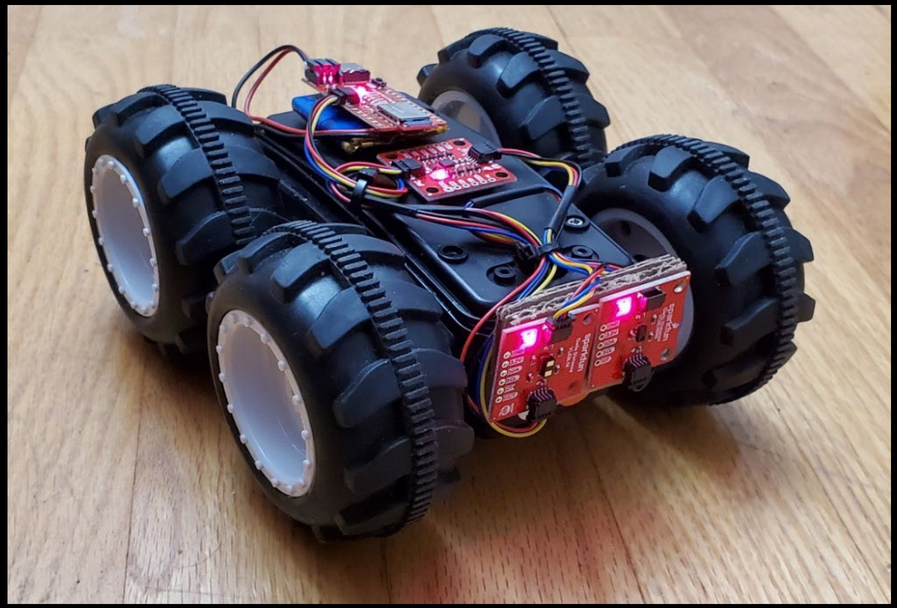
Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

ECE 4160/5160
MAE 4910/5910

Fast Robots
T-matrices

Robot Configurations

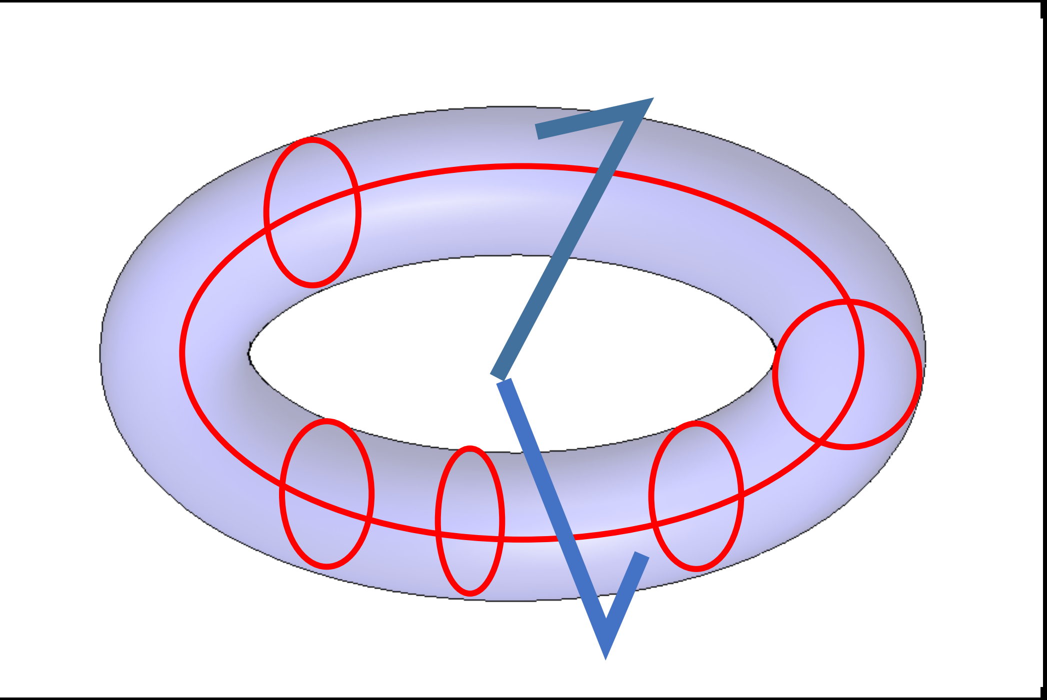
- Objective: Coordinate transformations for robotics
  - "Rigid-body kinematics"
- Robot configuration specifies all points on the robot
- The robot C-space is the space of all configurations
- The DOF is the dimension of the C-space



What is the DOF of these?

Configuration, Configuration space, Degrees of Freedom

- 2 DOF robot arm
  - C-space: 2 angles
  - J-space: Surface of a torus



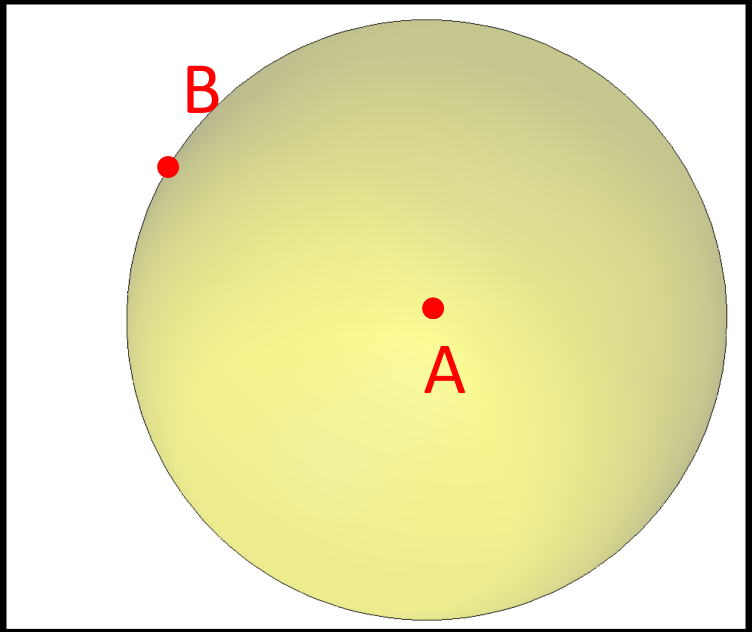
- *Every robot configuration has a unique point on the torus*
- *Every point on the torus is a unique robot configuration*

Robot Configurations

- Point A: {x, y, z}

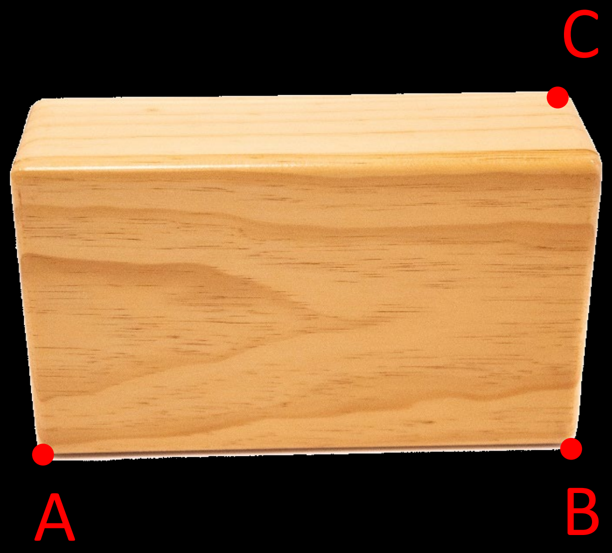
Robot Configurations

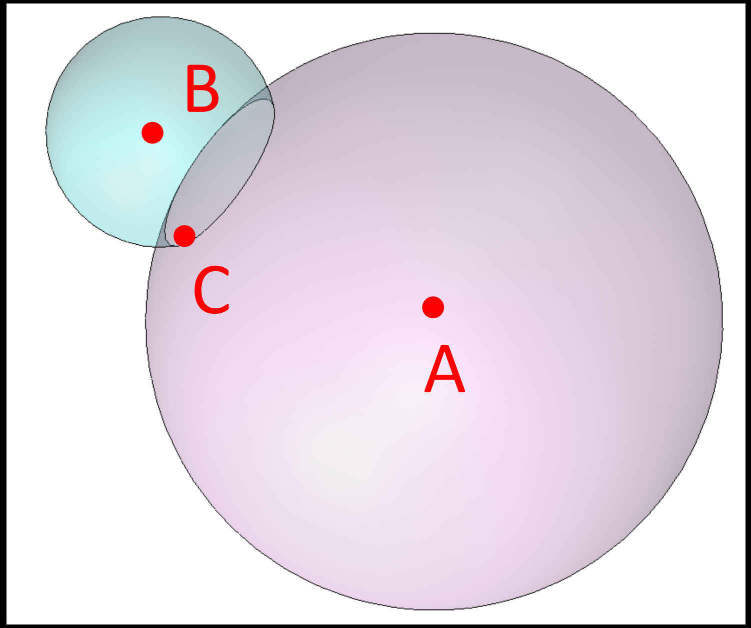
- Point A: {x, y, z}
- Point B: {θ, ϕ}



Robot Configurations

- Point A: {x, y, z}
- Point B: {θ, ϕ}
- Point C: {ψ}
- A rigid body in 3D has 6 DOF
- A rigid body in 2D has 3 DOF
- A rigid body in 4D has 10 DOF



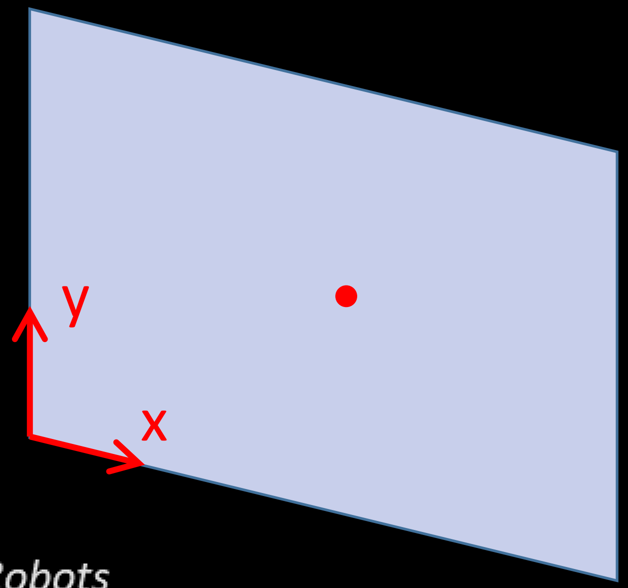


Point	Coords	Ind. constraints	Real freedoms
A	3	0	3
B	3	1	2
C	3	2	1
D	3	3	0
Total			6

DOF = Σ(freedoms of points – no. of independent constraints)

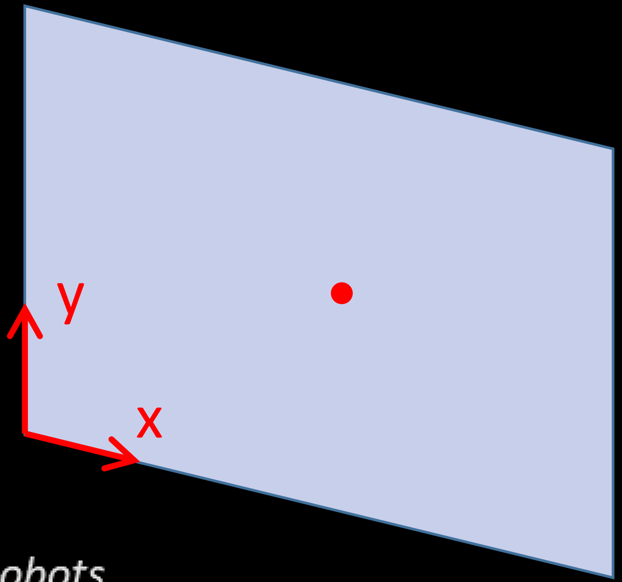
Topology Representation

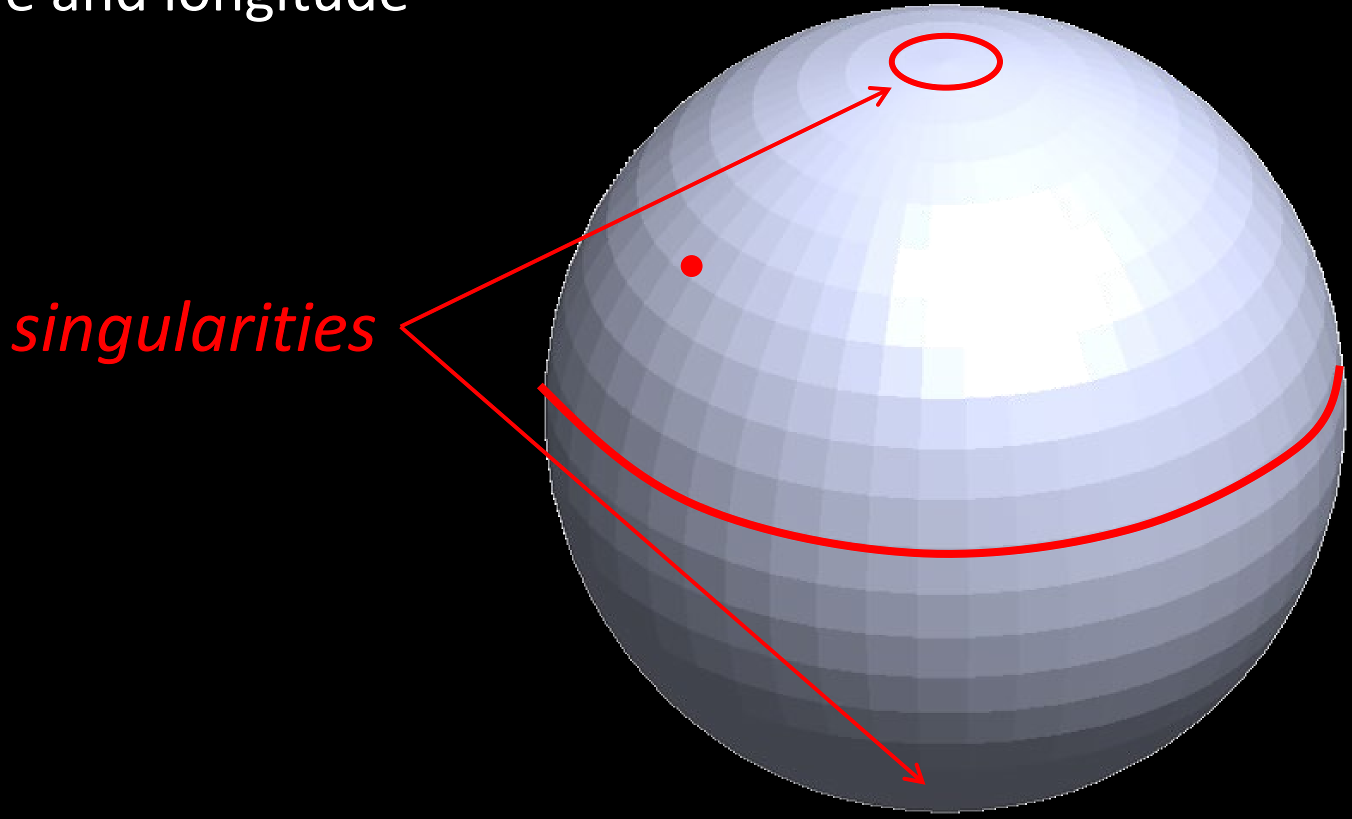
- Point on a plane
  - Origin and 2 orthogonal coordinate axis



Topology Representation

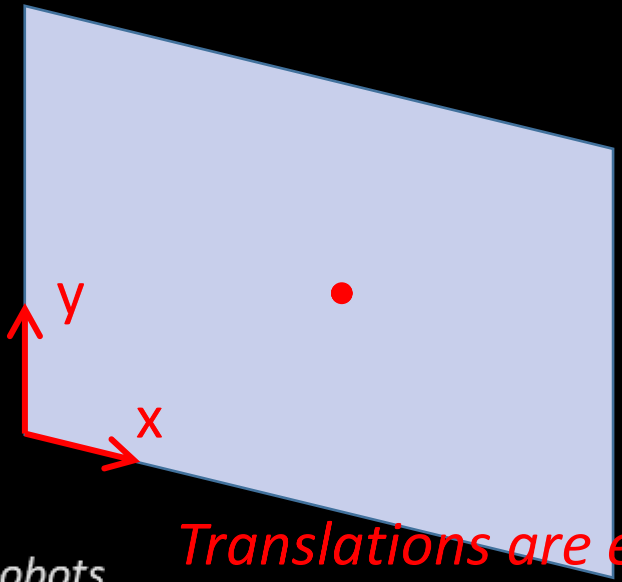
- Point on a plane
  - Origin and 2 orthogonal coordinate axis
- Points on the surface of a sphere
  - "Explicit representation": Latitude and longitude

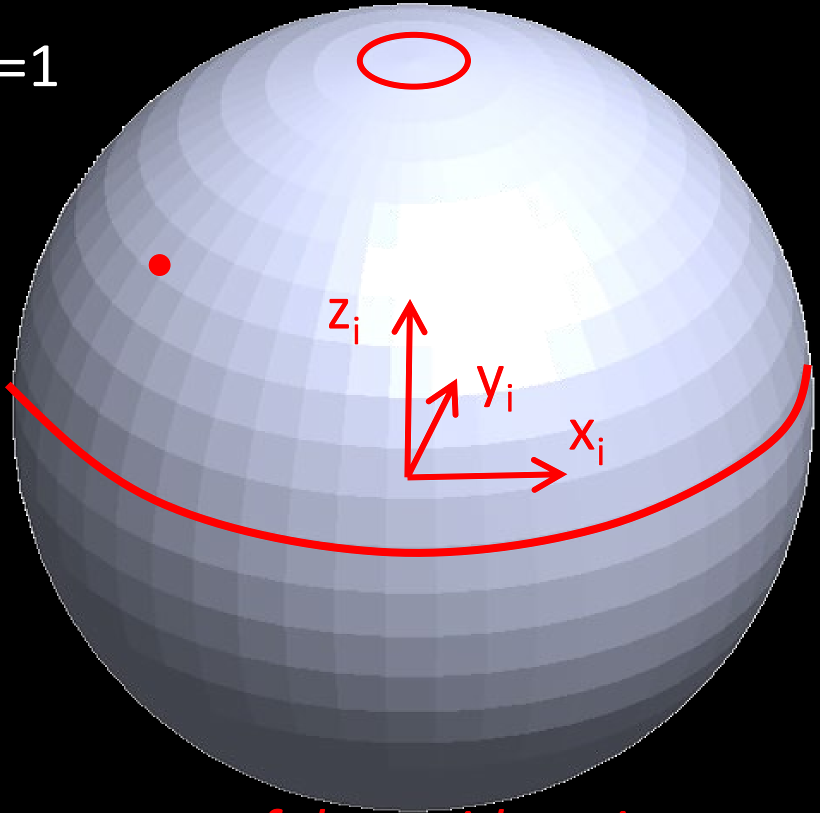




Topology Representation

- Point on a plane
  - Origin and 2 orthogonal coordinate axis
- Points on the surface of a sphere
  - "Explicit representation": Latitude and longitude
  - "Implicit representation": {X, Y, Z}, such that $x^2+y^2+z^2 = 1$
    - Slightly more complex, but singularity free!
      - 3D → Rotation matrix

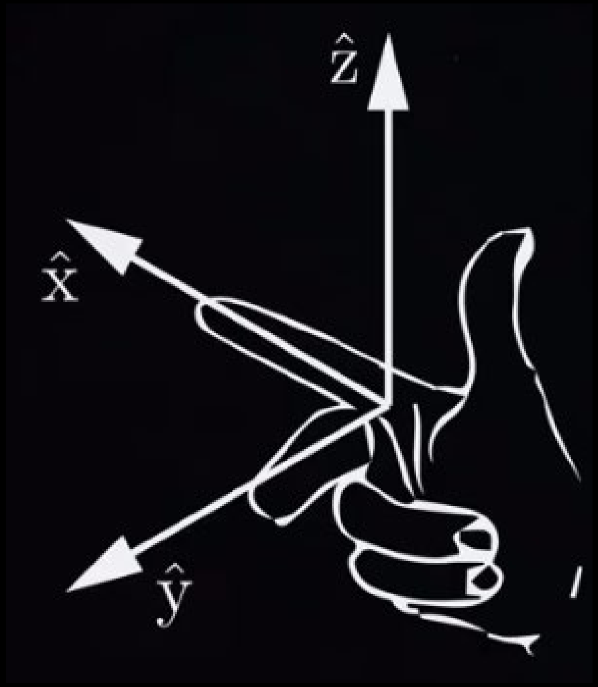


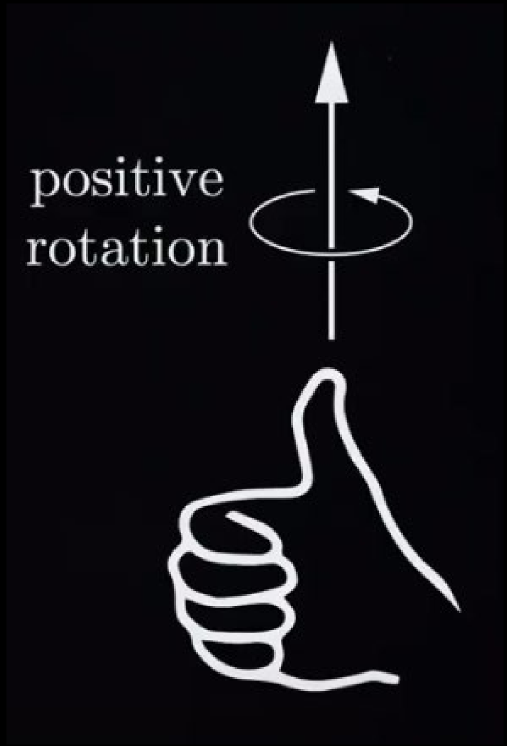


Translations are easy… Rotations require more careful consideration

Coordinate Frames / Conventions

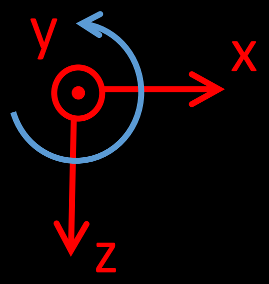
- Reference frames (origin and {x, y, z}-coordinates)
  - Right hand frames and rotations



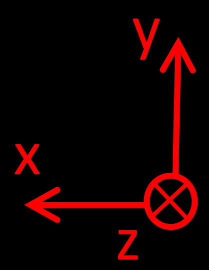


Are these right-handed or left-handed?

A)

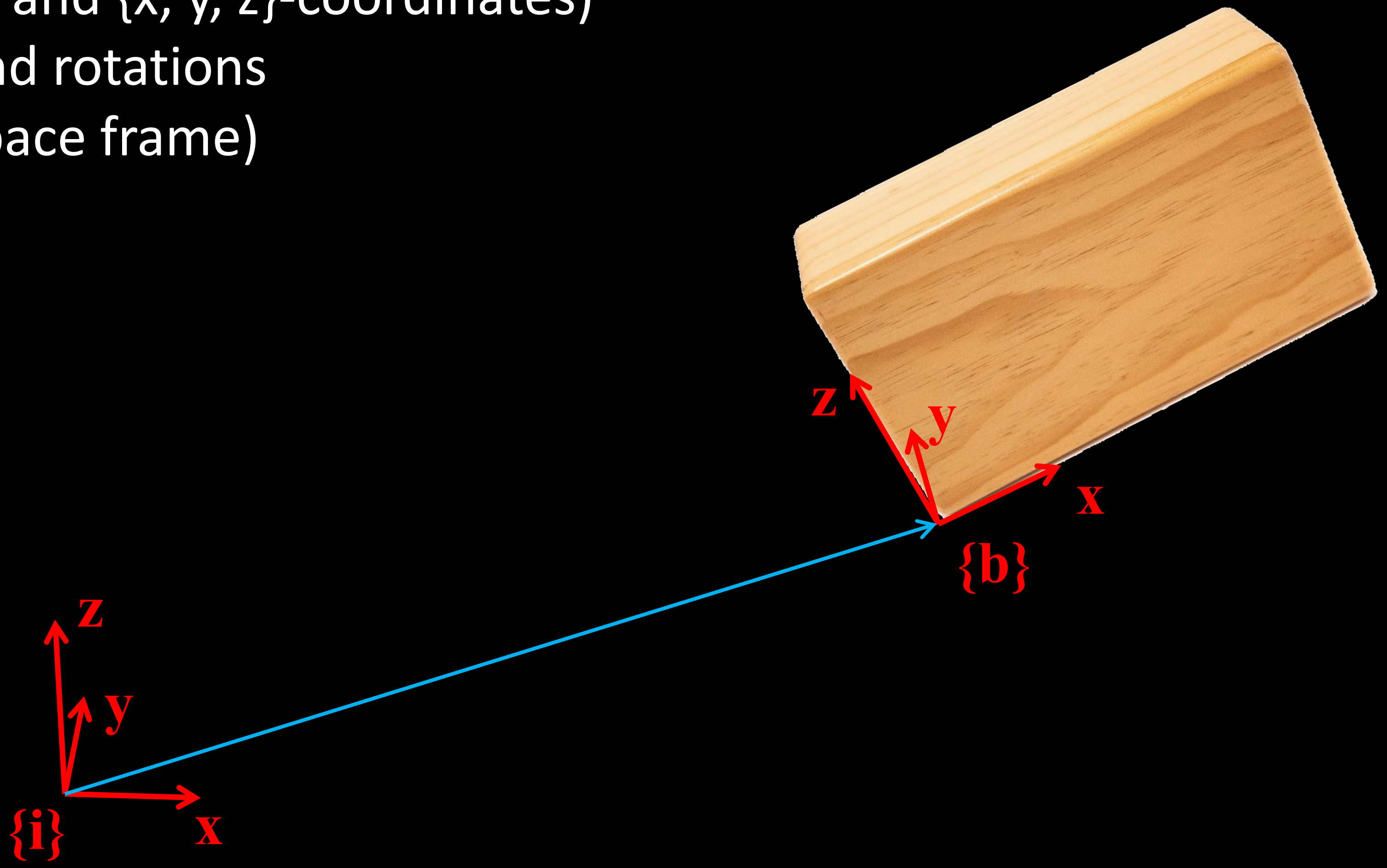


B)



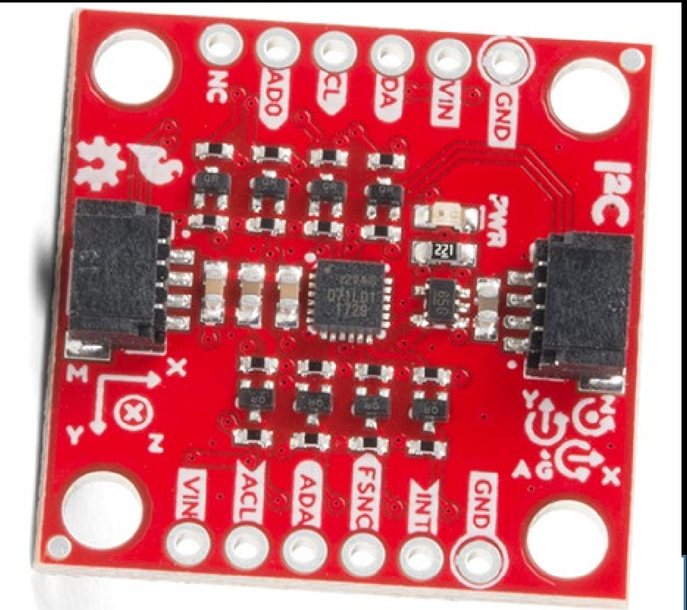
Coordinate Frames

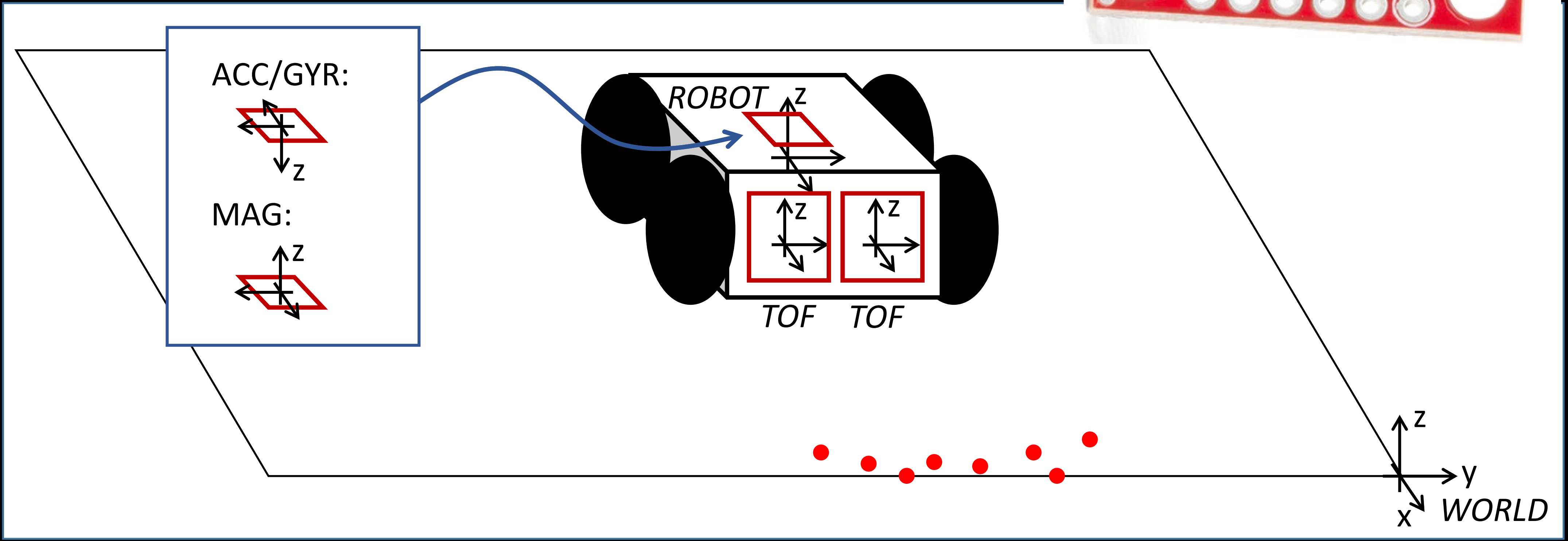
- Reference frames (origin and {x, y, z}-coordinates)
  - Right hand frames and rotations
- Inertial frame (/world/space frame)
- Body frame



Coordinate Frames

- Reference frames (origin and {x, y, z}-coordinates)
  - Right hand frames and rotations
- Inertial frame (/world/space frame)
- Body frame

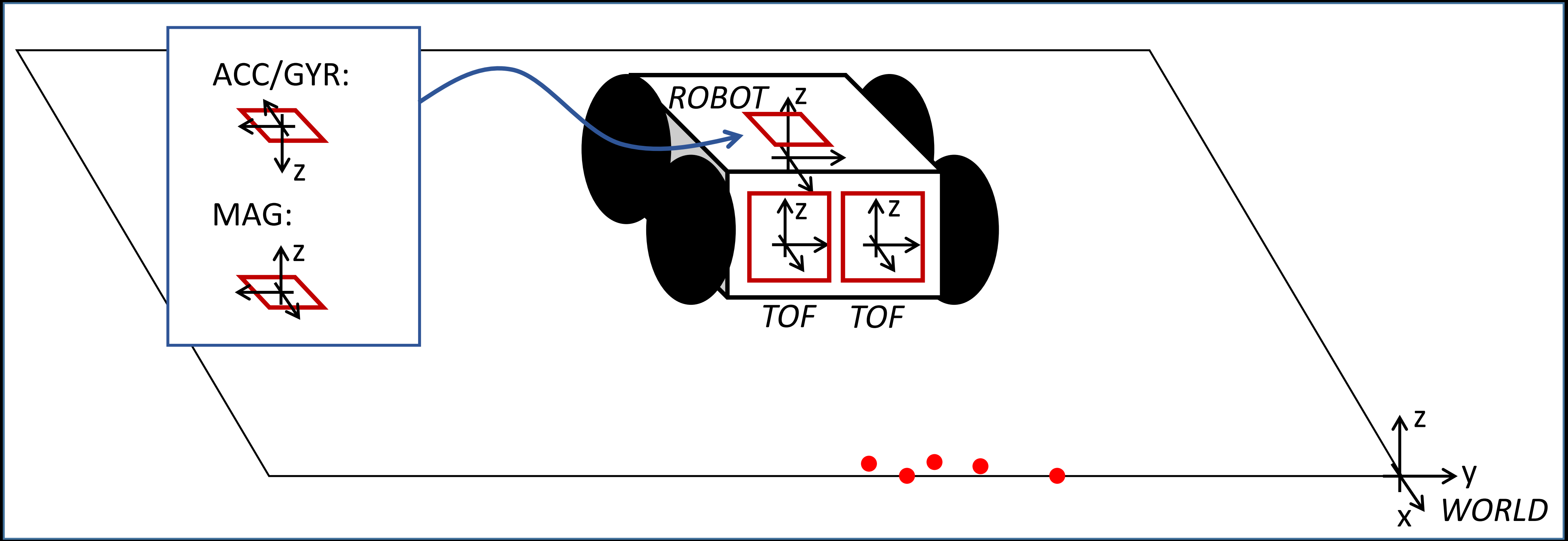




Homogeneous Transformation Matrix

rotation translation

$$T = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \boxed{\begin{matrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{matrix}} & \boxed{\begin{matrix} x \\ y \\ z \end{matrix}} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}$$



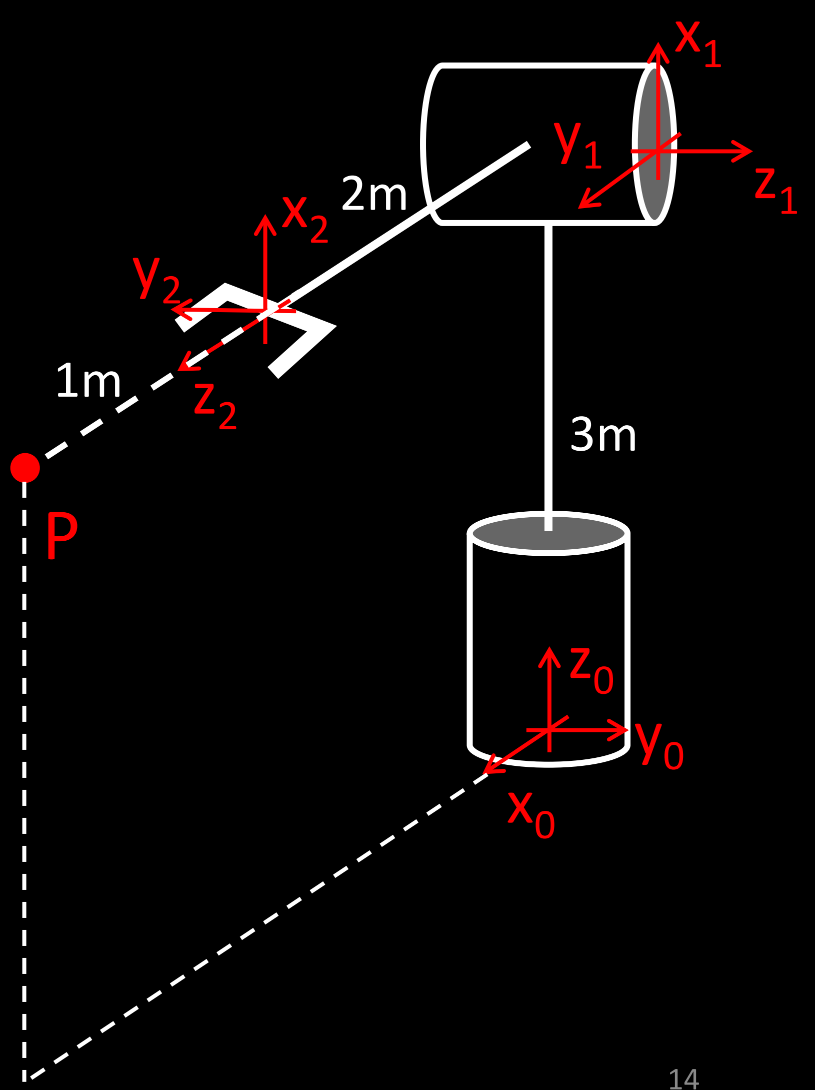
Homogeneous Transformation Matrix

- What is the location of the point P in reference frame 2?

$$P^2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

- What is the location of point P in reference frame 0?

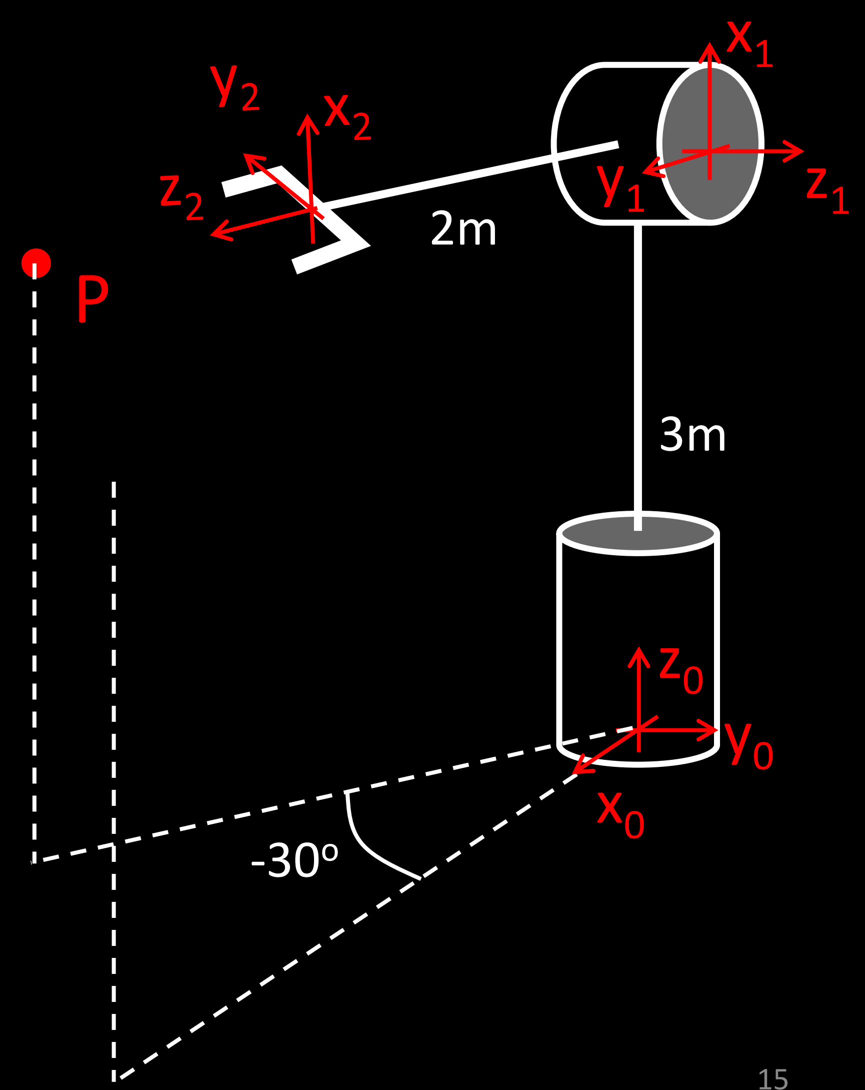
$$P^0 = \begin{bmatrix} 3 \\ 0 \\ 3 \end{bmatrix}$$



Homogeneous Transformation Matrix

- What is the location of the point P in reference frame 2?

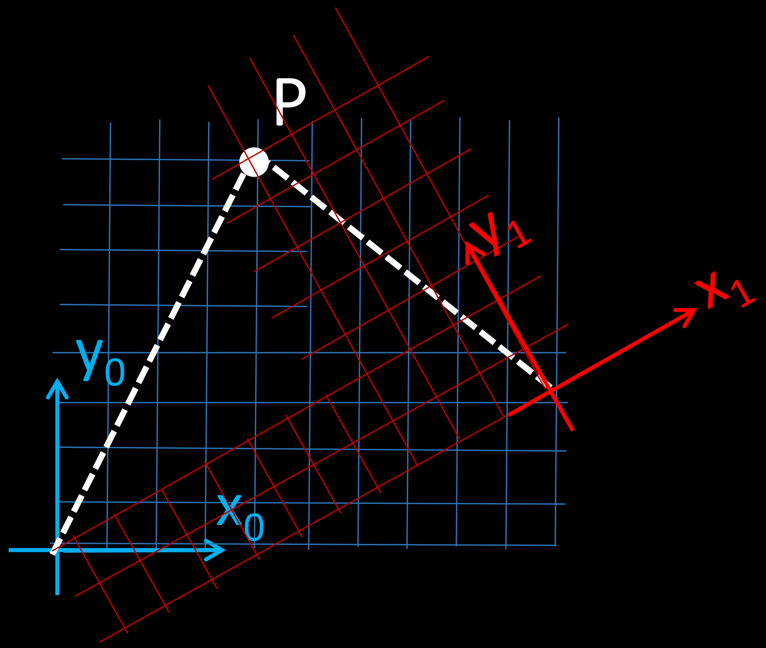
- What is the location of point P in reference frame 0?



Homogeneous Transformation Matrix

- The change in position and orientation between frames is described using transformation matrices

$P^0 = \begin{bmatrix} 4 \\ 8 \end{bmatrix}$ $P^1 = \begin{bmatrix} -3 \\ 7 \end{bmatrix}$ $O_1^0 = \begin{bmatrix} 10 \\ 3.3 \end{bmatrix}$ $O_0^1 = \begin{bmatrix} -10.3 \\ 2 \end{bmatrix}$



How do we express P^0 if we know P^1 and the relative location of O_1^0?

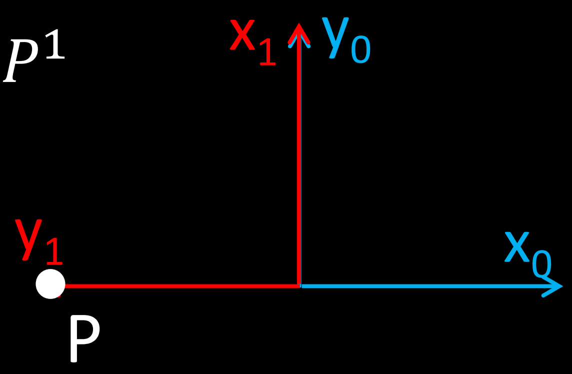
$$P^0 \neq P^1 + O_1^0$$

Homogeneous Transformation Matrix

- We need both translation and rotation!

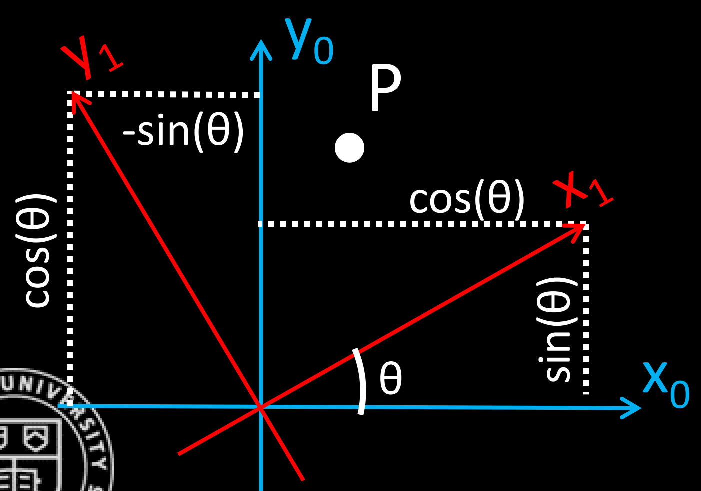
$$R_1^0 = [x_1^0 \quad y_1^0] \qquad x_1^0 = \begin{bmatrix} cos(\theta) \\ sin(\theta) \end{bmatrix} \qquad y_1^0 = \begin{bmatrix} -sin(\theta) \\ cos(\theta) \end{bmatrix} \qquad R_1^0 = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

$$P^0 = R_1^0 P^1 = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} P^1$$



e.g. if $P^1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $\theta = 90^o$:

$$P^0 = R_1^0 P^1 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$



Homogeneous Transformation Matrix

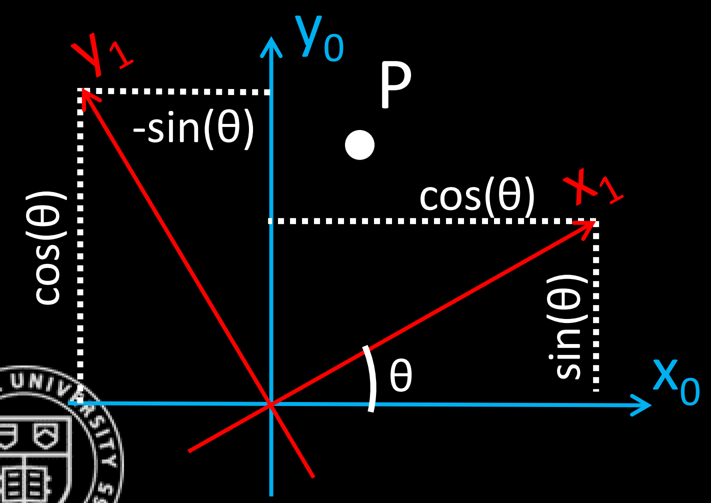
- We need both translation and rotation!

$$R_1^0 = [x_1^0 \quad y_1^0] \qquad x_1^0 = \begin{bmatrix} cos(\theta) \\ sin(\theta) \end{bmatrix} \qquad y_1^0 = \begin{bmatrix} -sin(\theta) \\ cos(\theta) \end{bmatrix}$$

$$R_1^0 = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

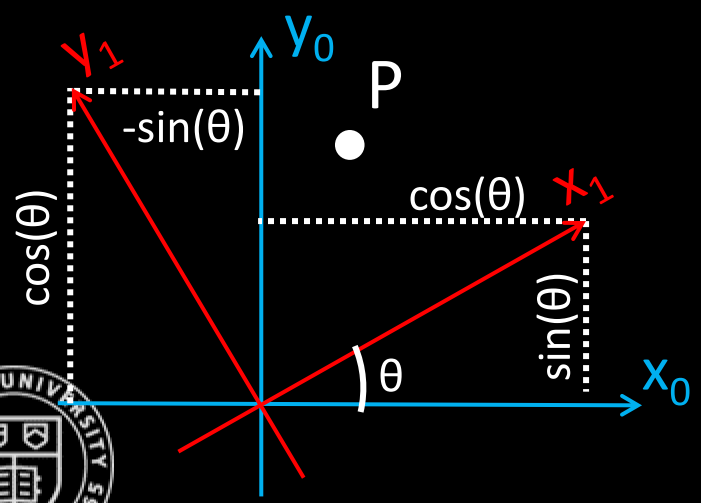
$$\Downarrow$$

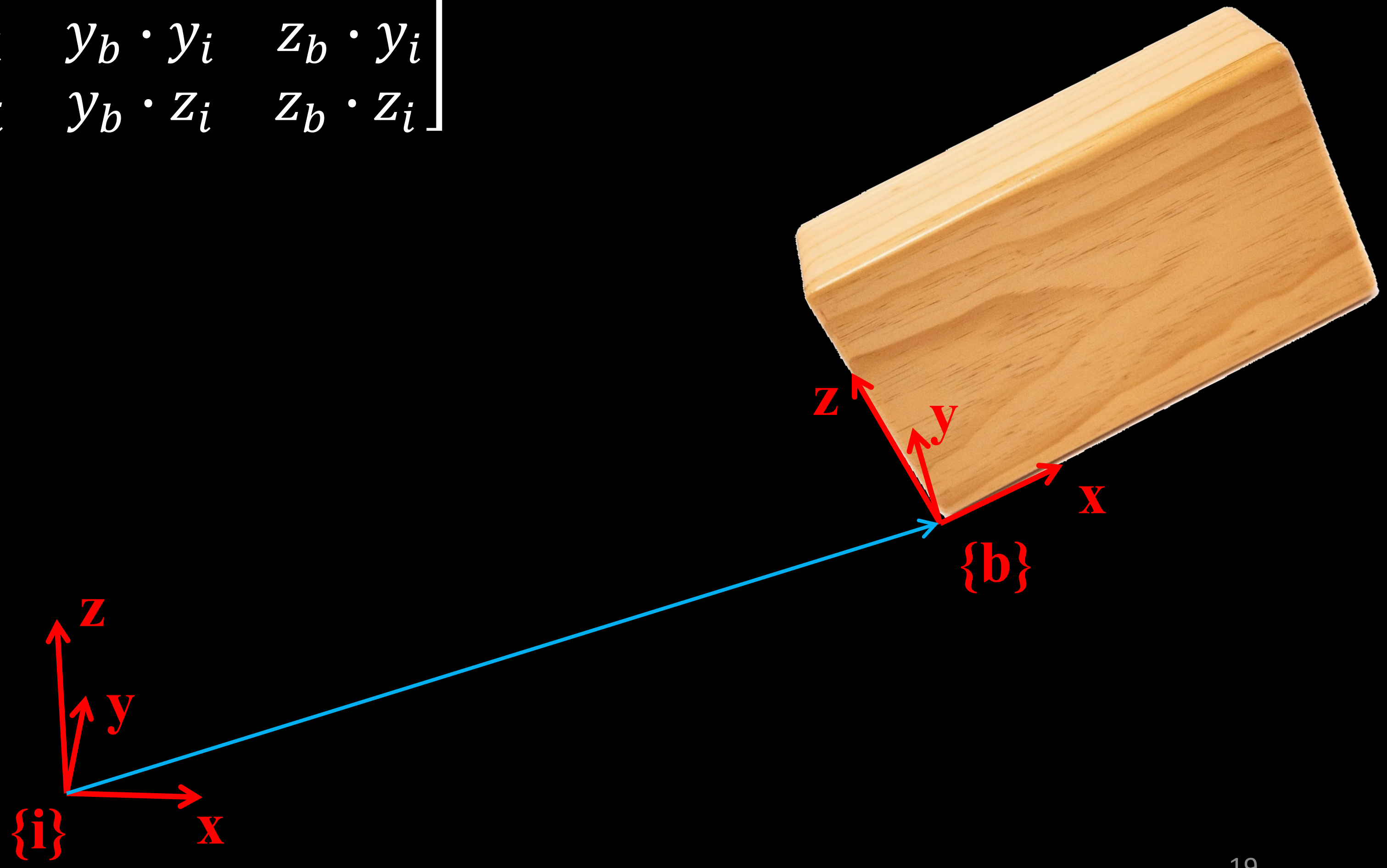
$$R_1^0 = \begin{bmatrix} x_0 \cdot x_1 & y_0 \cdot x_1 \\ x_0 \cdot y_1 & y_0 \cdot y_1 \end{bmatrix}$$



Rotation Matrix in 3D

$$R_b^i = \begin{bmatrix} x_b^i & y_b^i & z_b^i \end{bmatrix} = \begin{bmatrix} x_b \cdot x_i & y_b \cdot x_i & z_b \cdot x_i \\ x_b \cdot y_i & y_b \cdot y_i & z_b \cdot y_i \\ x_b \cdot z_i & y_b \cdot z_i & z_b \cdot z_i \end{bmatrix}$$



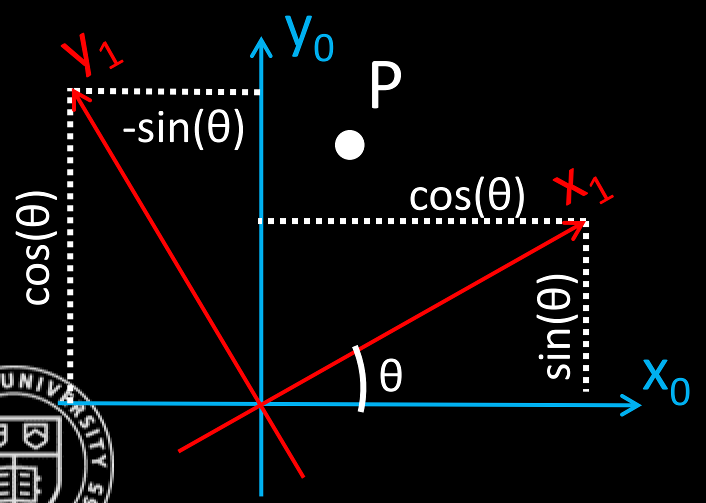


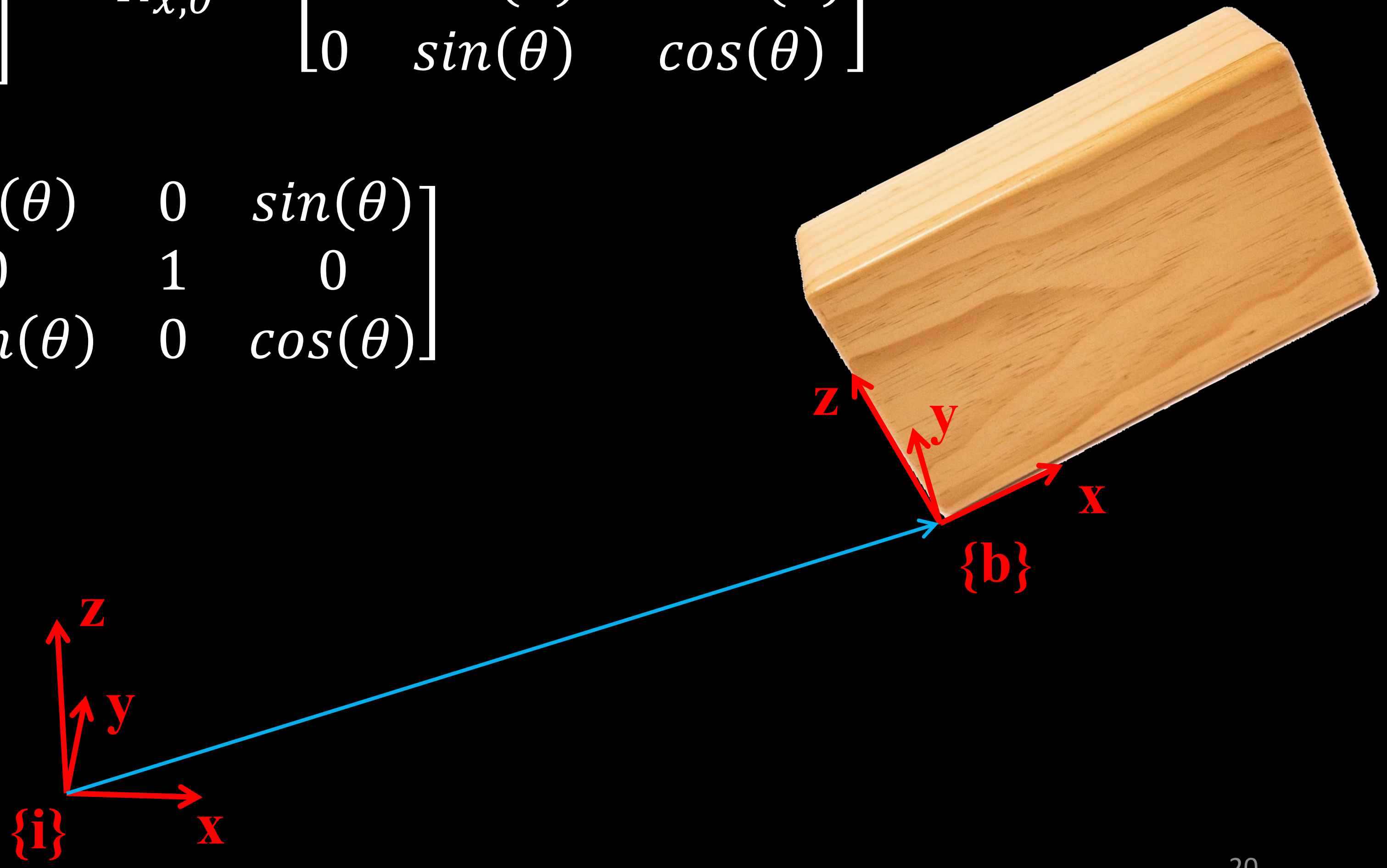
Rotation Matrix in 3D

- Find the rotation matrix $R_{z,\theta}$ for a rotation θ about z

$$R_{z,\theta} = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix}$$

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



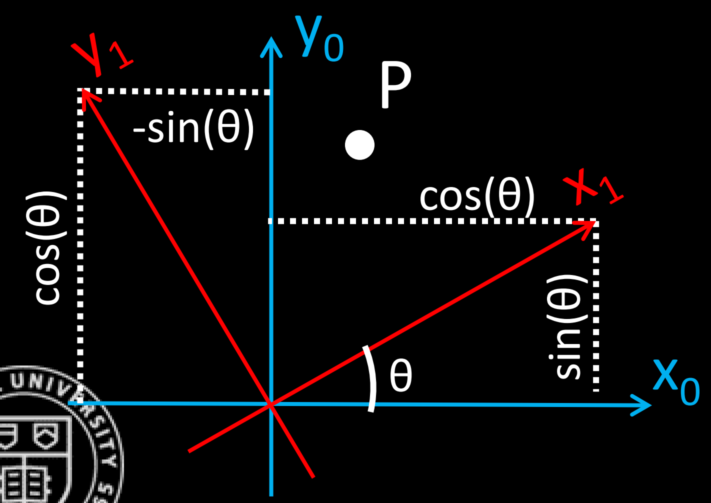


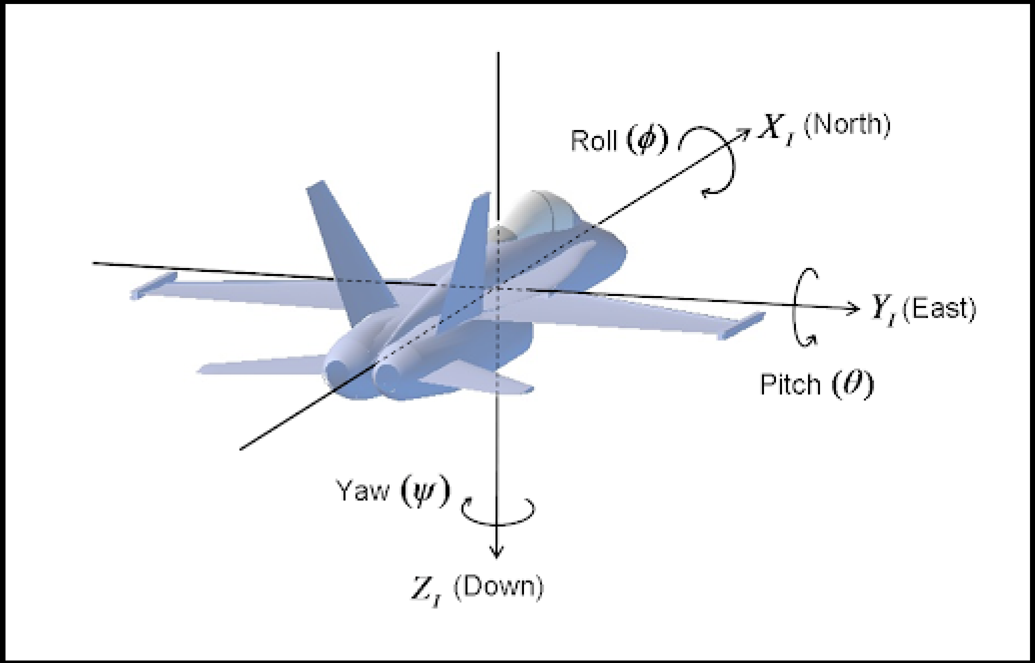
Rotation Matrix in 3D

- Find the rotation matrix $R_{z,\psi}$ for a rotation ψ about Z

$$R_{z,\psi} = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}$$

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



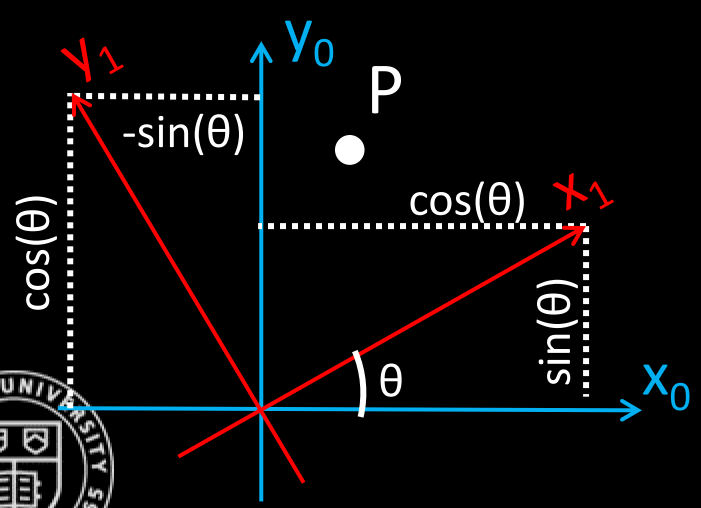


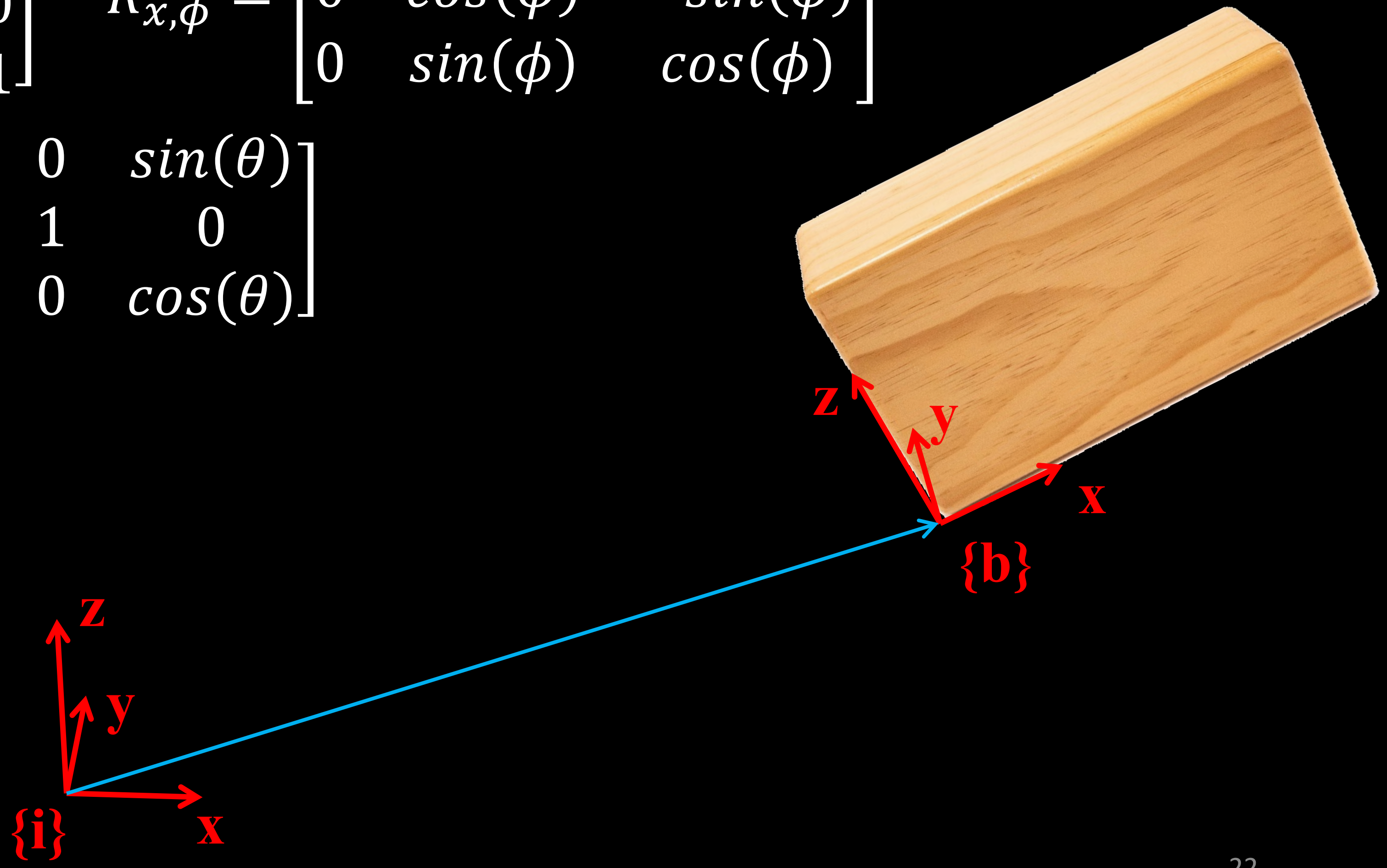
Rotation Matrix in 3D

- Find the rotation matrix $R_{z,\psi}$ for a rotation ψ about Z

$$R_{z,\psi} = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}$$

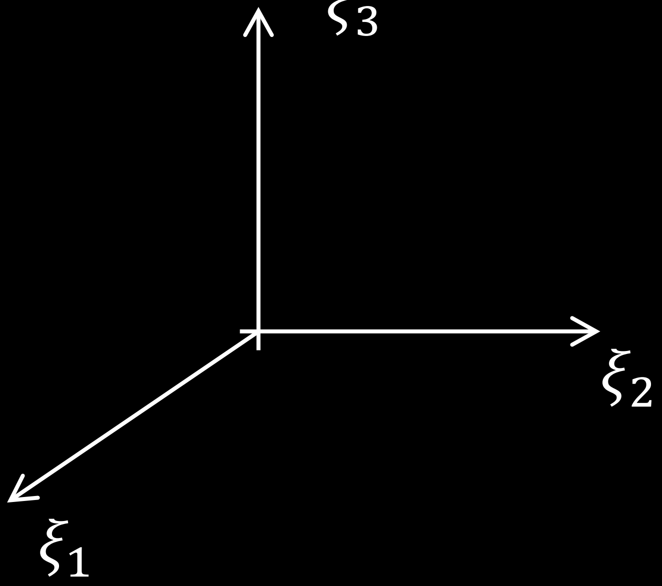
$$R_{y,\theta} = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix}$$





Euler

- *"Any rotation can be described by three successive rotations about linearly independent axis."*
  - **Proper Euler angles**
    - *z-x-z, x-y-x, y-z-y, z-y-z, x-z-x, y-x-y*
  - **Tait–Bryan angles**
    - *x-y-z, y-z-x, z-x-y, x-z-y, z-y-x, y-x-z*
  - Most commonly z-y-z or *x-y-z*

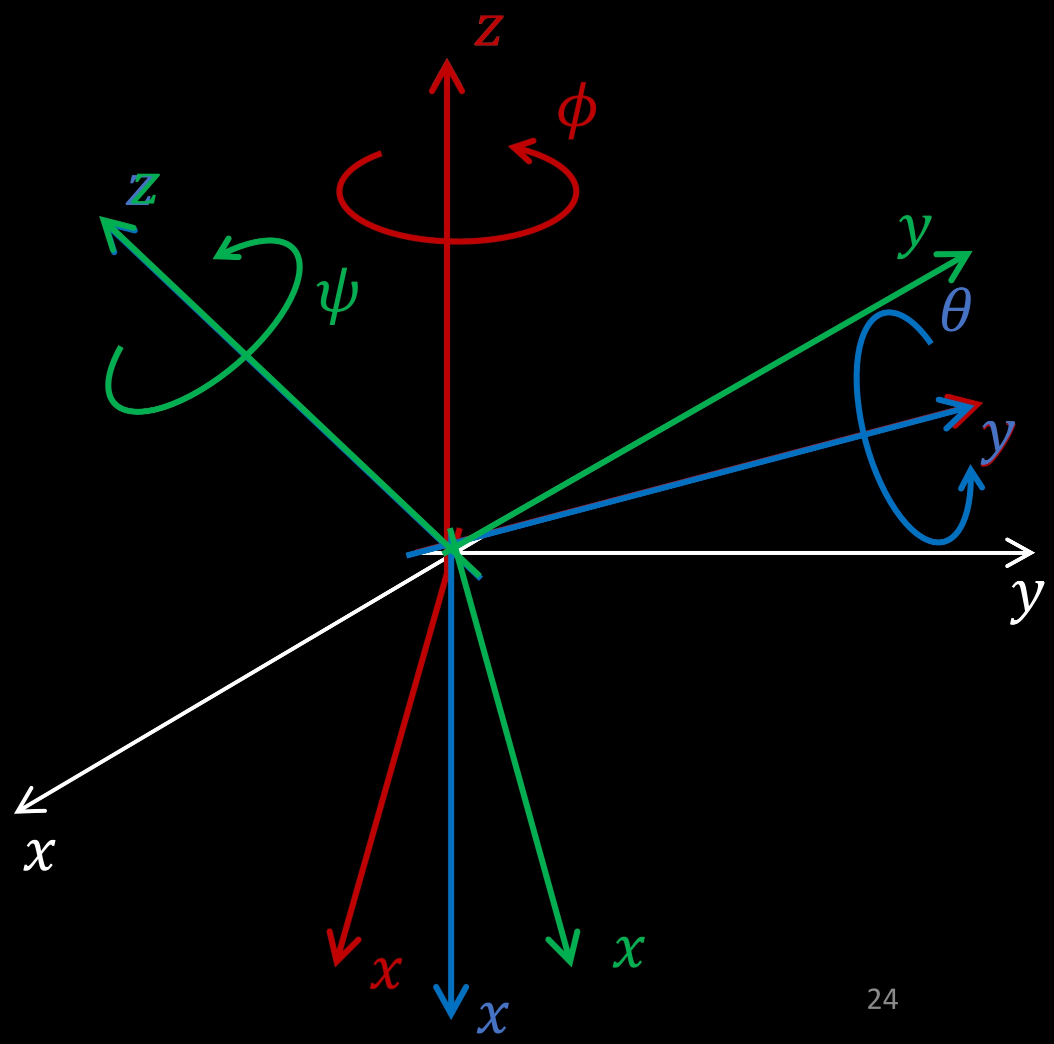


Rotation Matrix using ZYZ

$$R_{ZYZ} = R_{z,\phi} R_{y,\theta} R_{z,\psi}$$

$$= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\phi s_\psi & s_\phi s_\psi & c_\theta \end{bmatrix}$$



Rotation Matrix using Roll-Pitch-Yaw (X-Y-Z)

$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}$$

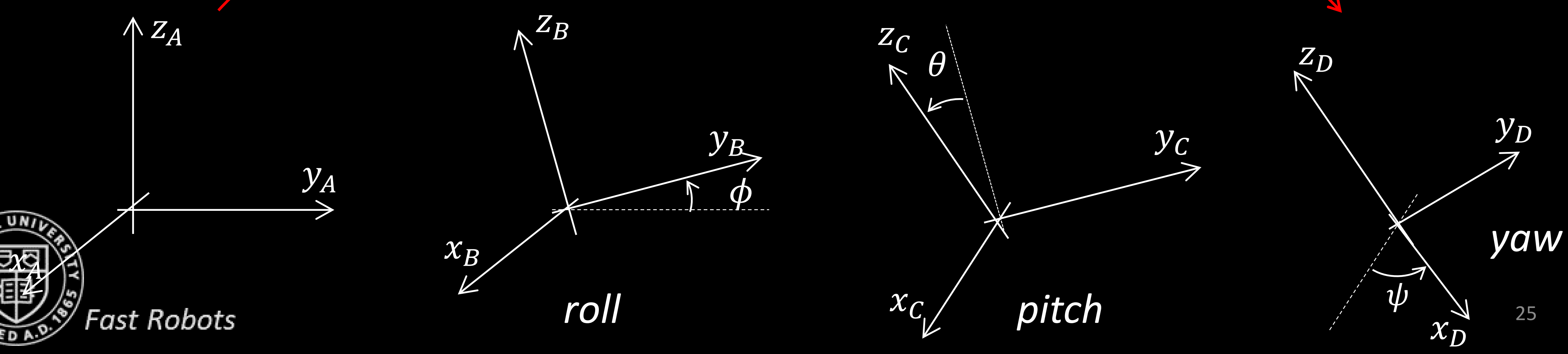
$$R_{XYZ} = R_{x,\phi} R_{y,\theta} R_{z,\psi}$$

$$= \begin{bmatrix} c_\theta c_\psi & -c_\theta s_\psi & s_\theta \\ c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi - s_\phi s_\theta s_\psi & -s_\phi c_\theta \\ s_\phi s_\psi - c_\phi s_\theta c_\psi & s_\phi c_\psi + c_\phi s_\theta s_\psi & c_\phi c_\theta \end{bmatrix}$$

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

$$R_{z,\psi} = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_D^A = R_B^A R_C^B R_D^C$$



Rotation Matrix using Roll-Pitch-Yaw (X-Y-Z)

$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}$$

$$R_{XYZ} = R_{x,\phi} R_{y,\theta} R_{z,\psi}$$

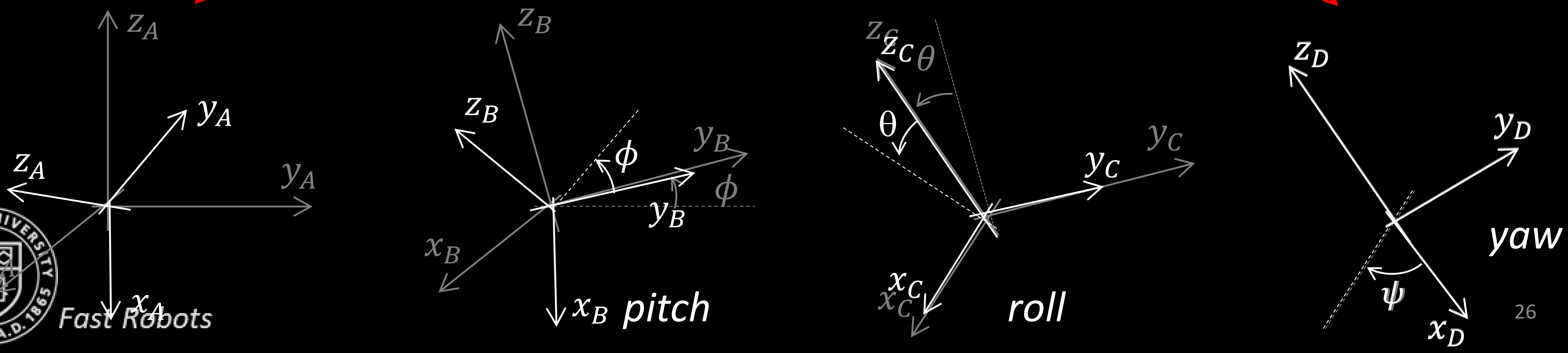
$$= \begin{bmatrix} c_\theta c_\psi & -c_\theta s_\psi & s_\theta \\ c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi - s_\phi s_\theta s_\psi & -s_\phi c_\theta \\ s_\phi s_\psi - c_\phi s_\theta c_\psi & s_\phi c_\psi + c_\phi s_\theta s_\psi & c_\phi c_\theta \end{bmatrix}$$

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

$$R_{z,\psi} = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Does the order matter? YES!

$R_D^A = \cancel{R_D^C R_C^B R_B^A}$?



Inverse Kinematics

– How to back out angles?

$$R_{XYZ} = R_{x,\phi} R_{y,\theta} R_{z,\psi}$$

$$= \begin{bmatrix} c_\theta c_\psi & -c_\theta s_\psi & s_\theta \\ c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi - s_\phi s_\theta s_\psi & -s_\phi c_\theta \\ s_\phi s_\psi - c_\phi s_\theta c_\psi & s_\phi c_\psi + c_\phi s_\theta s_\psi & c_\phi c_\theta \end{bmatrix}$$

- But the solution to acos is not unique
- atan(x) returns [- $\pi/2, \pi/2$]
- Instead use atan2(adj,opp)* which returns [- π, π]
  - θ = asin(r_{13})
  - ϕ = atan2(-r_{23}, r_{33})
  - ψ = atan2(-r_{12}, r_{11})

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

- Special case if r_{13}=1 (the z' axis is parallel to the x-axis)
  - $\theta = 90^o$, ψ = atan2(r_{21}, r_{22}), $\phi = 0^o$

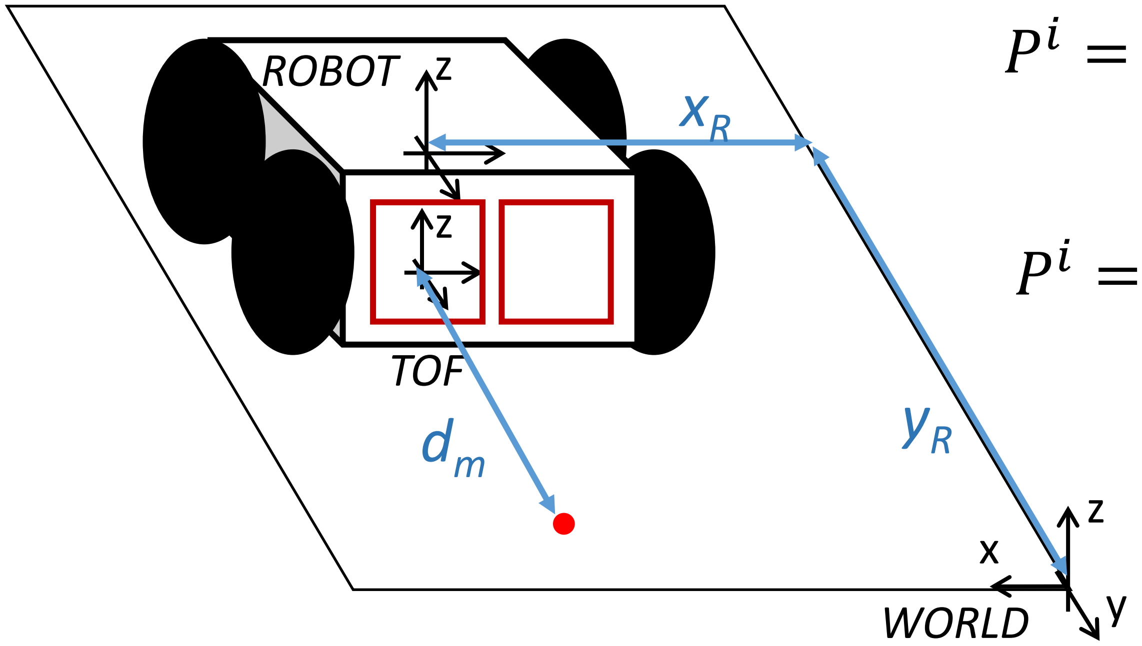
```
float atan2(float x, float y) {
    if (x > 0.0)
        return atan(y/x);
    if (x < 0.0) {
        if (y >= 0.0)
            return (PI + atan(y/x));
        else
            return (-PI + atan(y/x));
    }
    if (y > 0.0) // x == 0
        return PI_ON_TWO;
    if (y < 0.0)
        return -PI_ON_TWO;
    return 0.0; // Should be undefined
}
```

*These are not consistent across platforms!

Homogeneous Transformation Matrix

rotation translation

$$T = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_x \\ r_{21} & r_{22} & r_{23} & d_y \\ r_{31} & r_{32} & r_{33} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_\theta c_\psi & -c_\theta s_\psi & s_\theta & d_x \\ c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi - s_\phi s_\theta s_\psi & -s_\phi c_\theta & d_y \\ s_\phi s_\psi - c_\phi s_\theta c_\psi & s_\phi c_\psi + c_\phi s_\theta s_\psi & c_\phi c_\theta & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$P^i = T_R^i T_{TOF}^R P^{TOF}$$

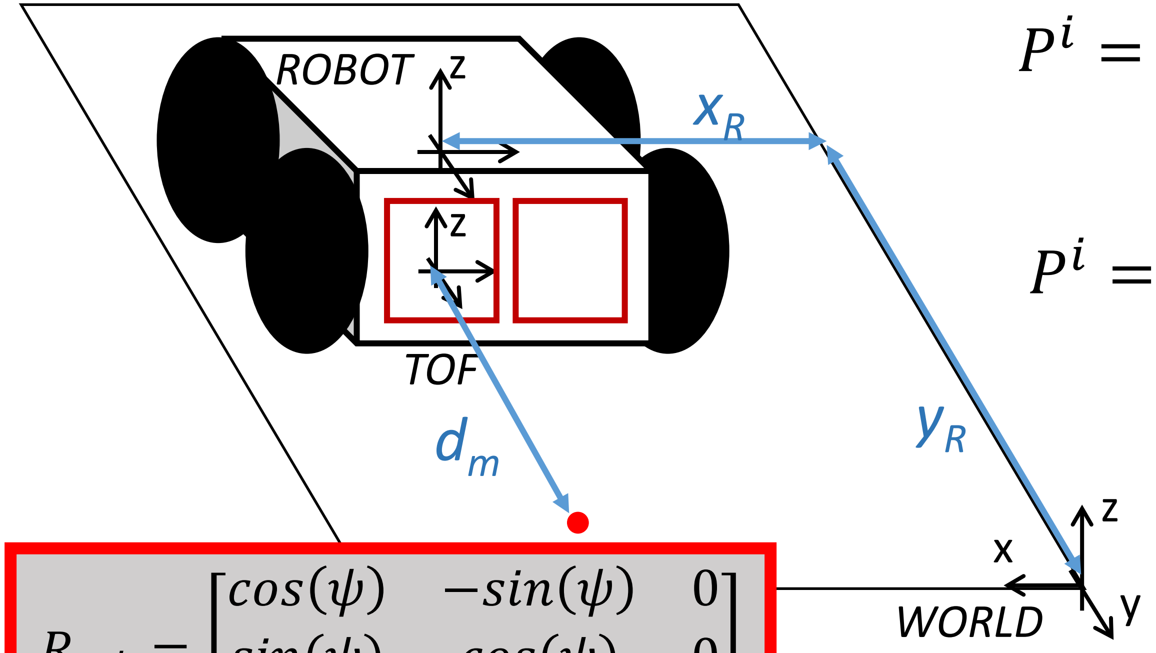
$$P^i = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} & & & \\ & & & \\ & & & \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_m \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Homogeneous Transformation Matrix

rotation

translation

$$T = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_x \\ r_{21} & r_{22} & r_{23} & d_y \\ r_{31} & r_{32} & r_{33} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_\theta c_\psi & -c_\theta s_\psi & s_\theta & d_x \\ c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi - s_\phi s_\theta s_\psi & -s_\phi c_\theta & d_y \\ s_\phi s_\psi - c_\phi s_\theta c_\psi & s_\phi c_\psi + c_\phi s_\theta s_\psi & c_\phi c_\theta & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$R_{z,\psi} = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P^i = T_R^i T_{TOF}^R P^{TOF}$$

$$P^i = \begin{bmatrix} \quad \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0.08 \\ 0 & 1 & 0 & -0.015 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_m \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

if $X_R = 1$, $y_R = 1$, $d_m = 1$:

$=[1.015 \quad 0.08 \quad 0 \quad 1]^T$

Sources and References

- Northwestern University, course on Modern Robotics
- Upenn Coursera course on Aerial Robotics
- MilfordRobotics youtube stream
- Mecademic

Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

Fast Robots
Lab 2

Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

Fast Robots
Data Types

Data types

- What data types will you have in your system?
  - Bluetooth: char
  - Time of flight: unsigned int
  - Serial.print: strings
  - IMU: float
  - PID: double
  - millis(): unsigned long
  - if-statements: bool

Data types

- Two's complement
    - 0 b 0 0 0 0 0 1 0 1 ?
        - = 5_{dec}
    - -5_{dec} ?
        - 0 b 0 0 0 0 0 1 0 1 > invert > 0 b 1 1 1 1 1 0 1 0 > add 1 > 0 b 1 1 1 1 1 0 1 1
    - 0 b 1 1 1 1 1 1 1 1 ?
        - = -1_{dec}

Data types

- Variable memory allocation depends on your processor *and* the compiler
  - Char
    - Char_{8bit} : 8 bits
    - Char_{32bit} : 8 bits
  - Int
    - Int_{8bit} : 16 bits
    - Int_{32bit} : 32 bits
  - Long
    - Long_{32bit} : 32bits
    - Long_{64bit} : 64 bits

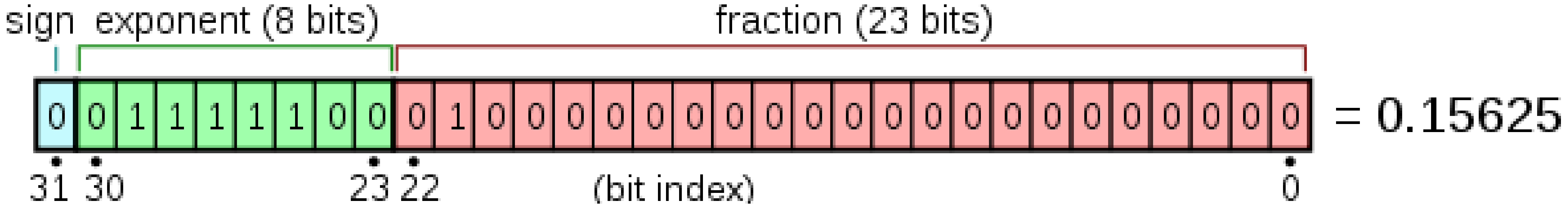
You can specify the length:
- int16_t
- uint32_t

- Bool
  - Bool_{8bit} : 8 bits
  - Bool_{32bit} : 32 bits

- Range
  - Signed char_{32bit} $= [-2^7; 2^7-1] = [-128; 127]$
  - Unsigned char_{32bit} $= [0; 2^8-1] = [0; 255]$
  - int_{32bit} $= [-2^{31} ; 2^{31}-1]$

Data types

- Variable memory allocation depends on your processor *and* the compiler
  - Float
    - Float_{8bit} : 32 bits
    - Float_{32bit} : 32 bits
    - Single-precision floating point number
      - Max value $\approx 3.4028235 \times 10^{38}$

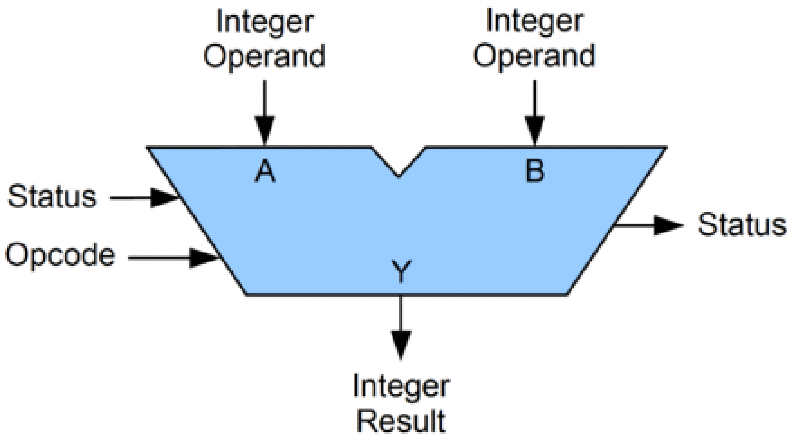


$$(-1)^{b_{31}} \times 2^{(b_{30}b_{29}\ldots b_{23})_2 - 127} \times (1.b_{22}b_{21}\ldots b_0)_2$$

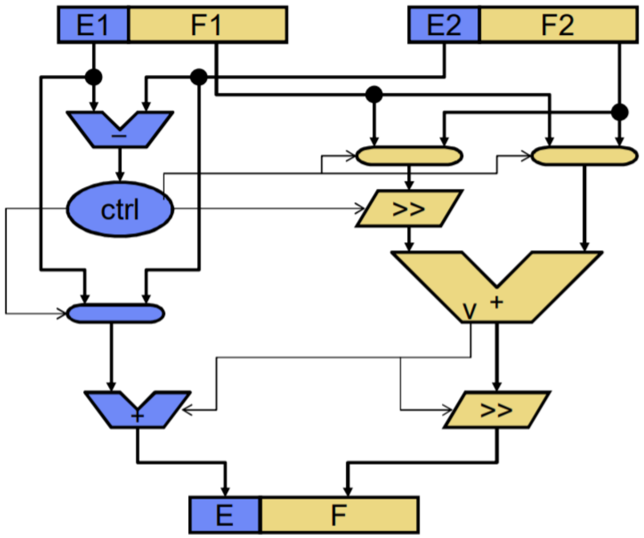
Data types

- Variable memory allocation depends on your processor *and* the compiler
  - Float
    - Float_{8bit} : 32 bits
    - Float_{32bit} : 32 bits
    - Single-precision floating point number
      - Max value $\approx 3.4028235 \times 10^{38}$

Integer ALU



Floating point ALU



Data types

- Variable memory allocation depends on your processor *and* the compiler
    - Float
        - Float_{8bit} : 32 bits
        - Float_{32bit} : 32 bits
        - Single-precision floating point number
            - Max value $\approx 3.4028235 \times 10^{38}$
    - Double
        - Double_{8bit} : 64 bits
        - Double_{32bit} : 64 bits
    - Long Double
        - 8, 12, 16 bytes

Data types

- What data types will you have in your system?
  - Bluetooth: char
  - Time of flight: unsigned int
  - Serial.print: strings
  - IMU: float
  - PID: double
  - millis(): unsigned long
  - if-statements: bool
- *Pay attention!*
- https://www3.ntu.edu.sg/home/ehchua/programming/java/datarepresentation.html

Action items

- *If you decide not to take the course, let Kirstin/Sharif know ASAP (40+ on the waitlist)*

- Jan 27th, midnight: Make a Github repository and build a Github page
  - Your name, a small introduction, the class number, and a photo
  - Share the page link over Canvas

- Labs start this week
  - Upload your write-up of Lab 1 by 8am the following week
    - (E.g. Tuesday lab write-ups are due the following Tuesday 8am)