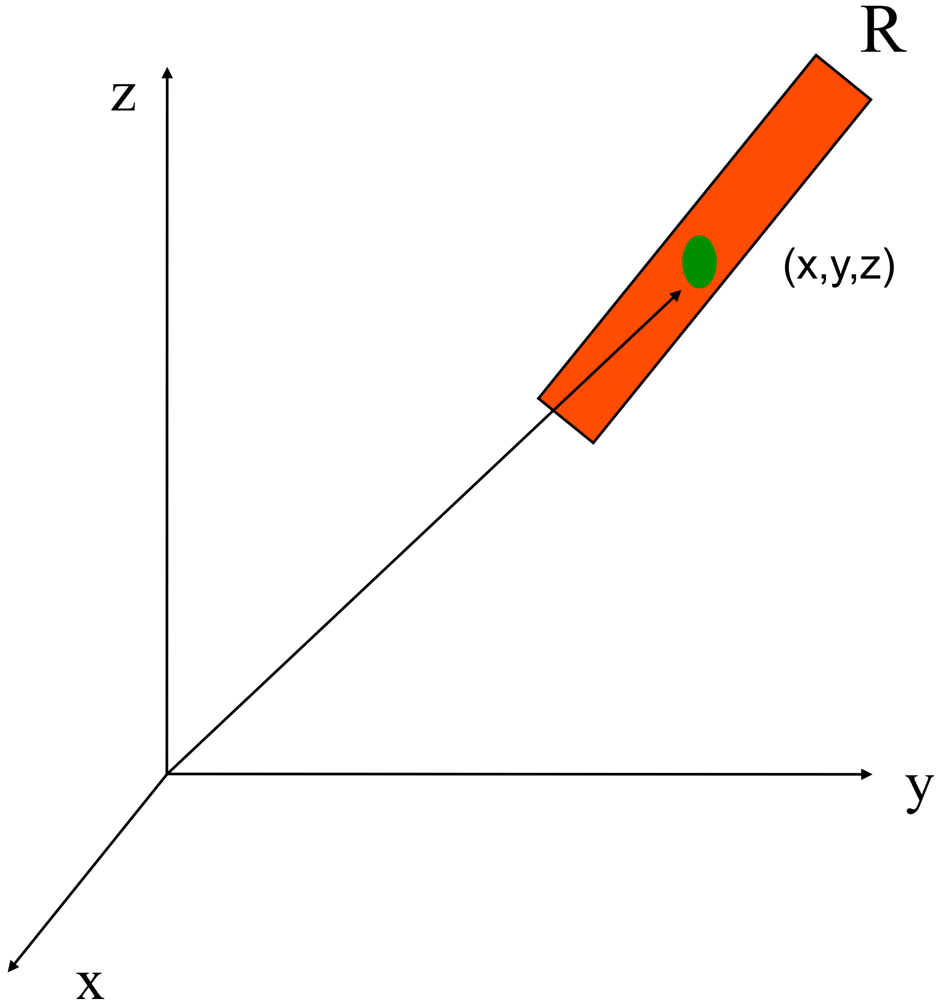
$$\text{Translate by } -p \to \text{Rotate by } \theta \to \text{Translate by } p$$

$$= \begin{pmatrix} I & p \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} R(\theta) & 0 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} I & -p \\ 0 & 1 \end{pmatrix}$$

Homogenous coordinates simplify things for computations in path planning

Rigid Body Transformations in 3D

- A rigid body in 3D is described completely by position and orientation of its reference frame **w.r.t a global coordinate frame**
- Two kinds of transformations possible: Translation and Rotation

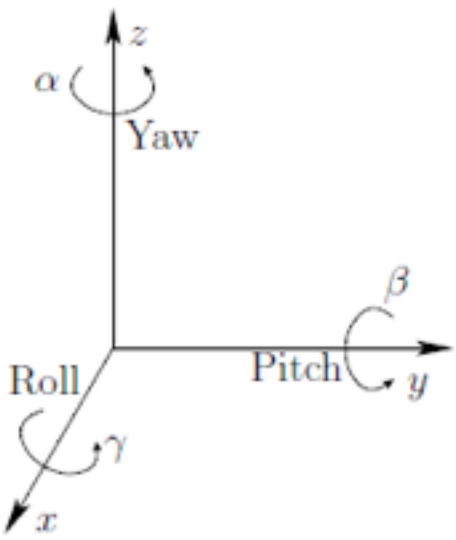


Rigid Body Translations in 3D

- **Rigid body translation** described by: $h(x,y,z) = (x + x_t, y + y_t, z + z_t)$
- **Boundary representation:** Apply ***h*** to each boundary point

Rigid body Rotations in 3D

- Rotation is considered about the origin
- There are 3 axis of rotations: x, y, z
- Each rotation is **counter**clockwise
- A commonly used convention is yaw-pitch-roll referring to rotations about z, y, x axis respectively.



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

yaw

$$R_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}$$

pitch

$$R_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}$$

roll

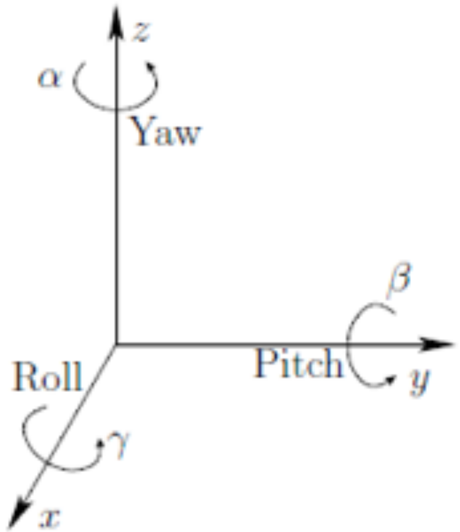
Successive Rotations in 3D

- Yaw, Pitch, Roll can be used to place 3D body in any orientation
- Successive rotations applied the same way as in 2D
- Order of rotations is important

Rotate by γ about x axis: $\boldsymbol{R_x(\gamma)}$

Rotate by β about y axis: $\boldsymbol{R_y(\beta)}$

Rotate by α about z axis: $\boldsymbol{R_z(\alpha)}$



$$R(\alpha,\beta,\gamma) = R_z(\alpha)\,R_y(\beta)\,R_x(\gamma) =$$
$$\begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}$$

(Note)

"Euler angles"

Euler's rotation theorem states that any rotation can be represented by no more than three rotations about coordinate axes, - no two successive rotations are about the same axis

Hence we can define

XYX, XZX, YXY, YZY, ZXZ, ZYZ

XYZ, XZY, YZX, YZX, ZYX, ZYX

typically referred as the yaw,pitch,roll

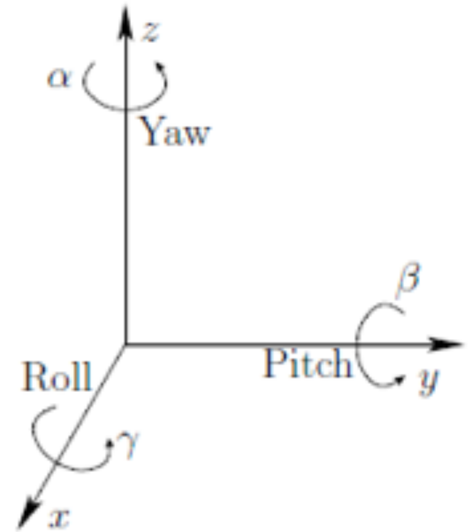
Successive Rotations in 3D

- Roll, Pitch, Yaw can be used to place 3D body in any orientation
- Successive rotations applied the same way as in 2D
- Order of rotations is important

Rotate by γ about x axis: $\boldsymbol{R_x(\gamma)}$

Rotate by β about y axis: $\boldsymbol{R_y(\beta)}$

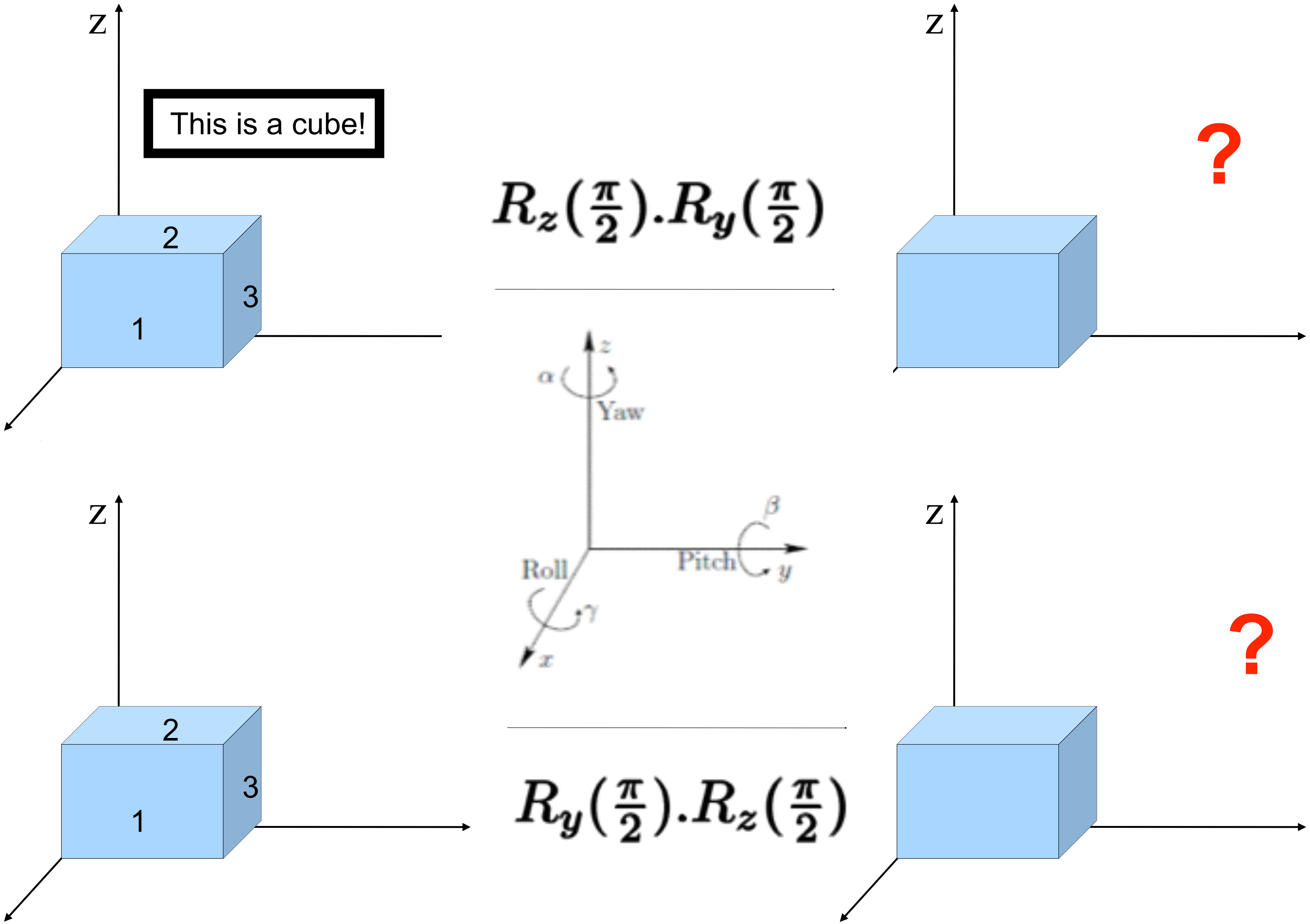
Rotate by α about z axis: $\boldsymbol{R_z(\alpha)}$



$$R(\alpha,\beta,\gamma) = R_z(\alpha)\,R_y(\beta)\,R_x(\gamma) = \begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}$$

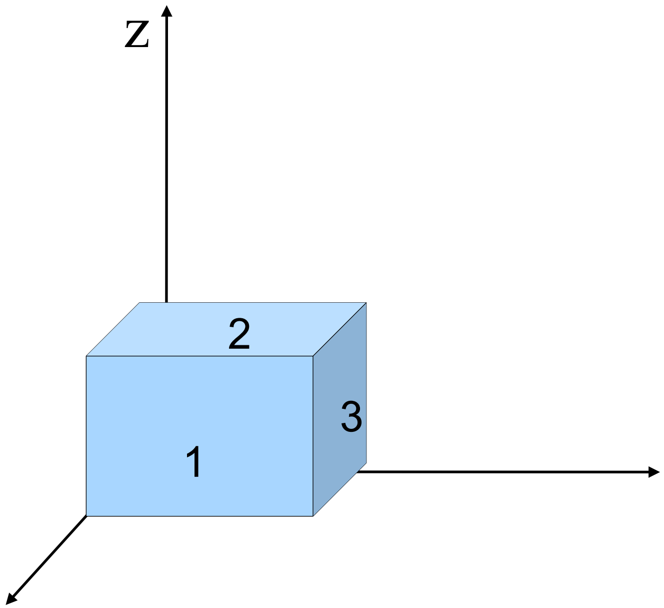
Question: Do rotations in 3D commute ?

Do 3D rotations commute?

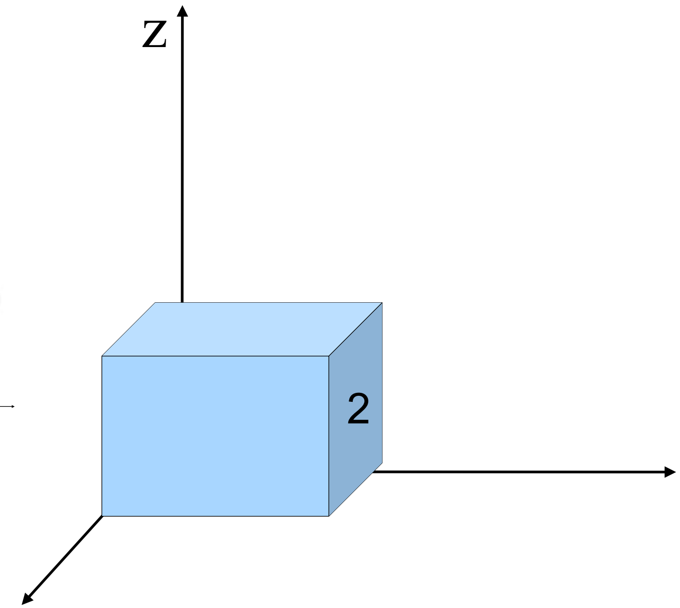


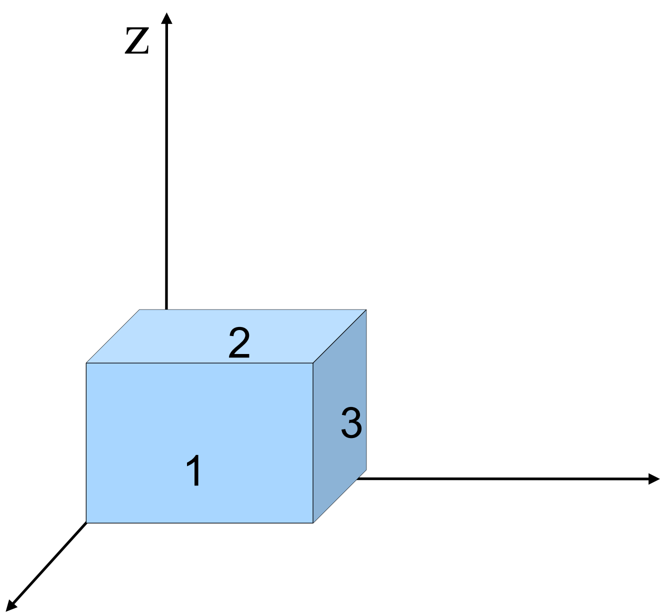
Try to estimate which side will be where side 3 is now

Do 3D rotations commute?

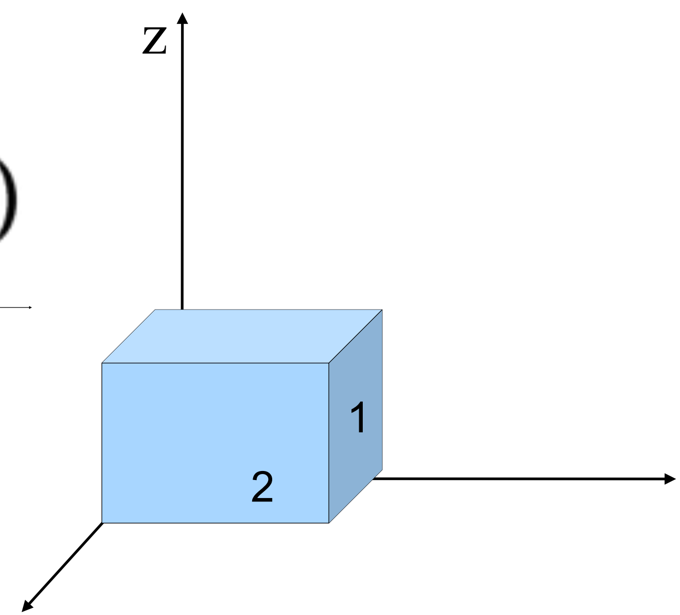


$$R_z(\tfrac{\pi}{2}).R_y(\tfrac{\pi}{2})$$





$$R_y(\tfrac{\pi}{2}).R_z(\tfrac{\pi}{2})$$



3D rotations do not commute while 2D rotations do commute !

General Rigid Body Motions in 3D

- Given a vector v representing a point on rigid body,

$$(R(\alpha,\beta,\gamma),t) : v \rightarrow R(\alpha,\beta,\gamma)v + t$$

- Same transformation in homogeneous coordinates:

$$\begin{pmatrix} R(\alpha,\beta,\gamma) & t \\ \mathbf{0} & 1 \end{pmatrix} \cdot \begin{pmatrix} v \\ 1 \end{pmatrix}$$

- Successive transformations can be easily computed in homogeneous coordinates (yaw,pitch,roll):

$$\text{Rotate by } (\gamma_1 \rightarrow \beta_1 \rightarrow \alpha_1) \rightarrow \text{Translate by } t_1$$

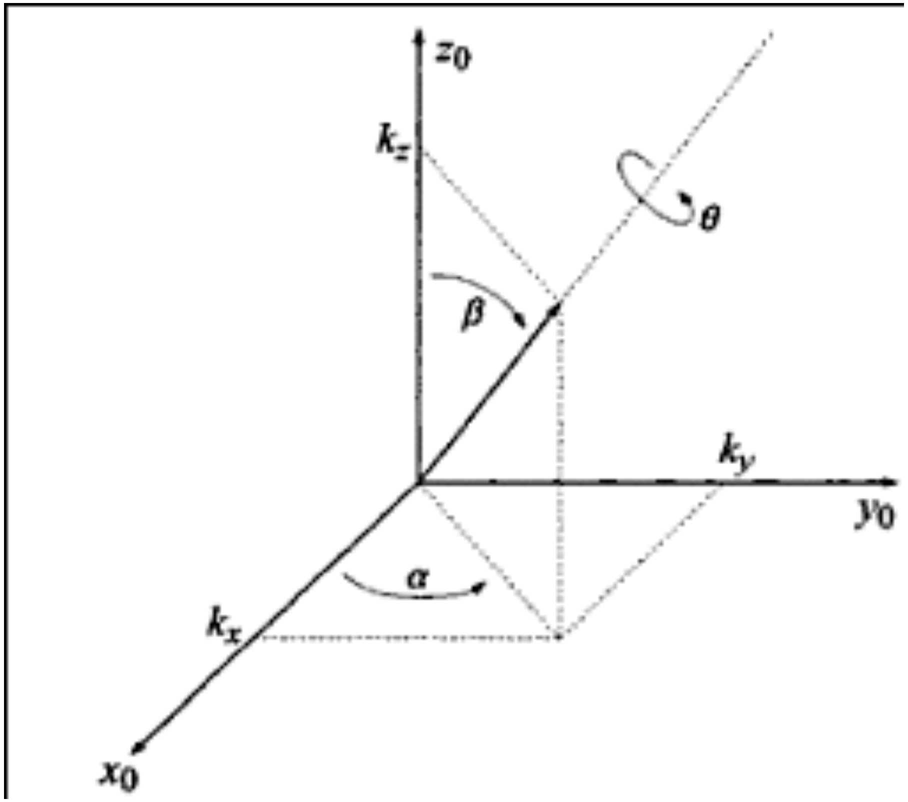
$$\text{Rotate by } (\gamma_2 \rightarrow \beta_2 \rightarrow \alpha_2) \rightarrow \text{Translate by } t_2$$

$$= \begin{pmatrix} R(\alpha_2,\beta_2,\gamma_2) & t_2 \\ \mathbf{0} & 1 \end{pmatrix} \cdot \begin{pmatrix} R(\alpha_1,\beta_1,\gamma_1) & t_1 \\ \mathbf{0} & 1 \end{pmatrix} \cdot \begin{pmatrix} v \\ 1 \end{pmatrix}$$

Question: How to compute rotations about arbitrary axis?

3D Rotations about an arbitrary axis

- **Problem:** Find rotation matrix corresponding to rotation about an arbitrary axis ***k*** ***(also called Axis-Angle Parametrization)***
- ***Solution: Transform to the (x,y,z) coordinate axis, apply the rotation, and then reverse the transformations.***



Unit vector along rotation axis $= (k_x, k_y, k_z)$

$$R_k(\theta) = R_z(\alpha)R_y(\beta)R_z(\theta)R_y(-\beta)R_z(-\alpha)$$