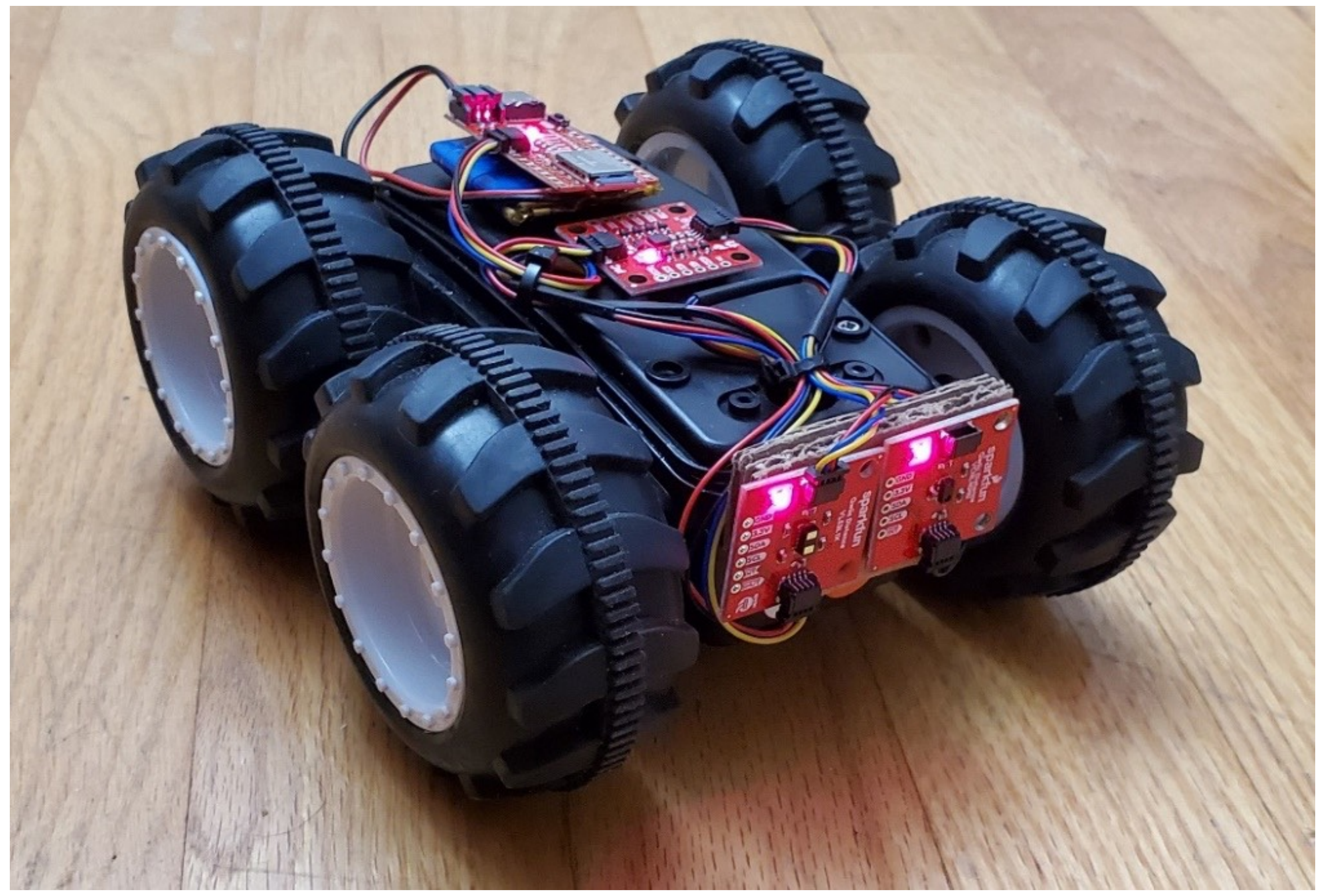
T-matrices

Fast Robots, ECE4160/5160, MAE 4190/5190

E. Farrell Helbling, 1/22/26
Slides adapted from Prof. Kirstin Petersen

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



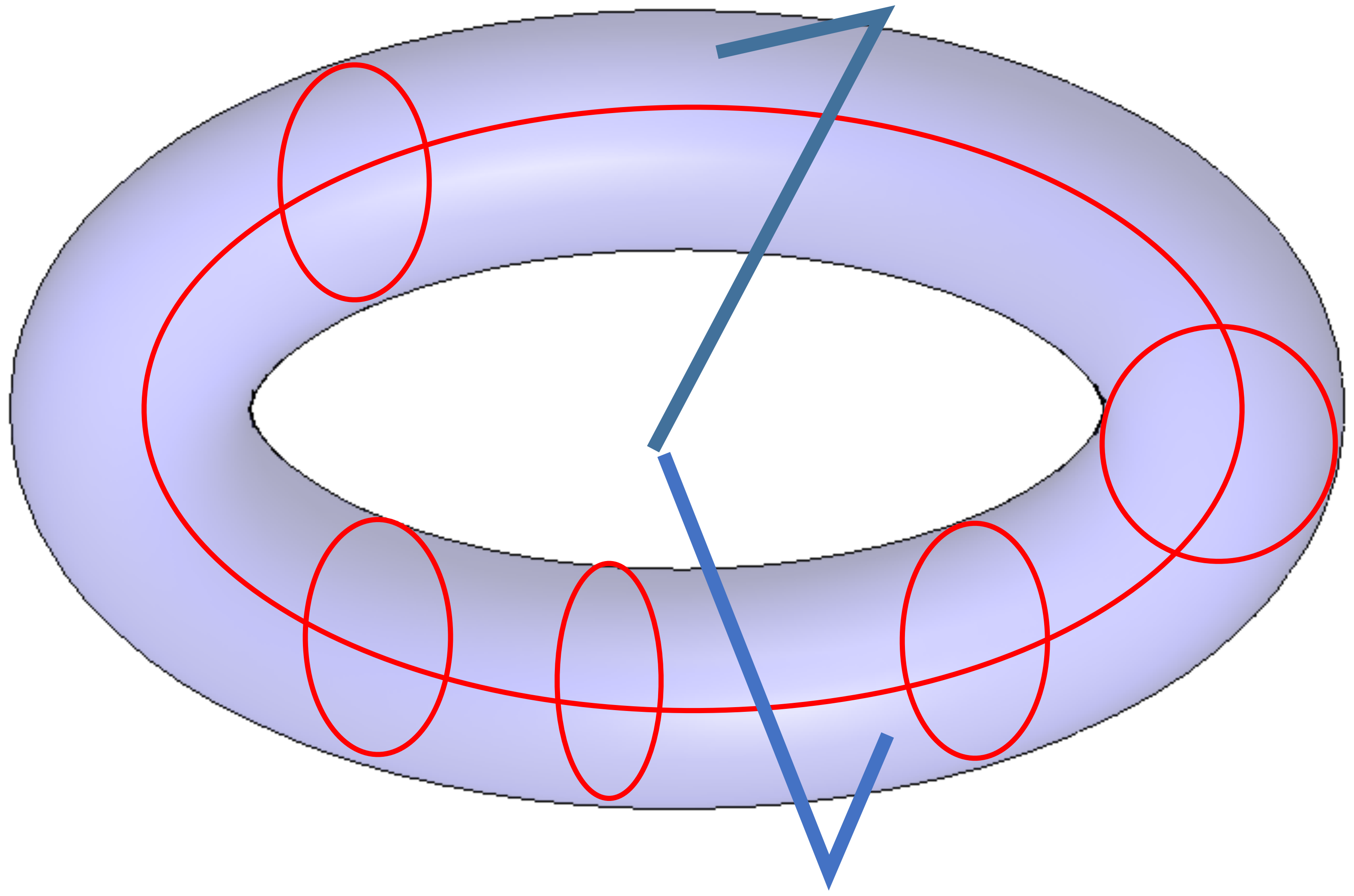
What is the DOF of these?

Configuration, C-space, DOFs

2 DOF robot arm

J-space: surface of a torus

C-space: 2 angles



Every robot configuration has a unique point on the torus

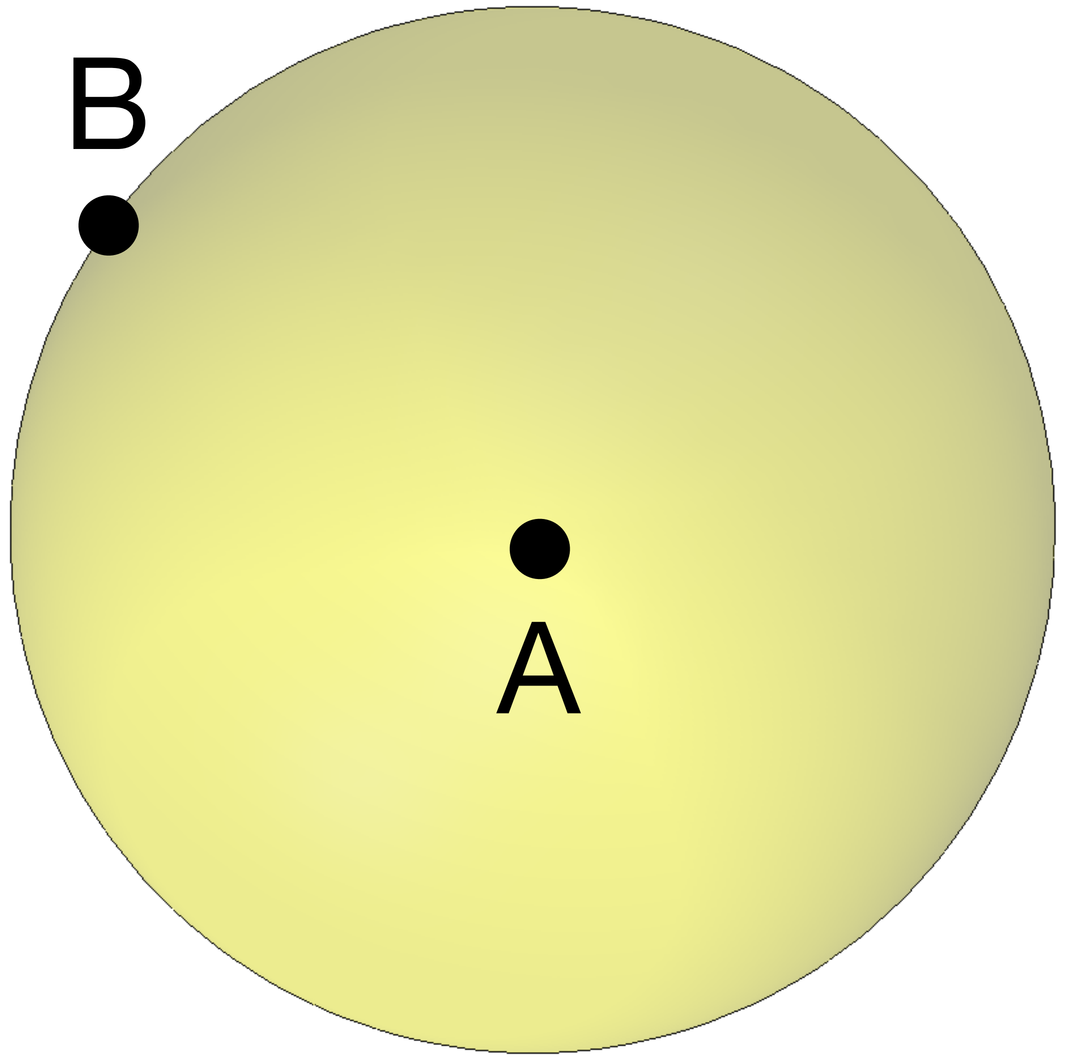
Every point on the torus is a unique robot configuration

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: $\{\theta, \phi\}$

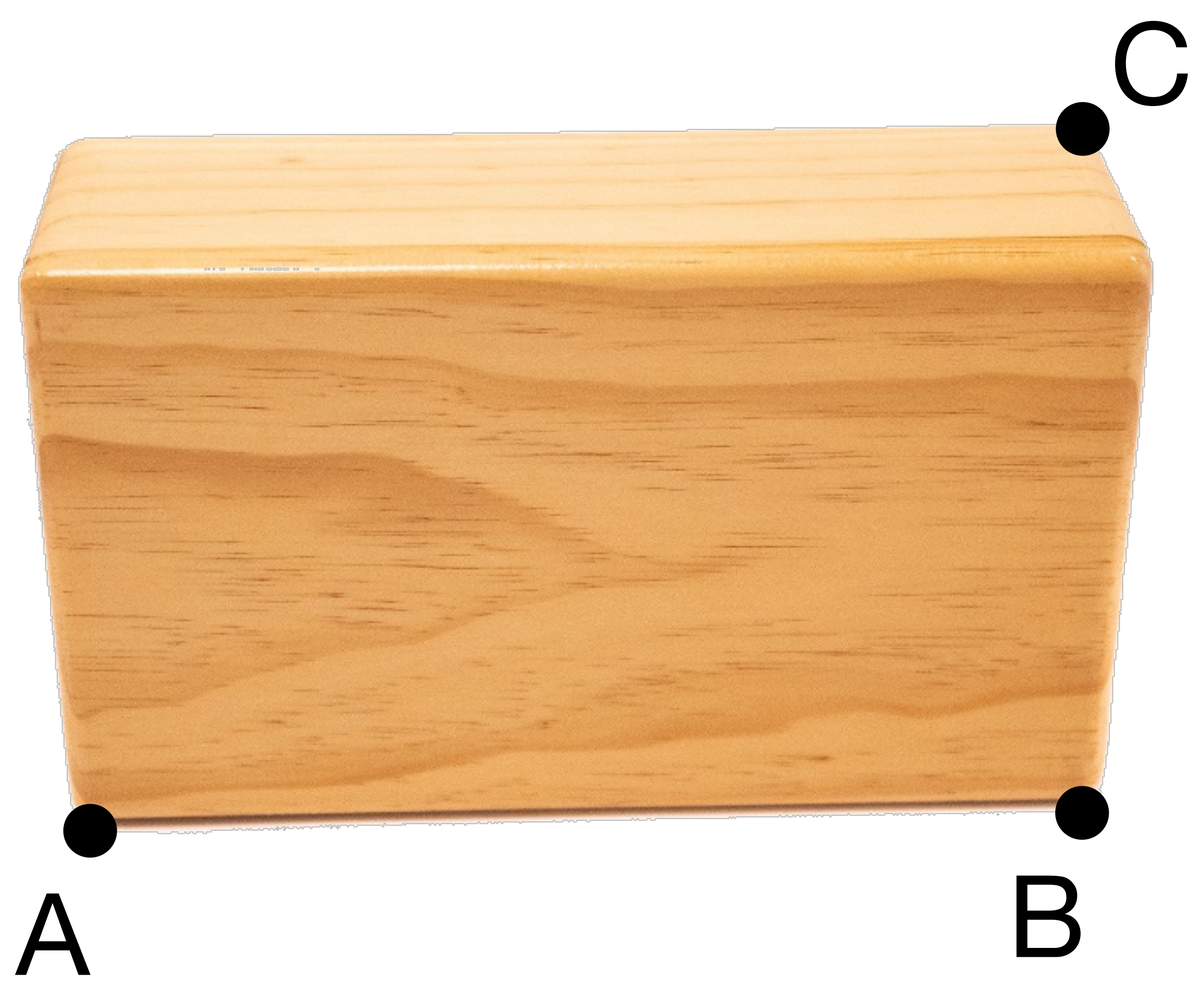
Point C: $\{\psi\}$

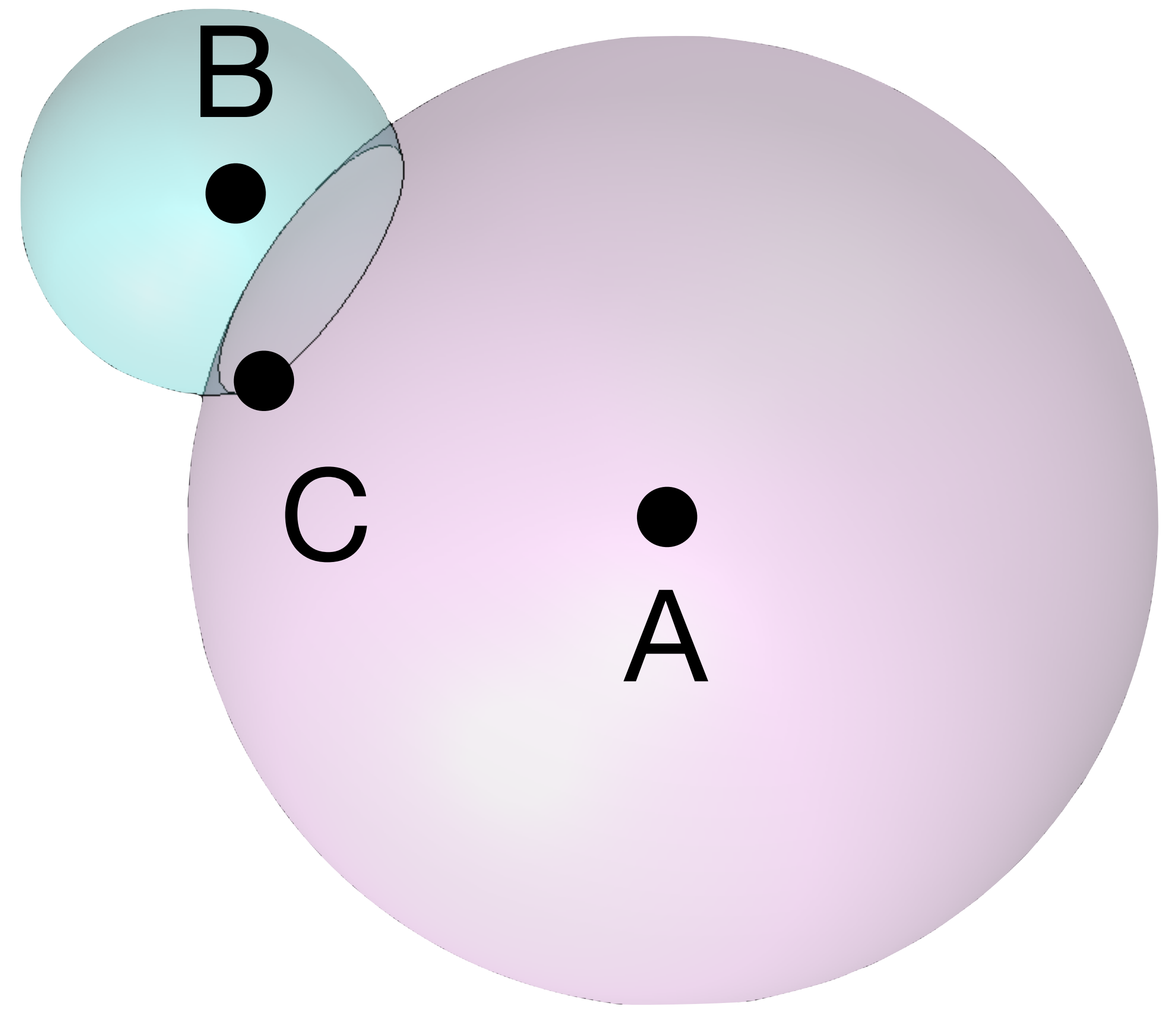
A rigid body in 3D has 6DOF

A rigid body in 2D has 3DOF

A rigid body in 4D has 10DOF

Point	Coords	Ind. Constraints	Real Freedoms
A	3	0	3
B	3	1	2
C	3	2	1
D	3	3	0
Total			6

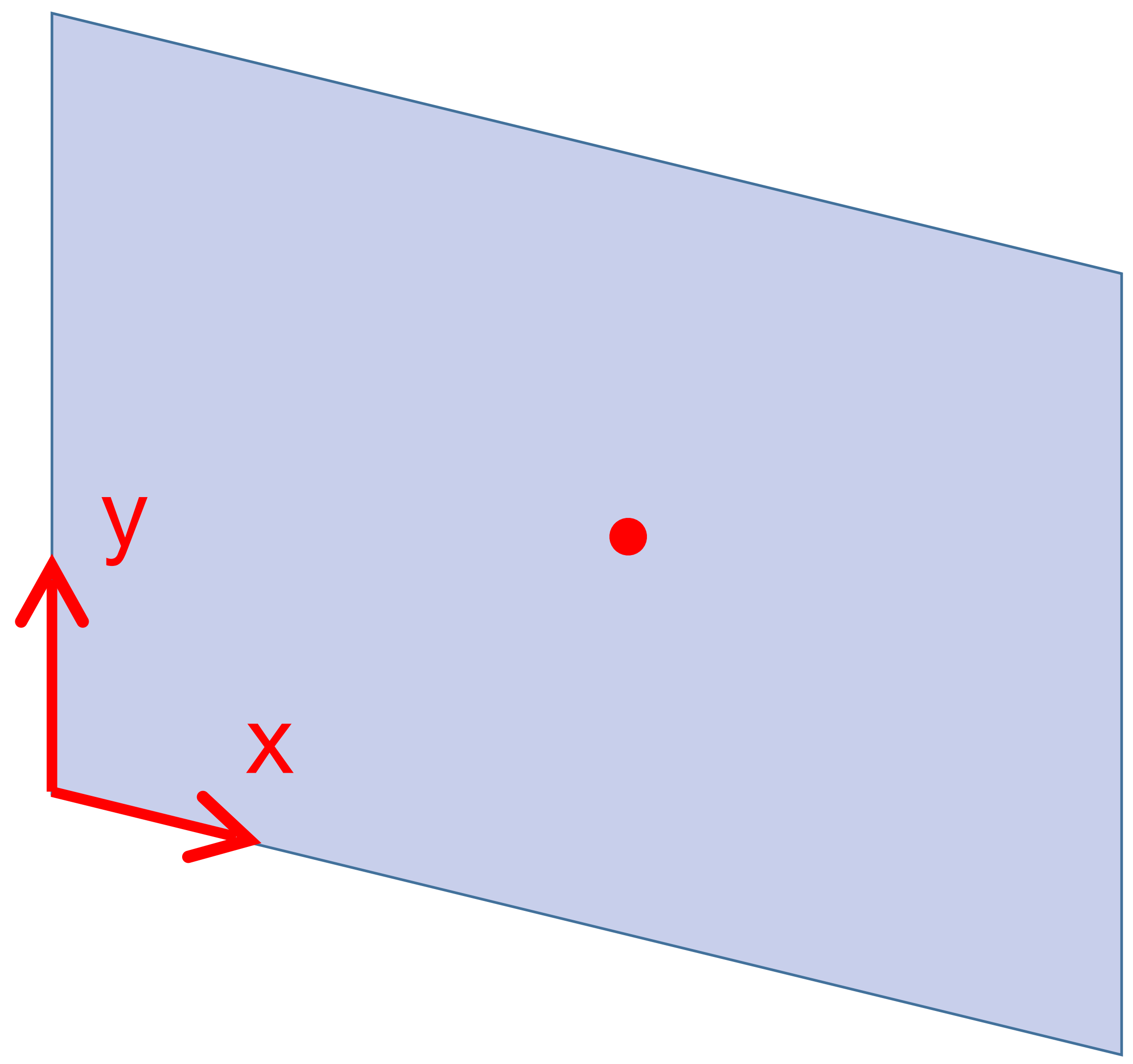




$$\boxed{\text{DOF} = \Sigma(\text{freedom of points - independent constraints})}$$

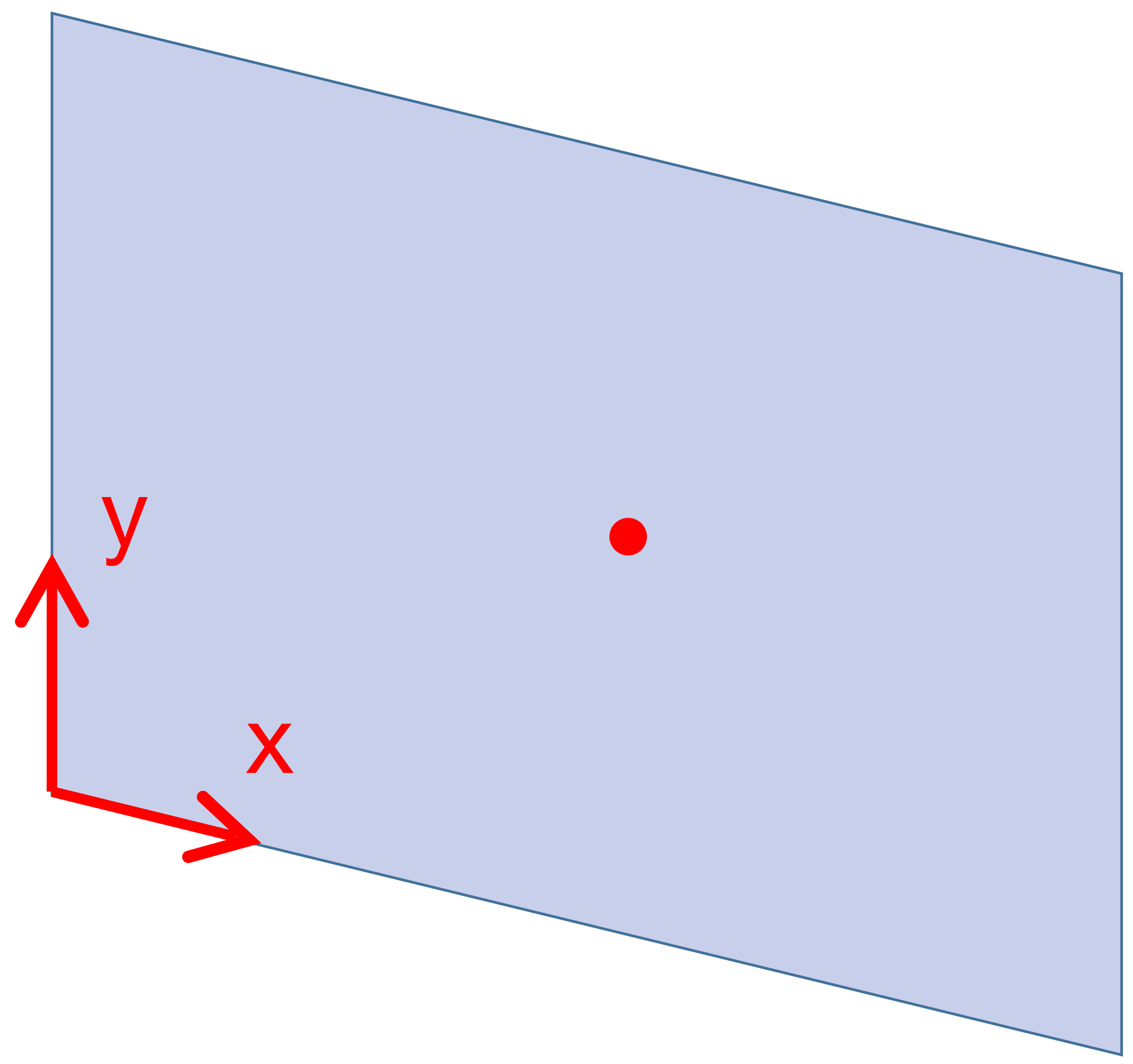
Topology Representation

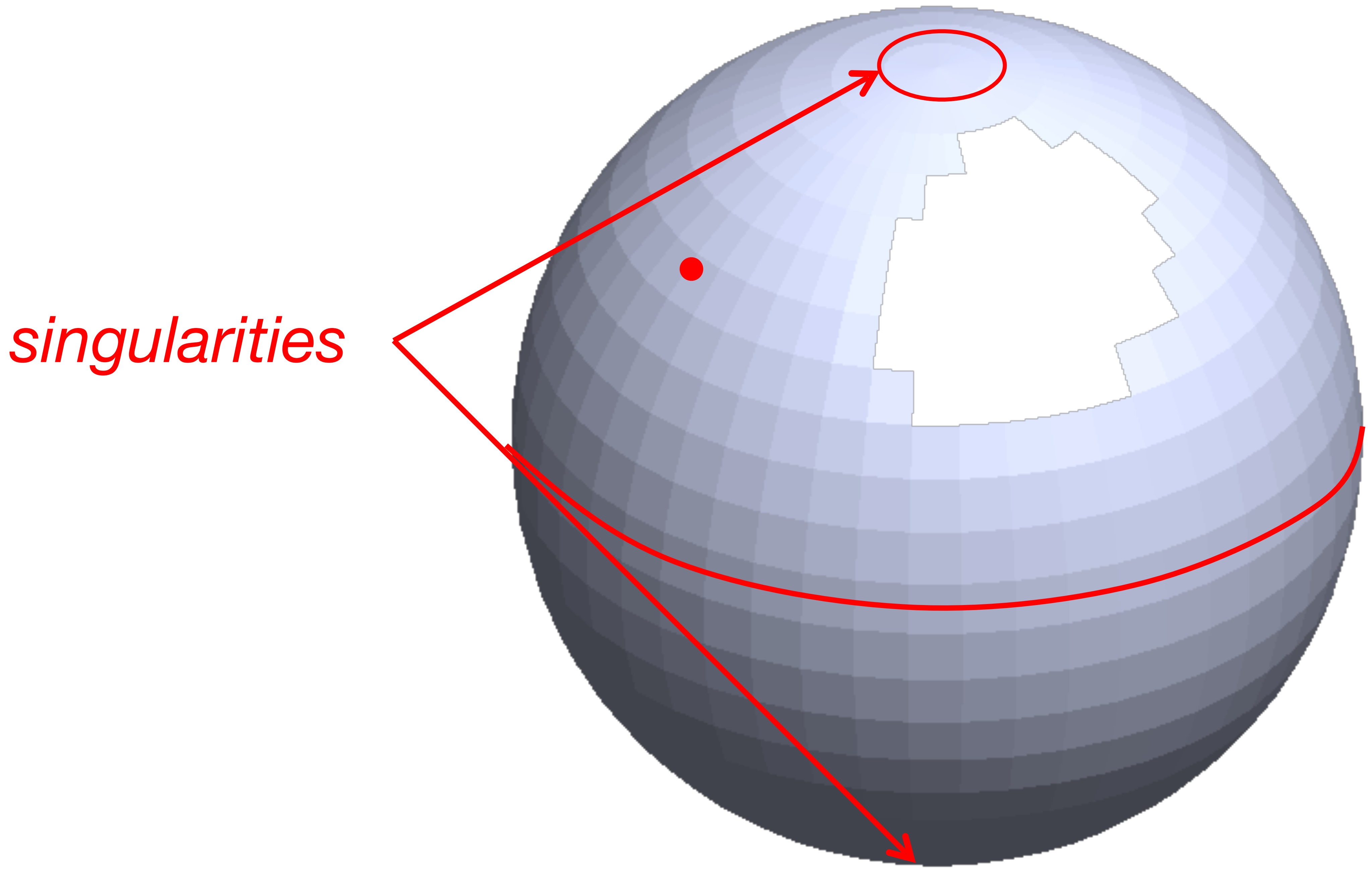
- Point on a plane: origin and 2 orthogonal coordinate axes.



Topology Representation

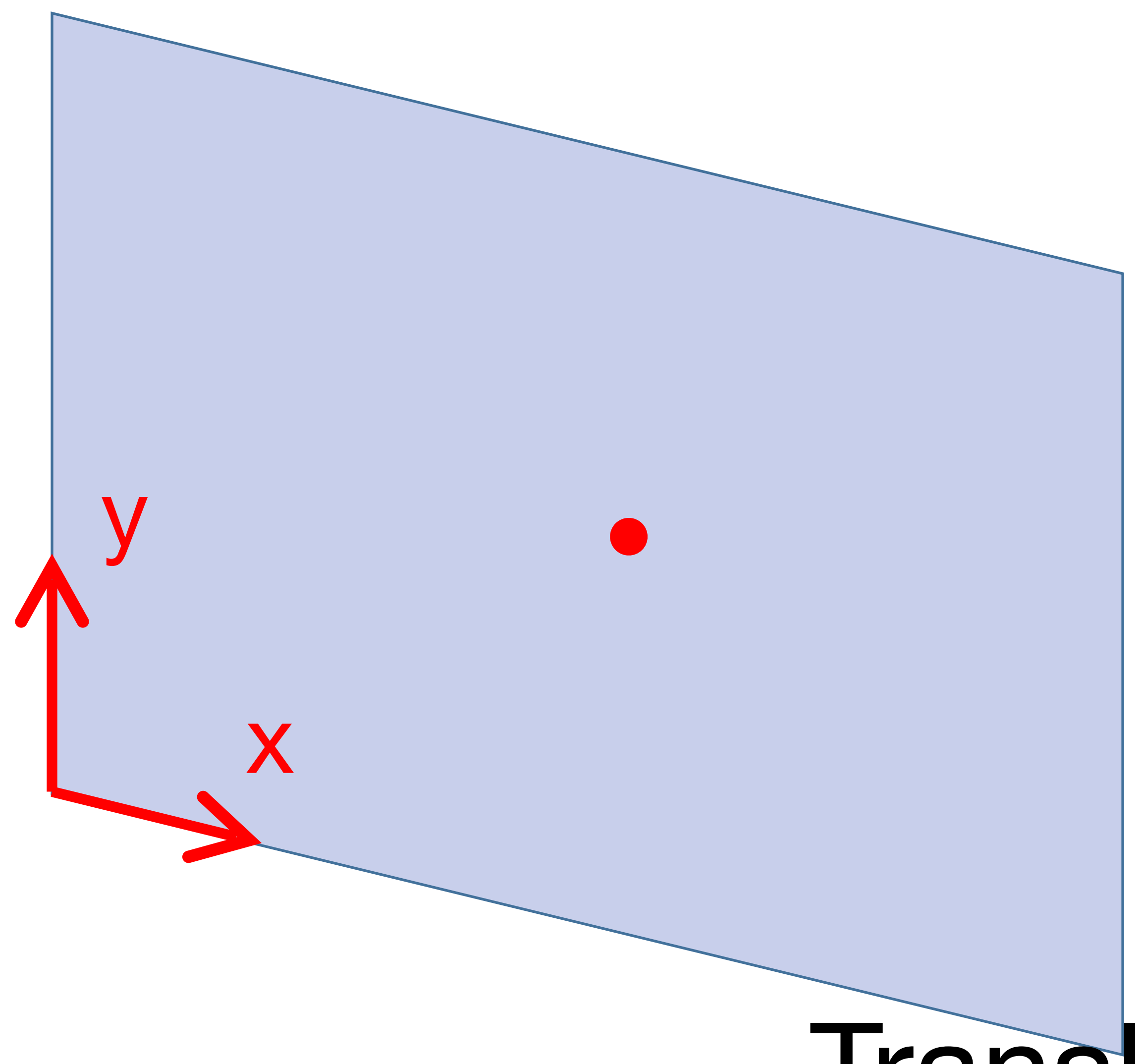
- Point on a plane: origin and 2 orthogonal coordinate axes.
- Points on the surface of a sphere:
  - “Explicit representation”: latitude and longitude

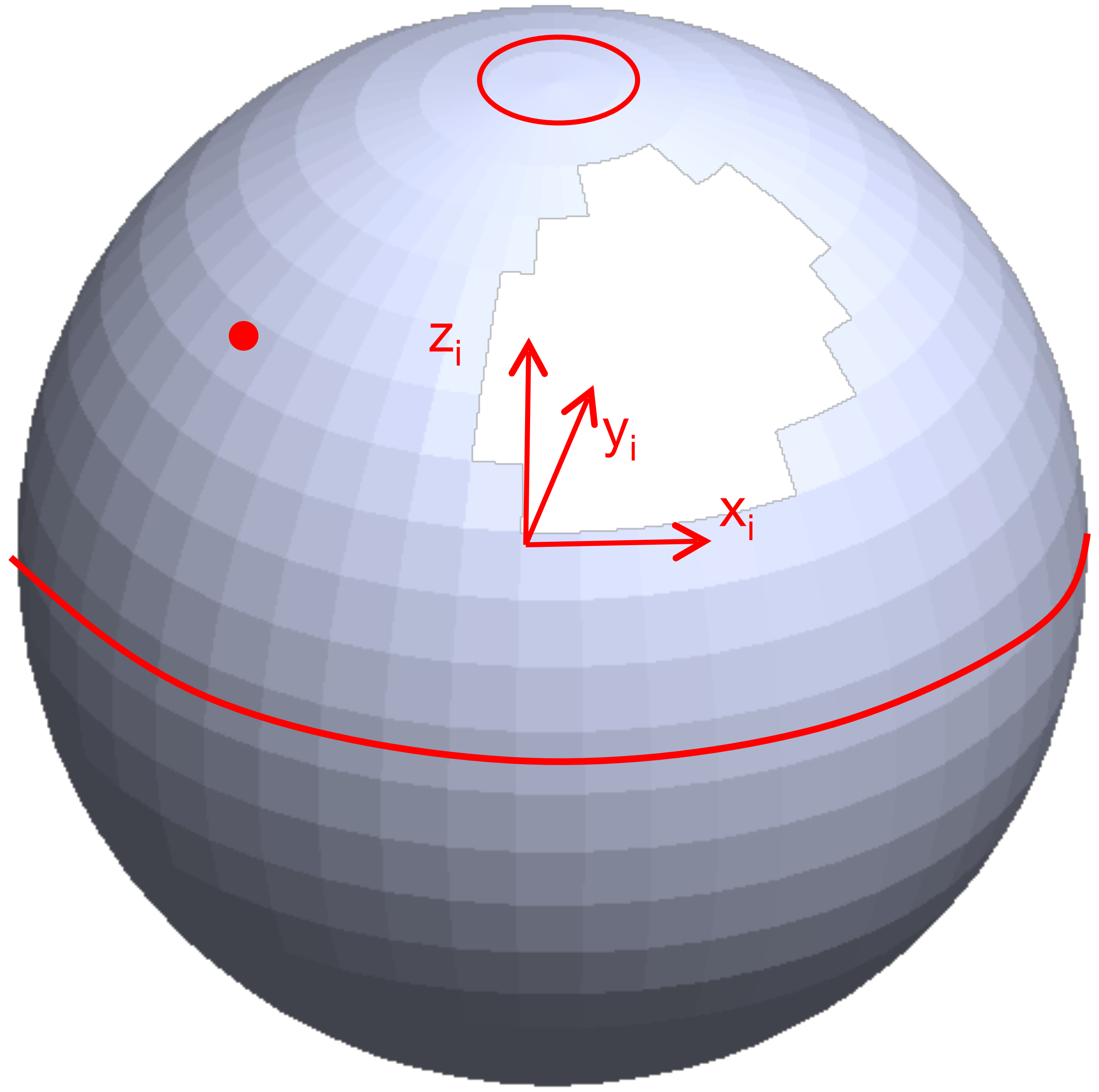




Topology Representation

- Point on a plane: origin and 2 orthogonal coordinate axes.
- Points on the surface of a sphere:
  - “Explicit representation”: latitude and longitude
  - “Implicit representation”: {X, Y, Z} such that $x^2+y^2+z^2 = 1$
    - More complex, but singularity free!
    - 3D — rotation matrix

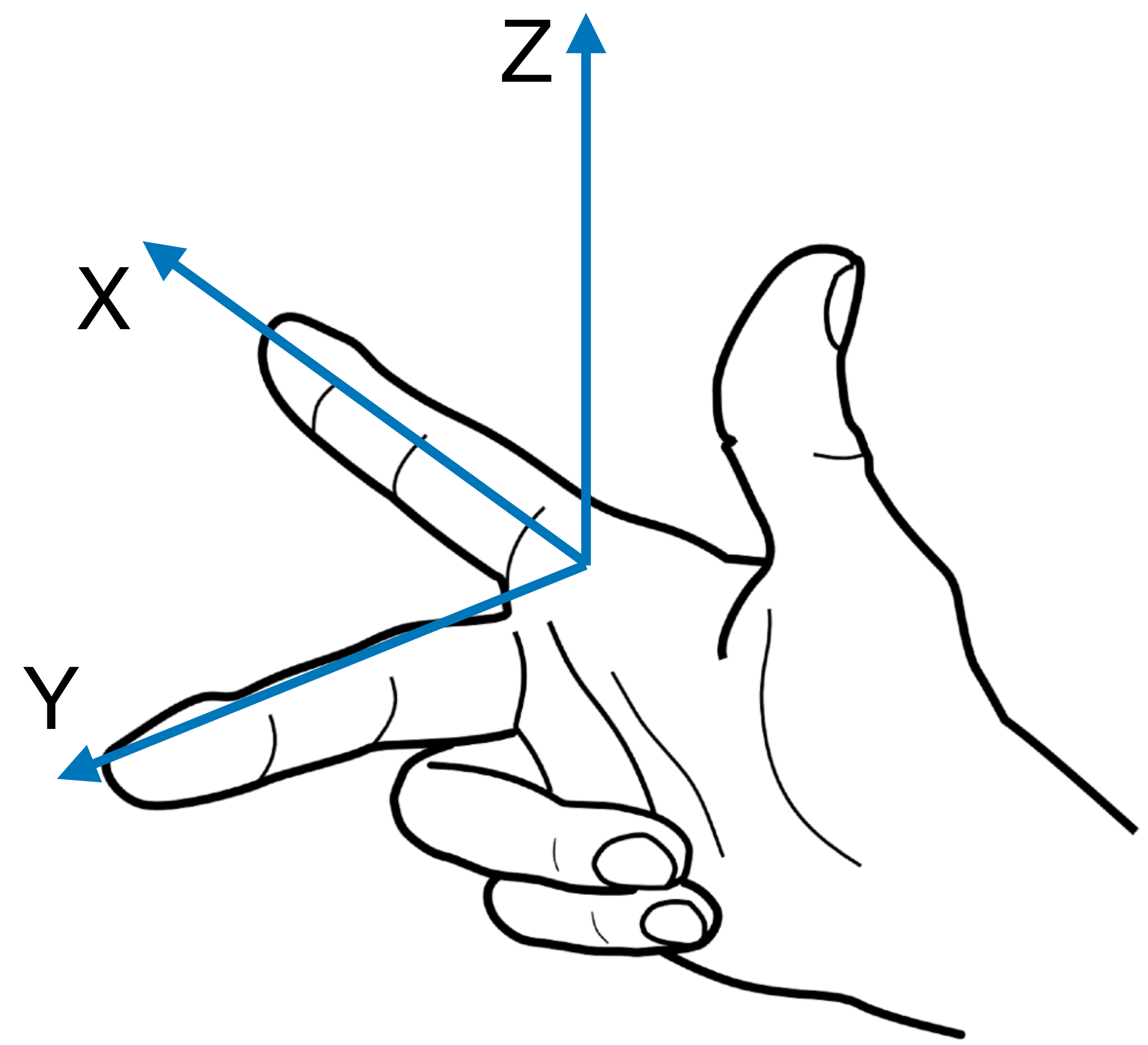




Translations are easy, rotations require more careful consideration

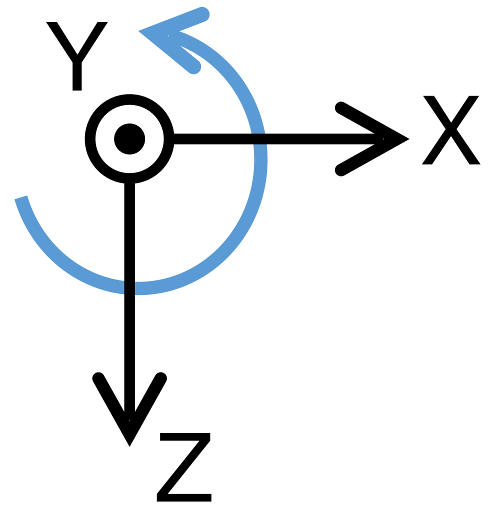
Coordinate Frames/ Conventions

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

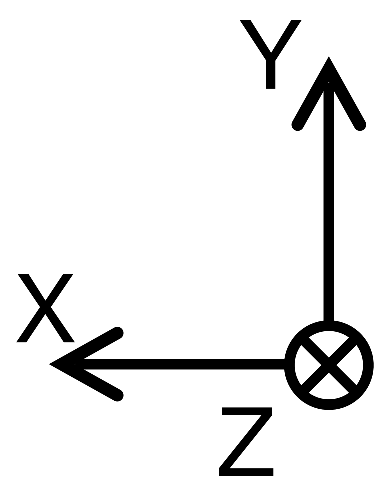


Are these valid right-hand frames and rotations?

A

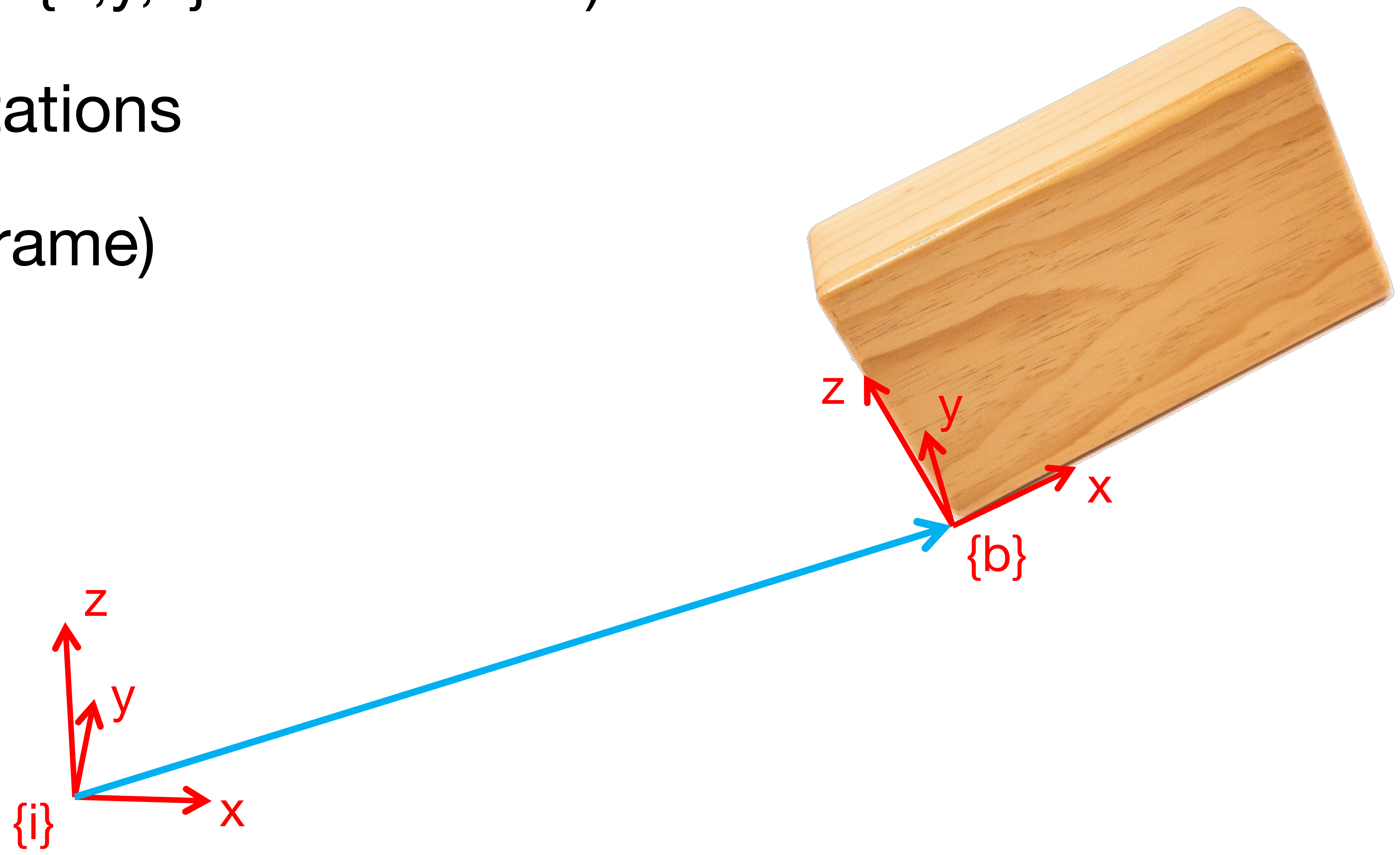


B



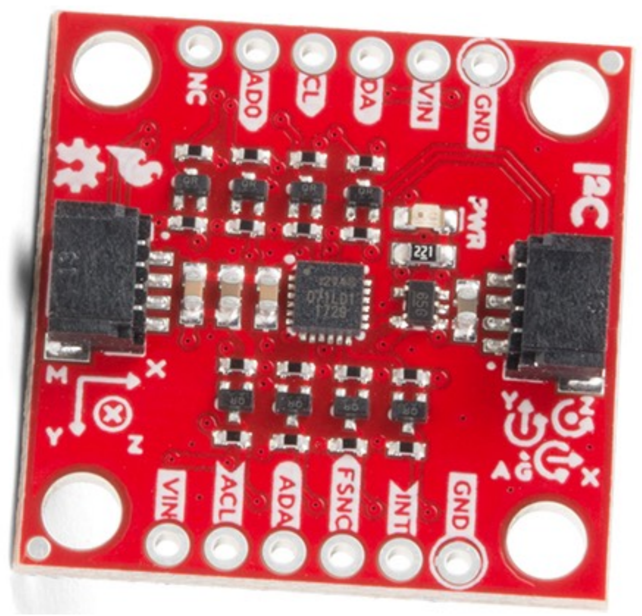
Coordinate Frames

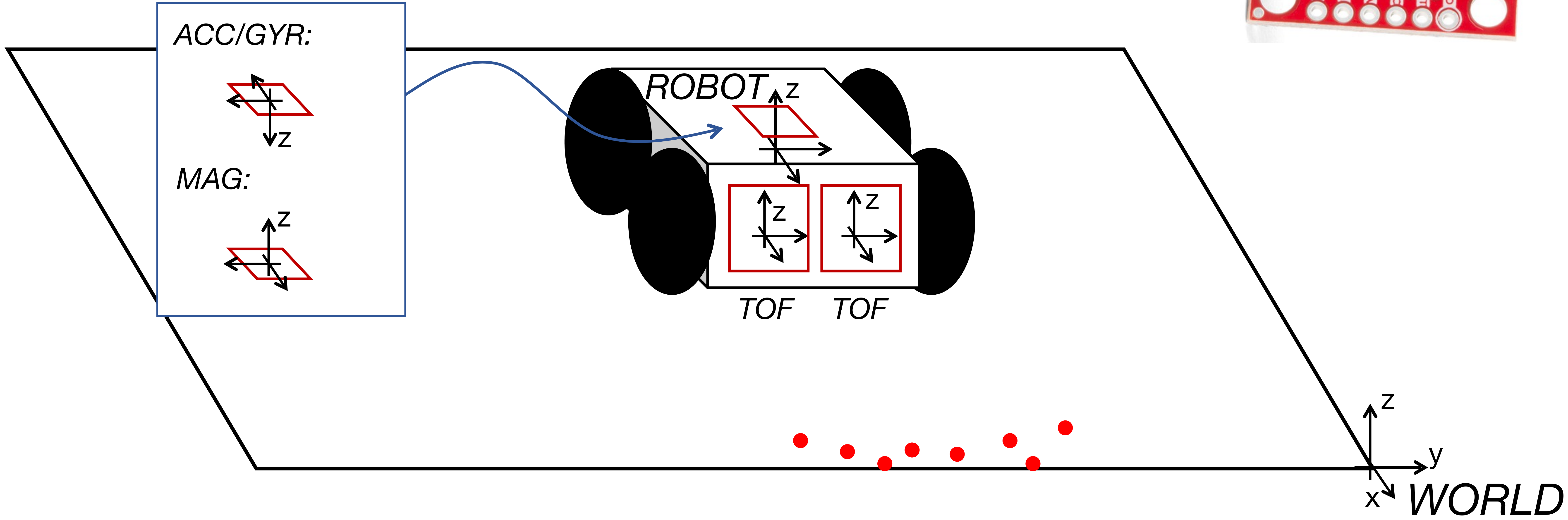
- Reference frames (origin and {x,y,z}-coordinates)
  - Right hand frames and rotations
- Inertial frame (world/space frame)
- Body frame



Coordinate Frames

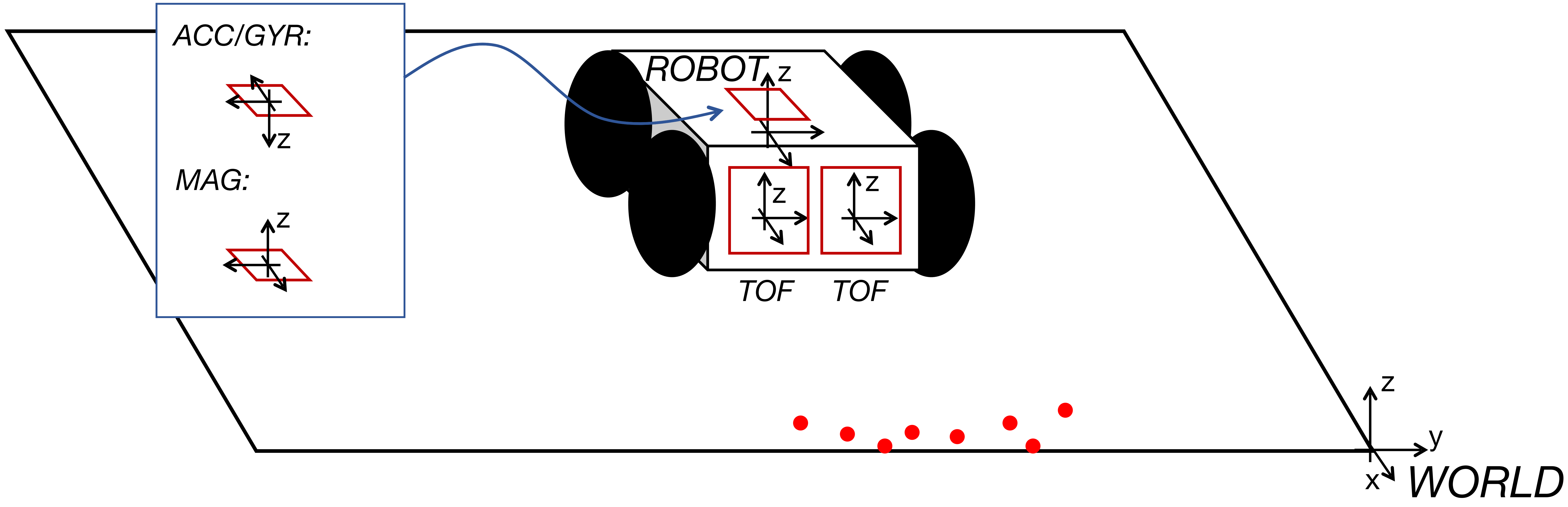
- Reference frames (origin and {x,y,z}-coordinates)
  - Right hand frames and rotations
- Inertial frame (world/space frame)
- Body frame





Homogeneous Transformation matrix

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



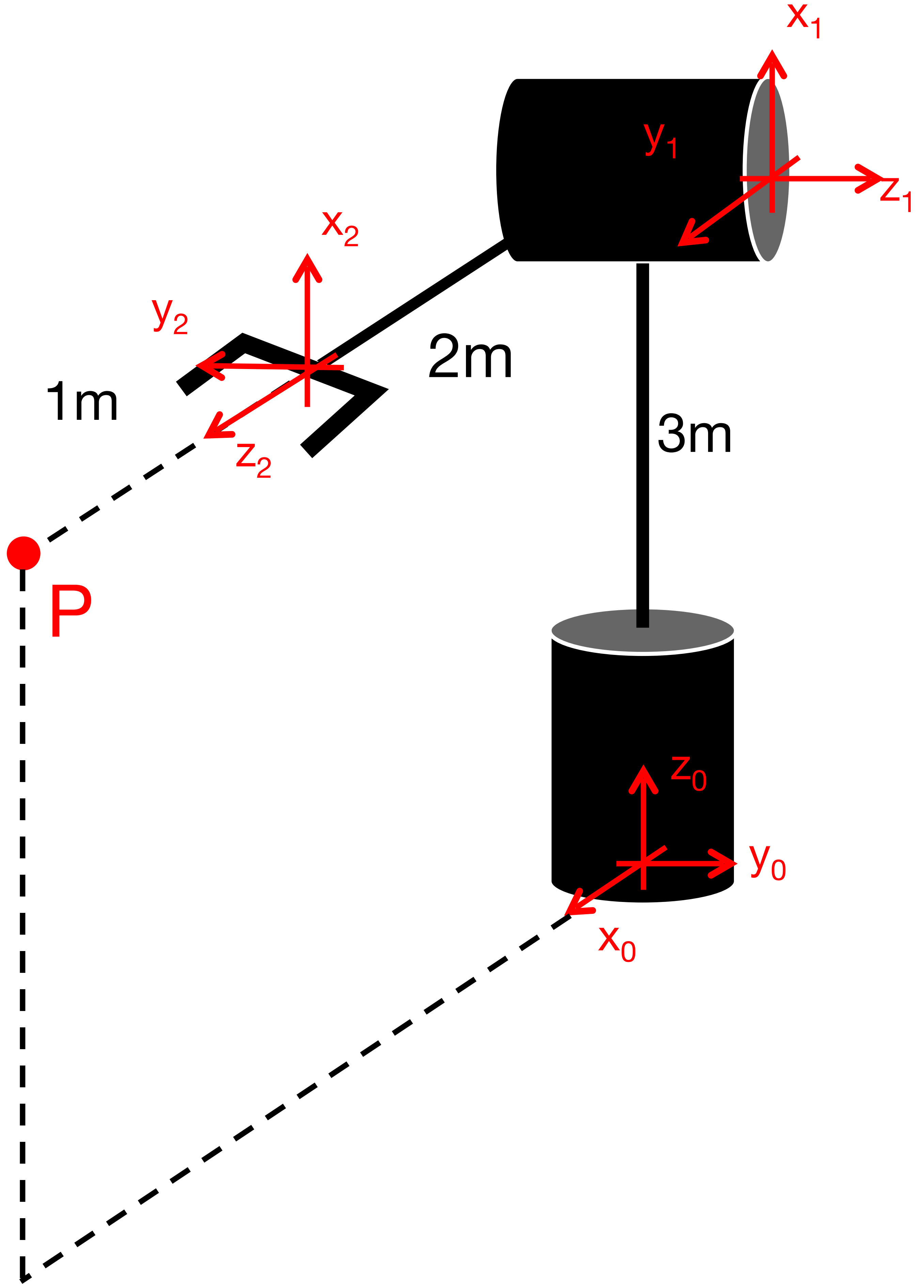
Homogeneous Transformation matrix

- 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?

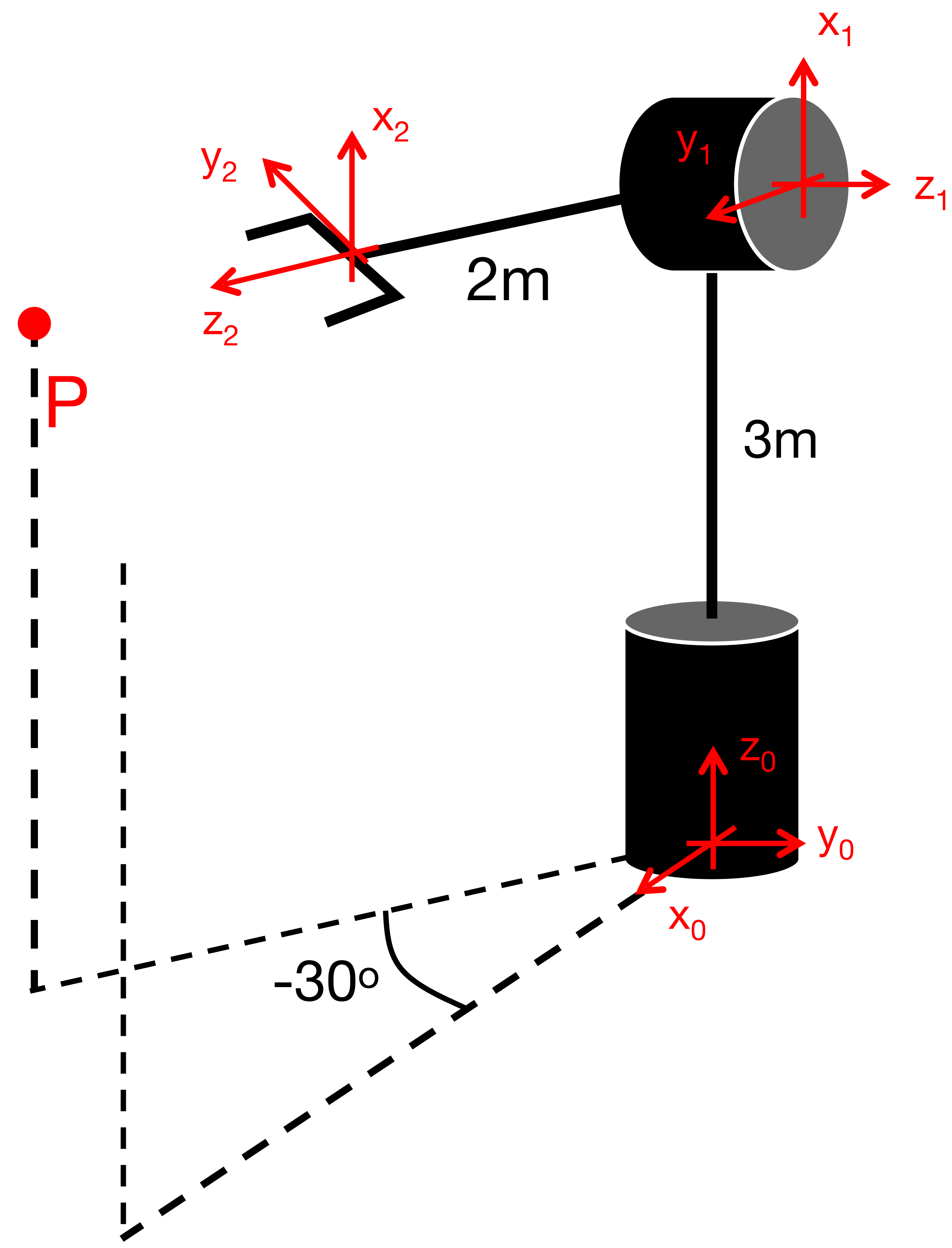
$$P^0 = \begin{bmatrix} 3 \\ 0 \\ 3 \end{bmatrix}$$



Homogeneous Transformation matrix

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

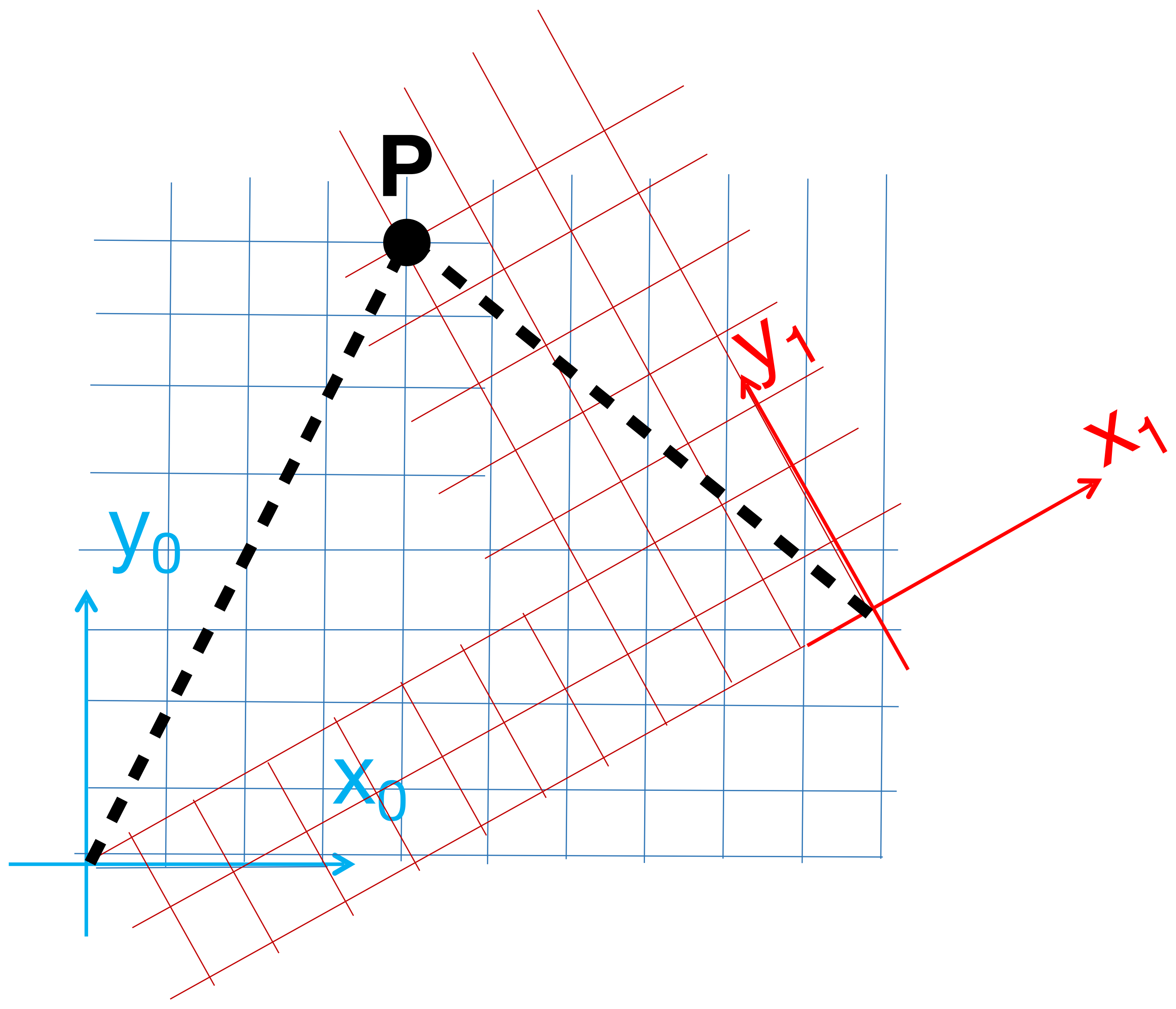
- What is the location of point P in reference frame 0?



Homogeneous Transformation matrix

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

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



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

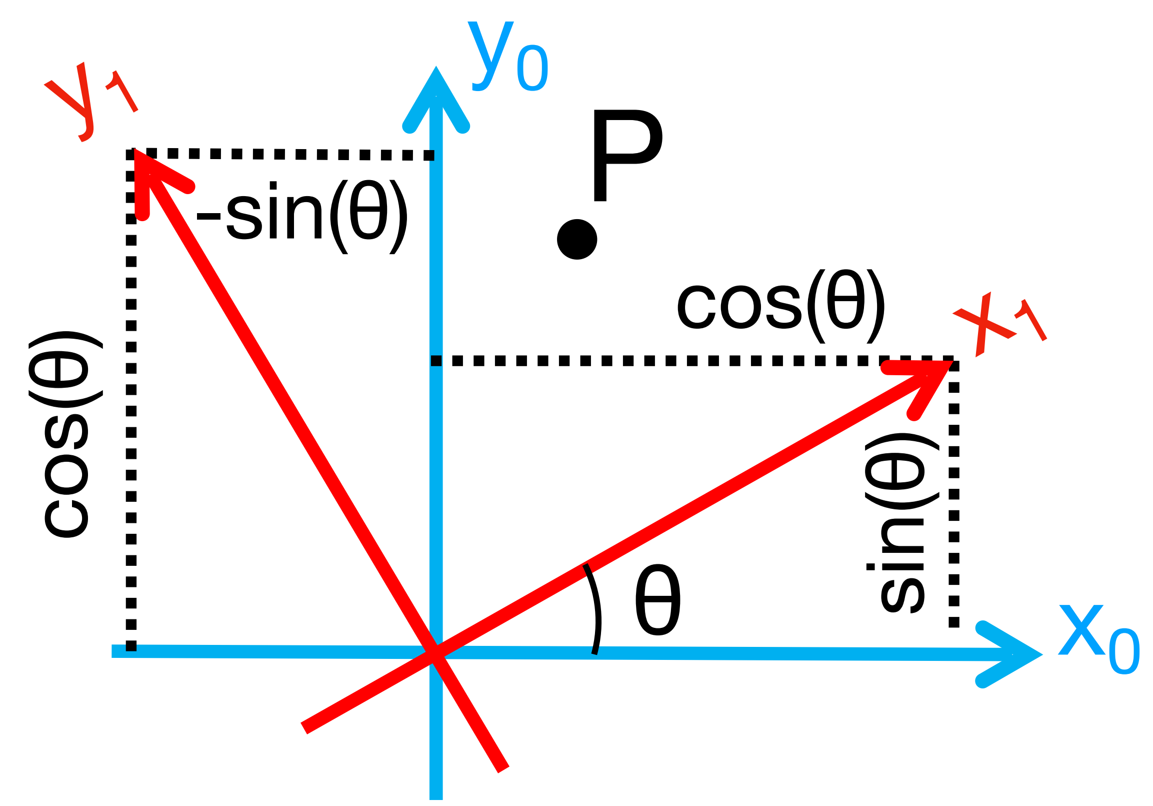
$$P^0 \cancel{=} P^1 + O_1^0$$

Homogeneous Transformation matrix

- We need both translation and rotation!

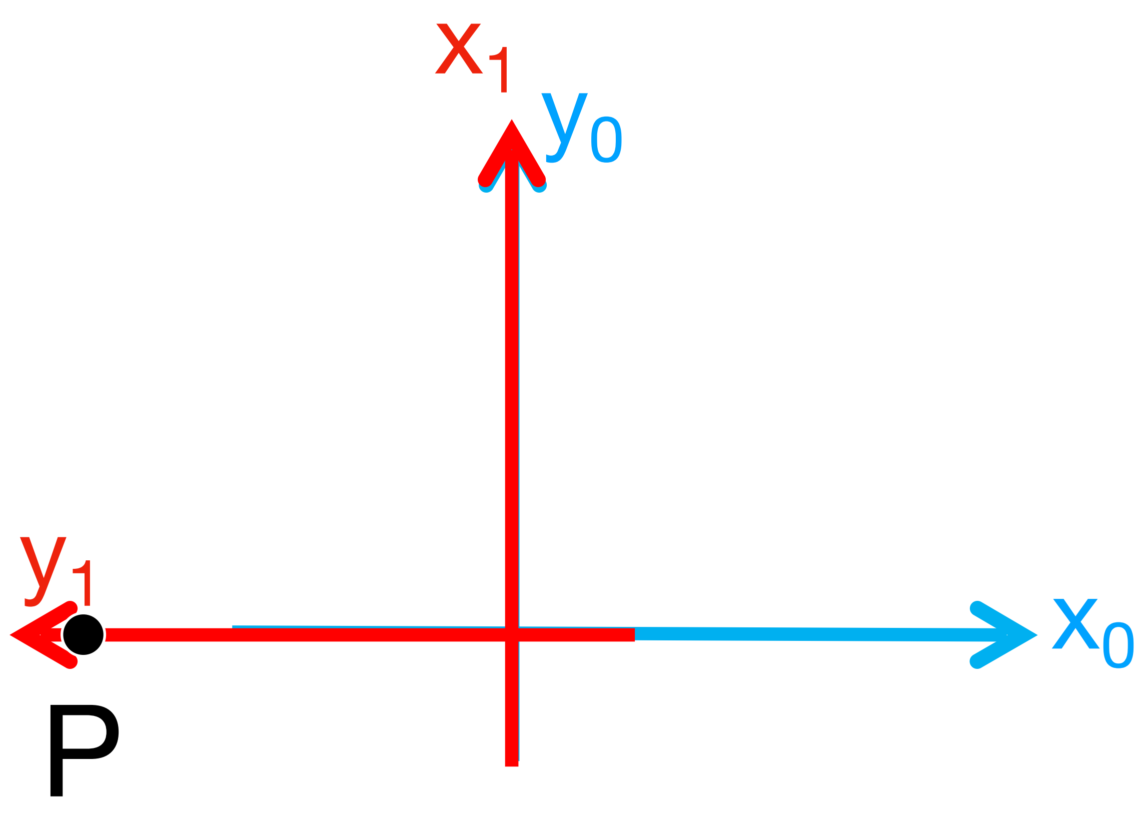
$$R_1^0 = \begin{bmatrix} x_1^0 & y_1^0 \end{bmatrix} \qquad x_1^0 = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \qquad y_1^0 = \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} \qquad R_1^0 = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

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



Example: $P^1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \theta = 90^\circ$

$$P^0 = R_1^0 P^1$$

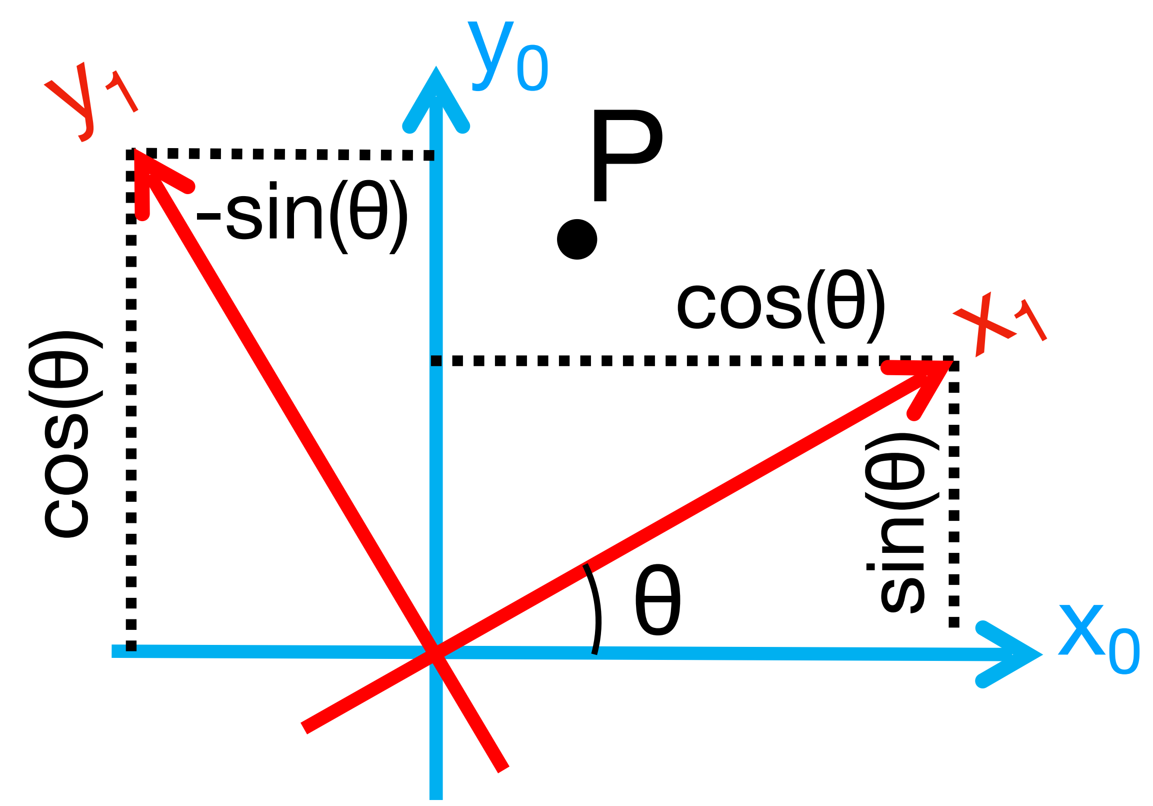


Homogeneous Transformation matrix

- We need both translation and rotation!

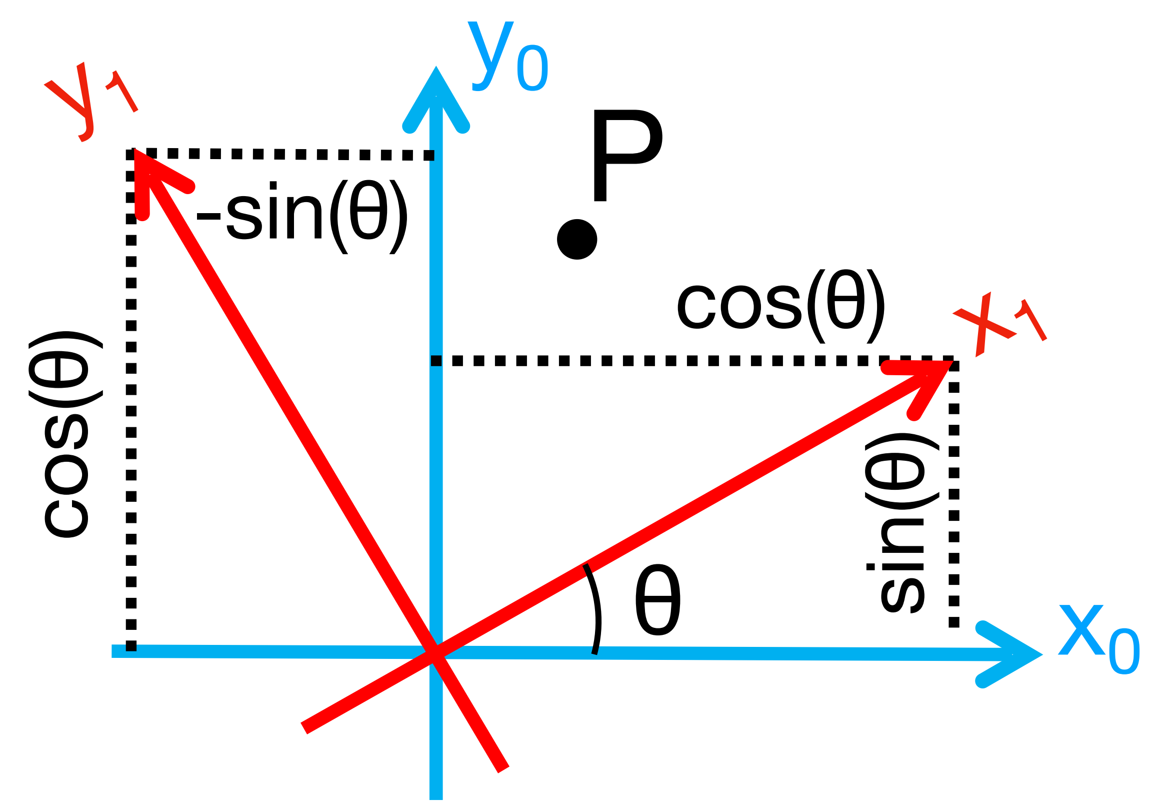
$$R_1^0 = \begin{bmatrix} x_1^0 & y_1^0 \end{bmatrix} \qquad x_1^0 = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \qquad y_1^0 = \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} \qquad R_1^0 = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

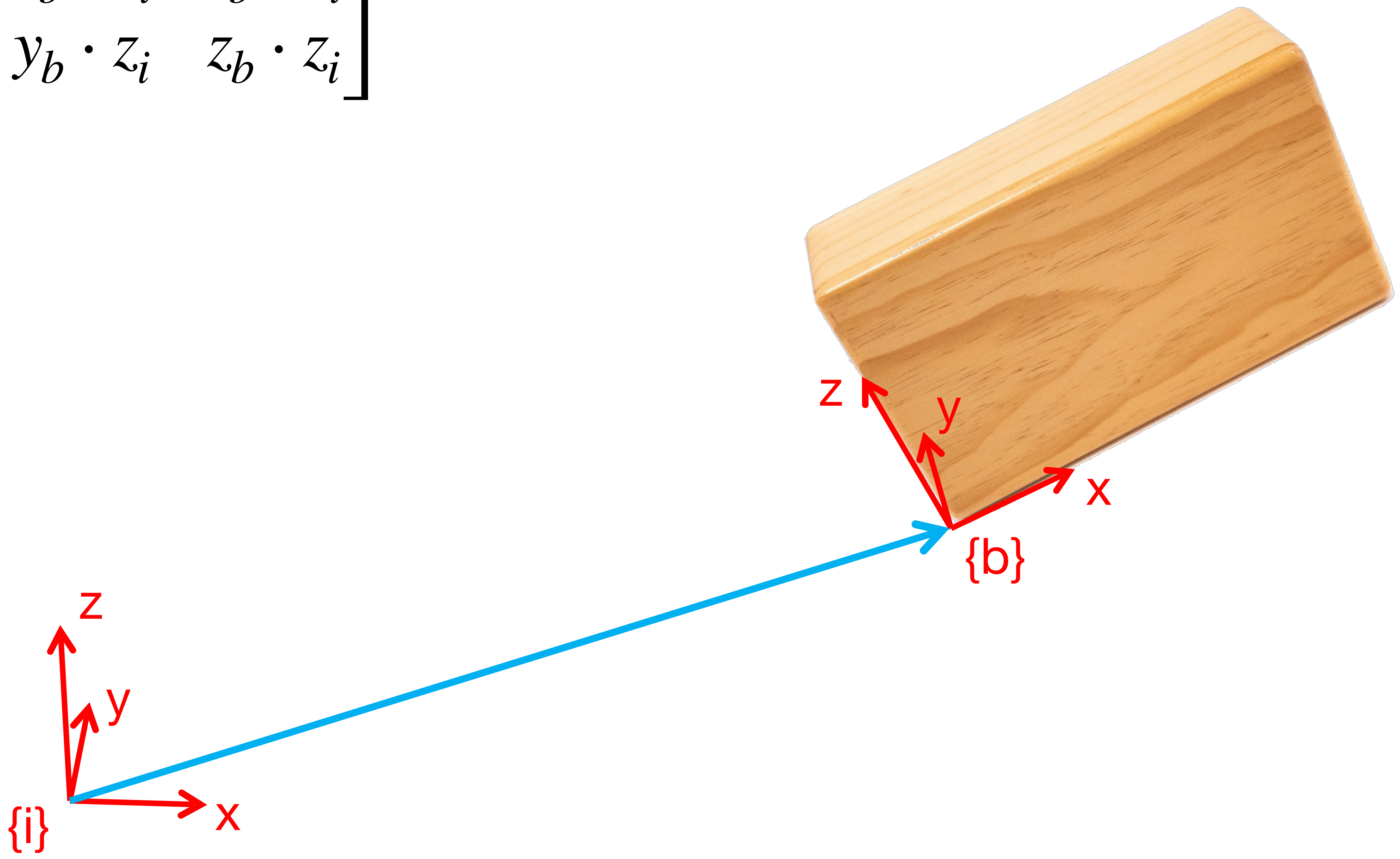
$$R_1^0 = \begin{bmatrix} x_0 \cdot x_1 & y_0 \cdot x_1 \\ x_0 \cdot y_1 & y_0 \cdot y_1 \end{bmatrix}$$



Rotation matrix in 3D

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



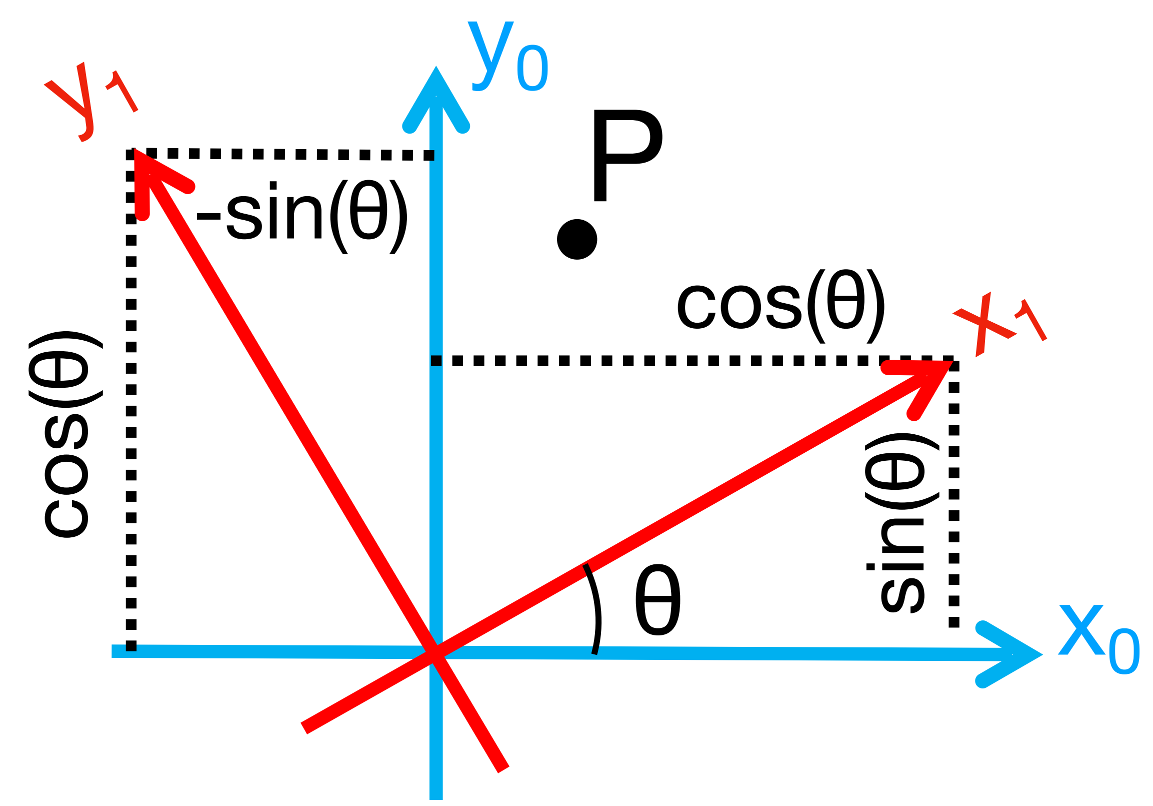


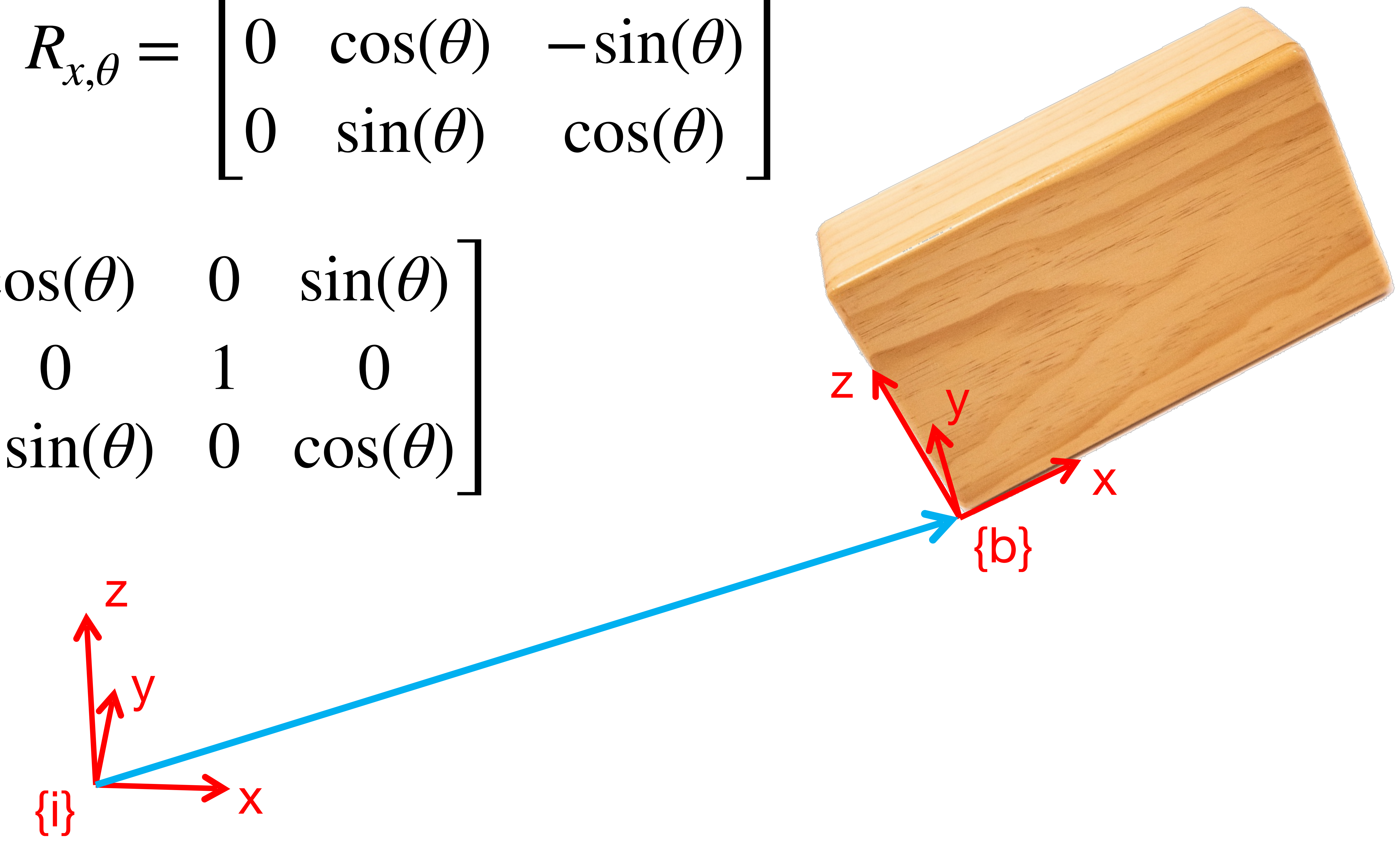
Rotation matrix in 3D

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

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

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



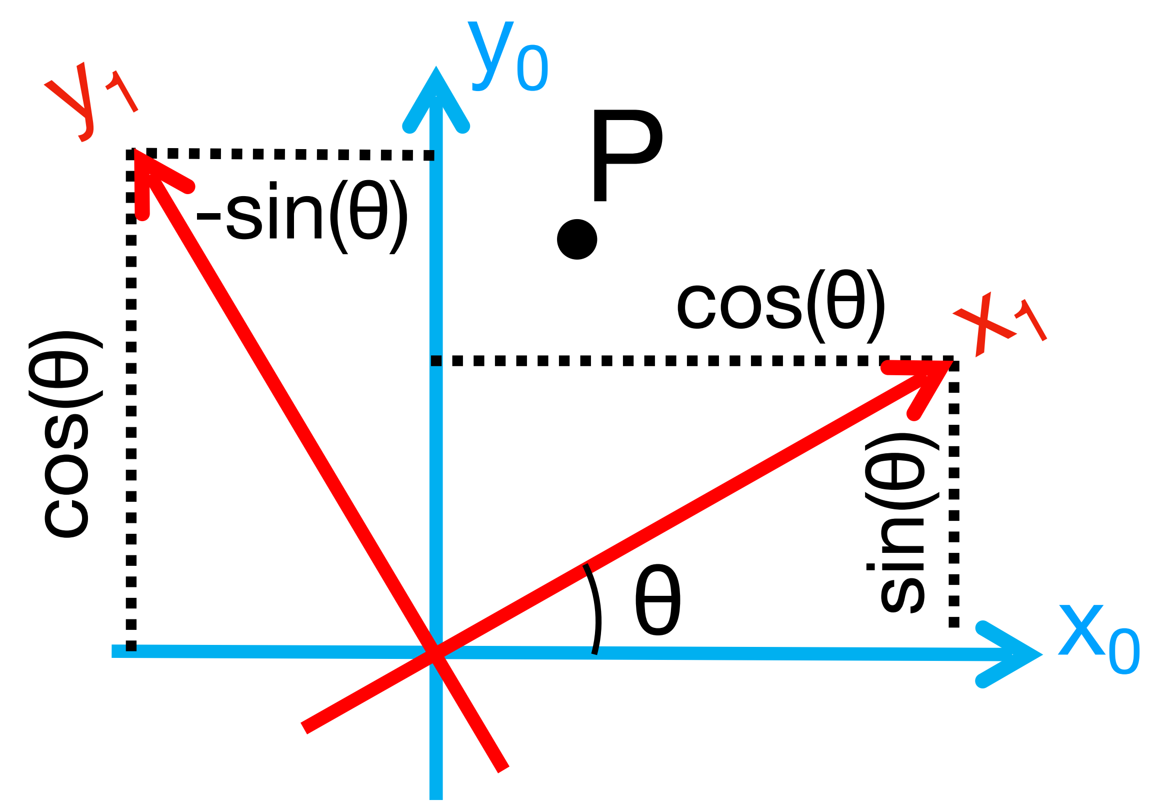


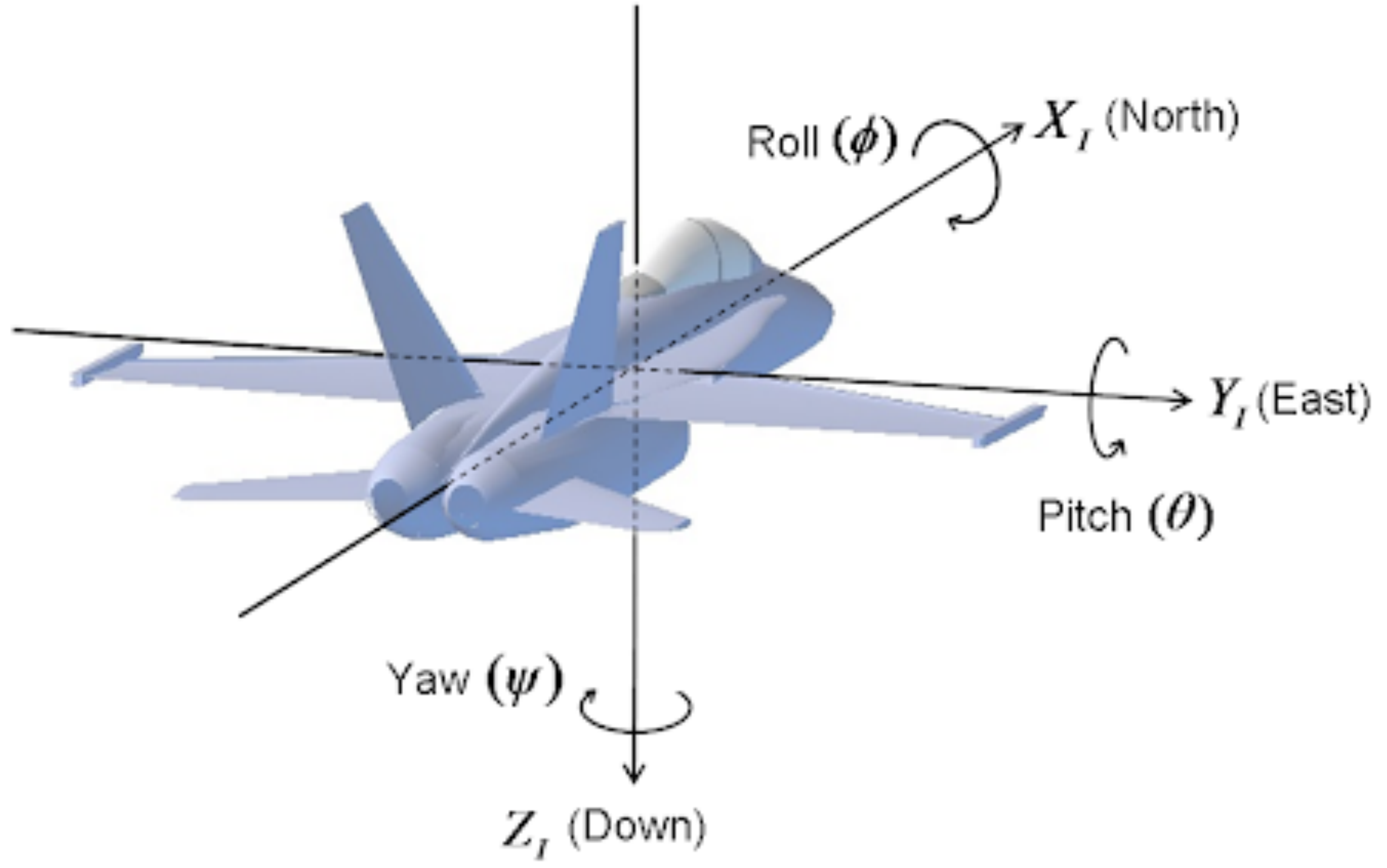
Rotation matrix in 3D

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

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

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



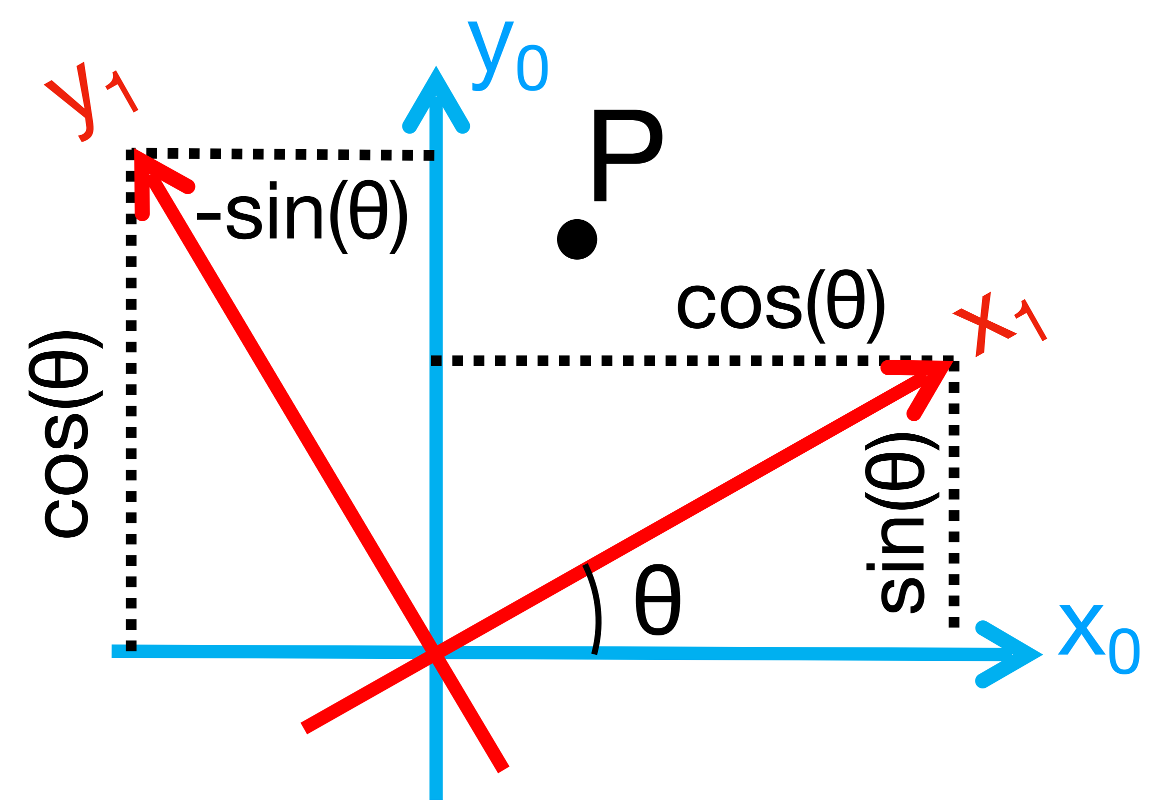


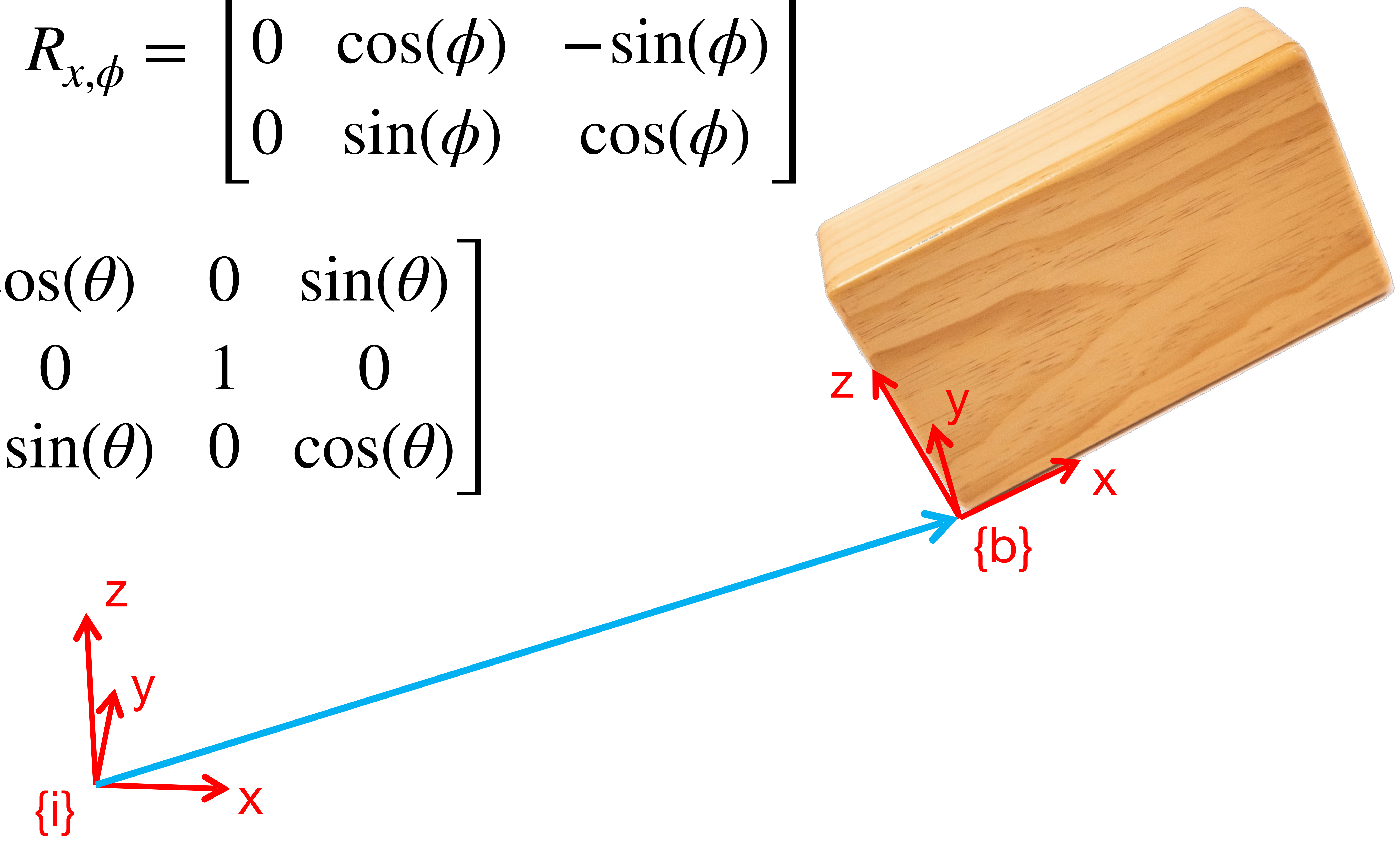
Rotation matrix in 3D

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

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

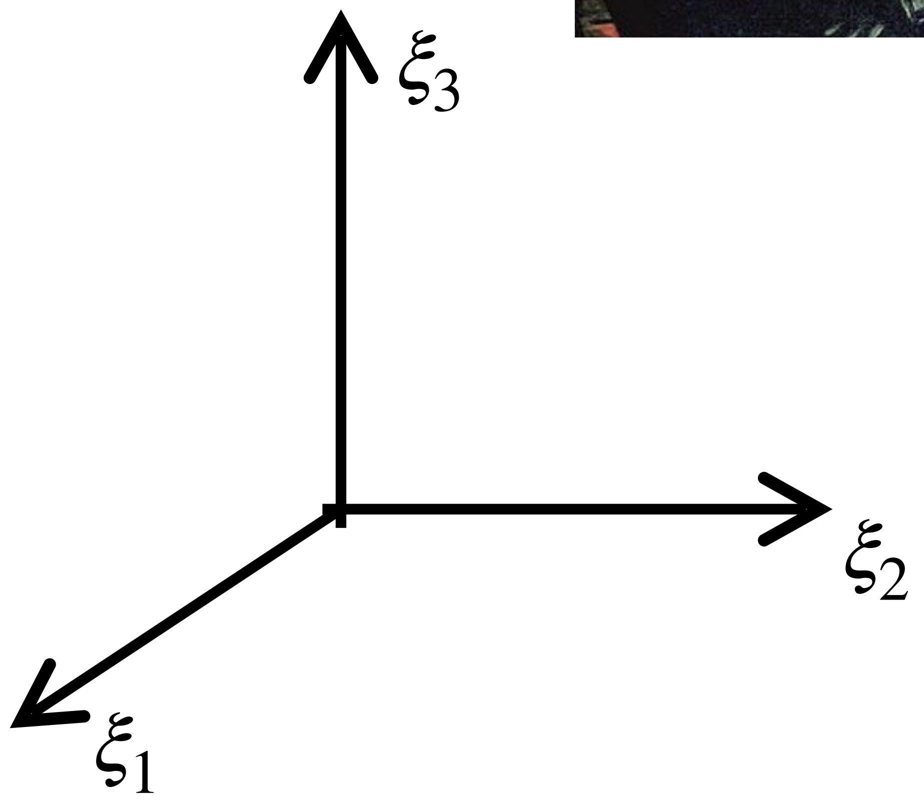
$$R_{y,\theta} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$





Euler

- "Any rotation can be described by three successive rotations about linearly independent axes"
  - 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-z, y-x-z
  - Most commonly used are z-y-z or x-y-z

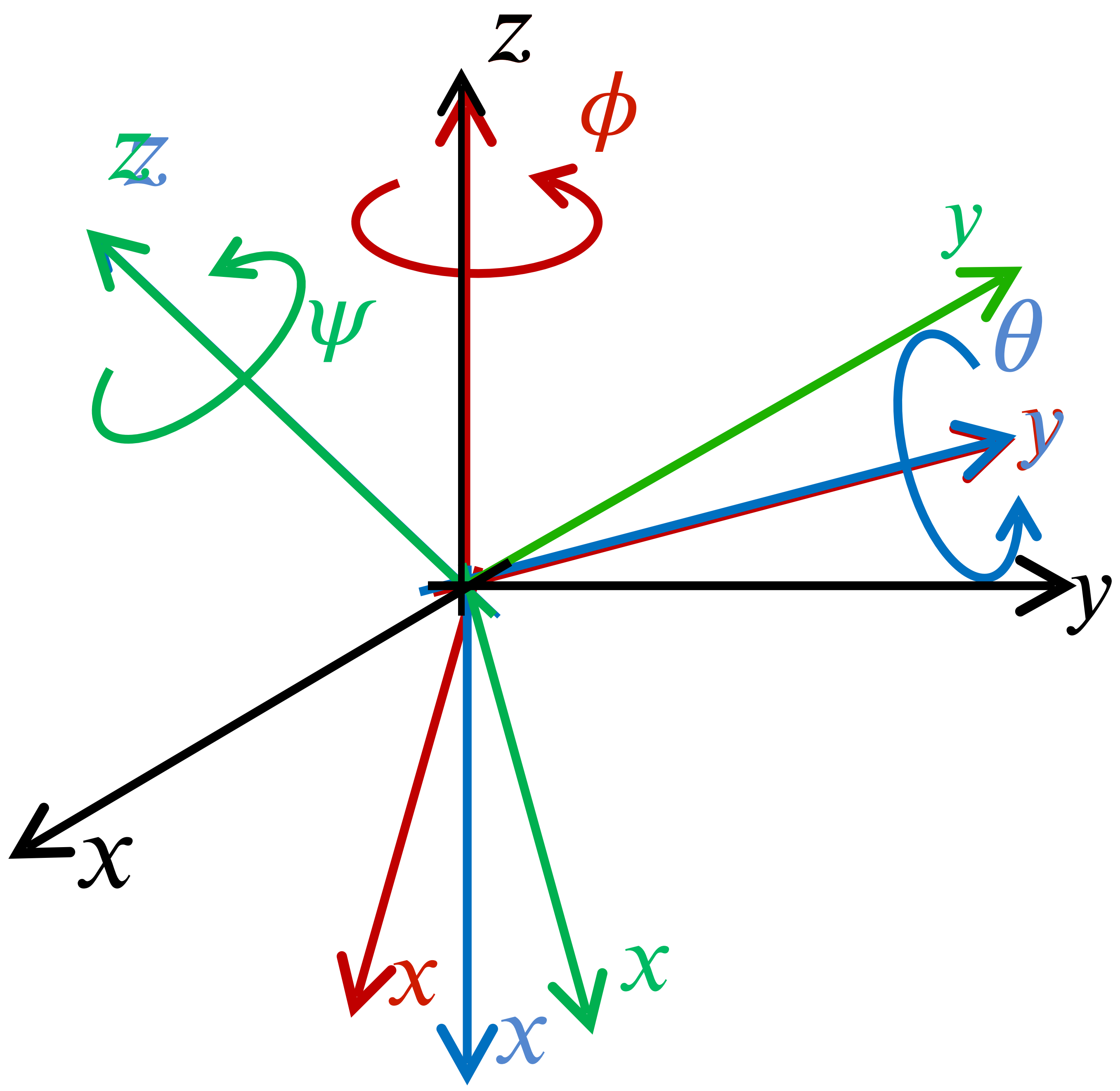


Rotation Matrix using ZYZ

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

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

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

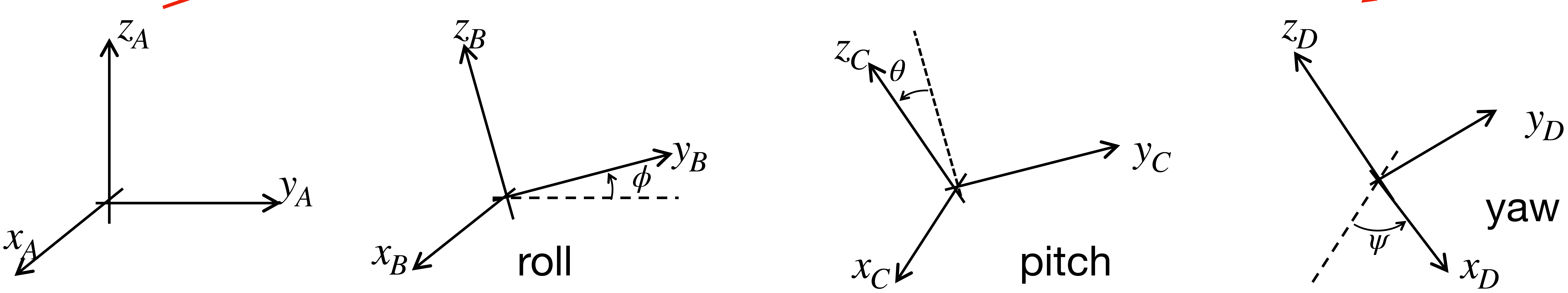


Rotation Matrix using Roll Pitch Yaw (XYZ)

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

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

$$R^A_D = R^A_B R^B_C R^C_D$$



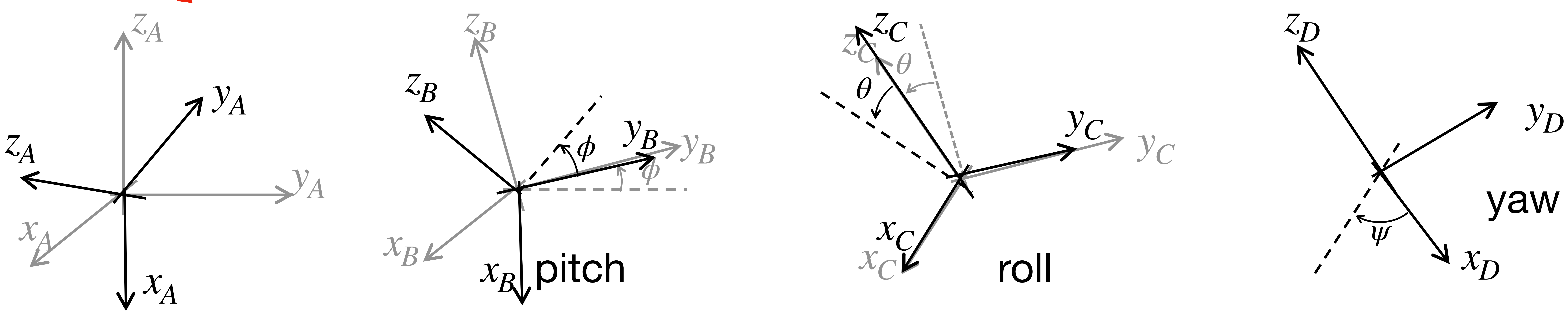
Rotation Matrix using Roll Pitch Yaw (XYZ)

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

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

Order matters!

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



Inverse Kinematics

How do we back out angles?

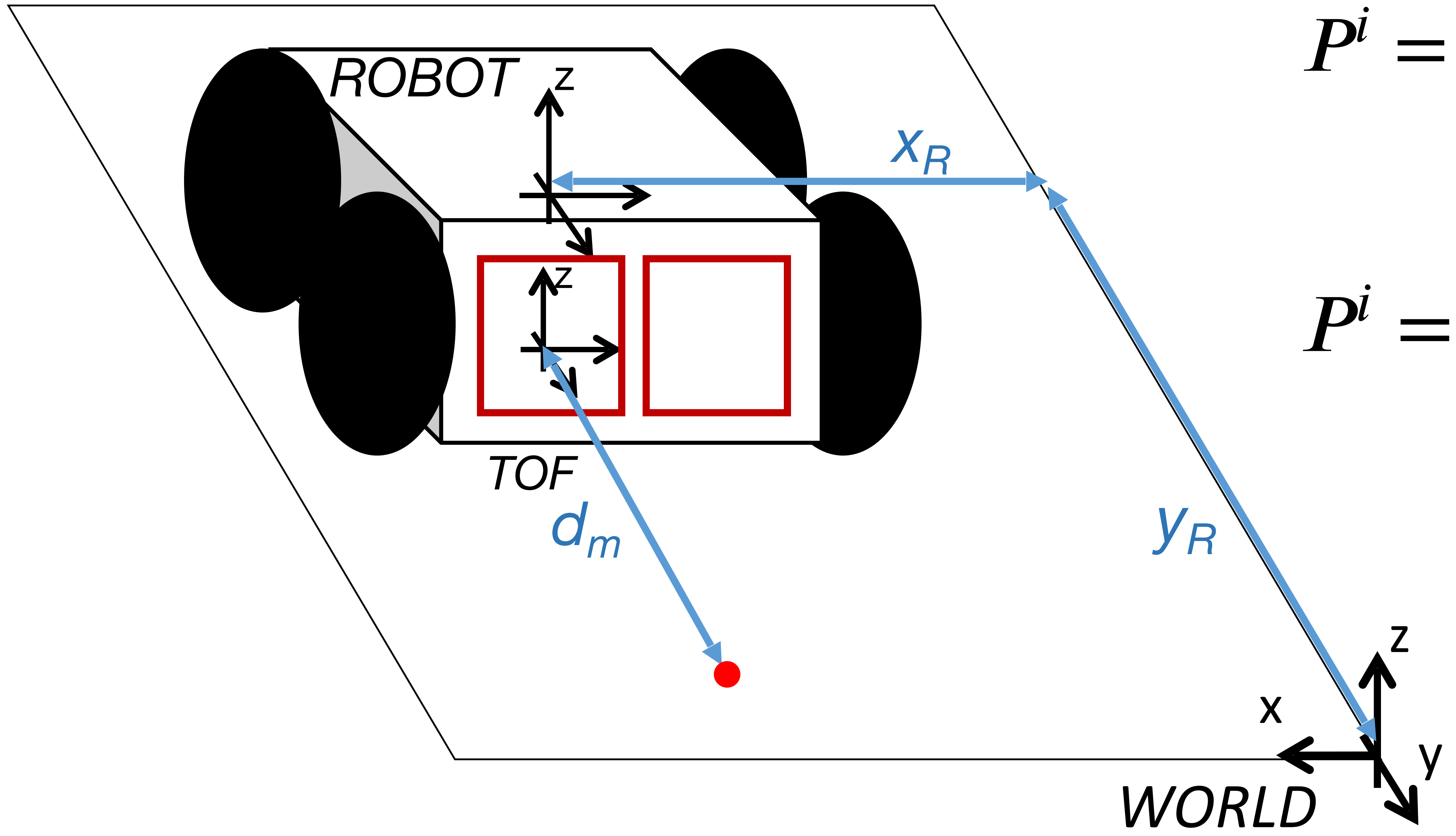
$$R_{XYZ} = R_{x,\phi}R_{y,\theta}R_{z,\psi} = \begin{bmatrix} c_\theta c_\psi & -c_\theta s_\psi & s_\theta \\ c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi - s_\phi s_\theta s_\psi & -s_\phi c_\theta \\ s_\phi s_\psi - c_\phi s_\theta c_\psi & s_\phi c_\psi + c_\phi s_\theta s_\psi & c_\phi c_\theta \end{bmatrix}$$

- The solution to acos is not unique
- atan(x) returns [-π/2, π/2]
- Instead use atan2(adj,opp)* which returns [-π, π]
  - $\theta = \mathrm{asin}(r_{13})$
    $\phi = \mathrm{atan2}(-r_{23}, r_{33})$
    $\psi = \mathrm{atan2}(-r_{12}, r_{11})$
- Special case if r13 = 1 (the z-axis is parallel to x-axis)
  - $\theta = 90^\circ,\ \psi = \mathrm{atan2}(r_{21}, r_{22}),\ \phi = 0^\circ$

```
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
}
```

Homogeneous Transformation Matrix

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

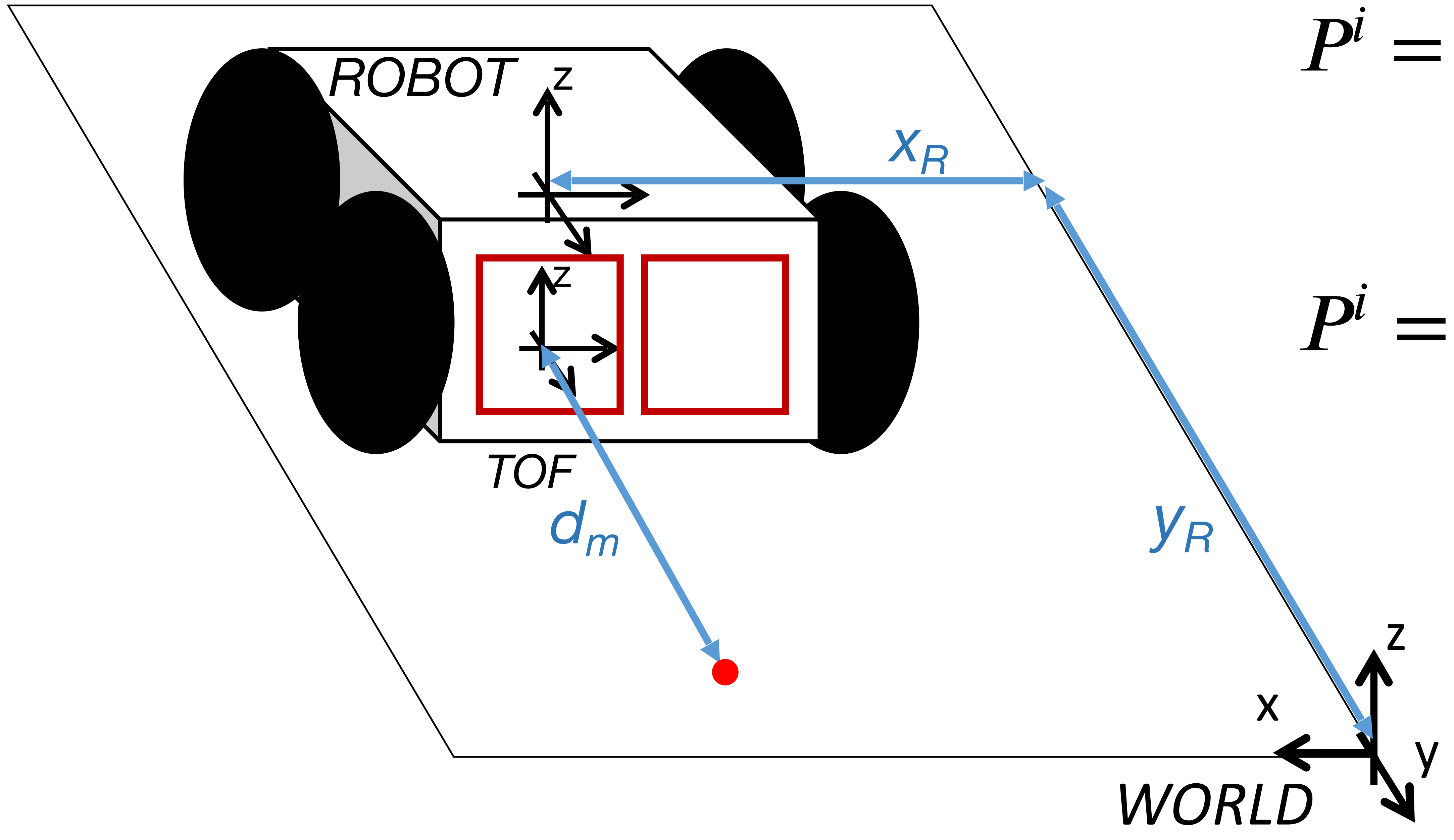


$$P^i = T^i_R T^R_{TOF} P^{TOF}$$

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

Homogeneous Transformation Matrix

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



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

$$P^i = \begin{bmatrix} c_\psi & -s_\psi & 0 & X_R \\ s_\psi & c_\psi & 0 & -y_R \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0.08 \\ 0 & 1 & 0 & -0.015 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_m \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

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

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

Sources and references

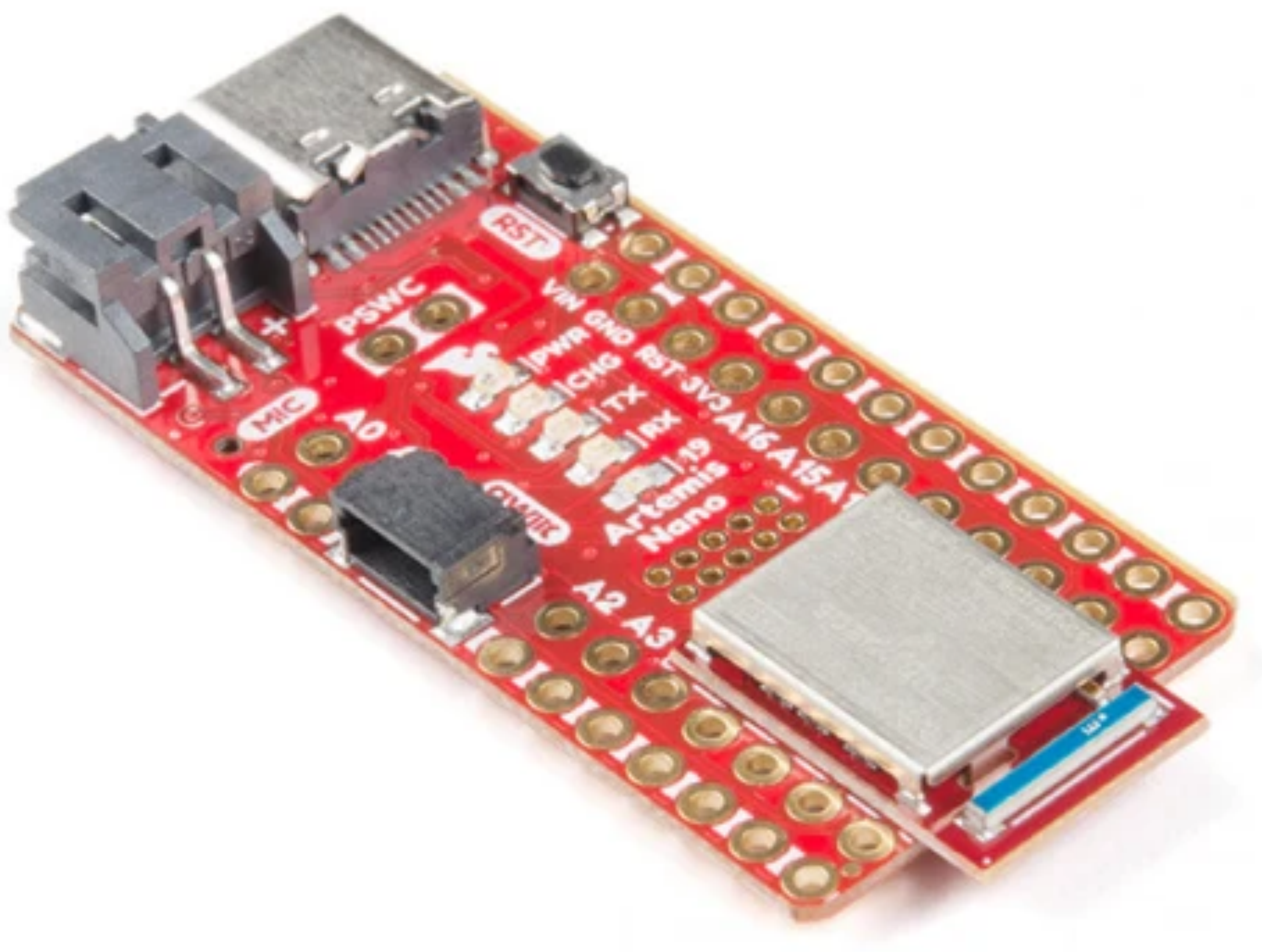
- Northwestern University, course on Modern Robotics
- UPenn Coursera course on Aerial Robtoics
- MilfordRobotics YouTube stream
- Mecademic
- Prof. Kirstin Petersen

Lab 1: Artemis and Bluetooth

Lab 1: Artemis and Bluetooth

Redboard Artemis Nano

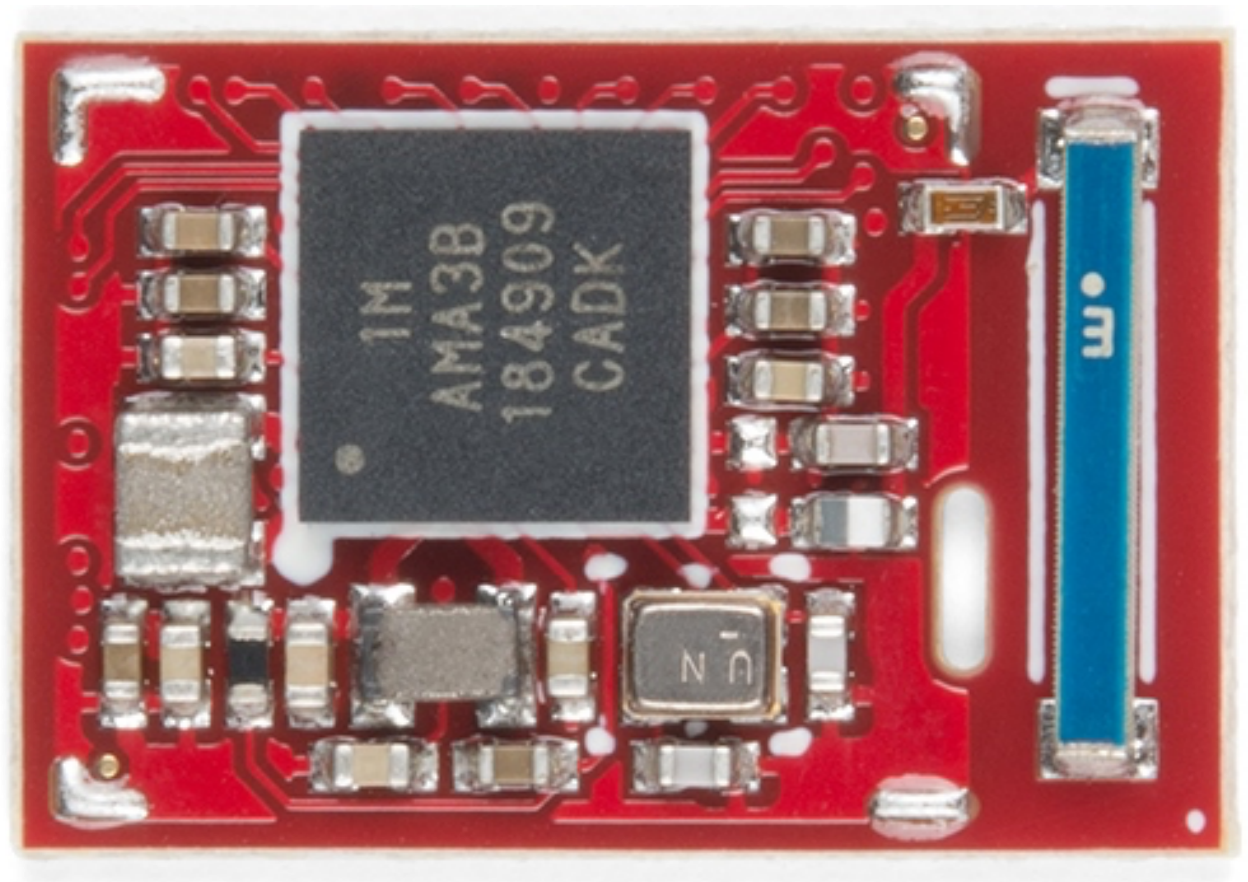
- Board: https://www.sparkfun.com/sparkfun-redboard-artemis-nano.html
- Integrating with Arduino: https://learn.sparkfun.com/tutorials/artemis-development-with-the-arduino-ide
- Sparkfun Community: https://community.sparkfun.com/
- Features:
  - USB programming/charging
  - LiPo JST connector
  - Qwiic connector
  - IMPORTANTLY it is a **3V BOARD**



Lab 1: Artemis and Bluetooth

Artemis module

- Bluetooth Low Energy Module
- PDM (pulse density microphone)
- ARM Cortex-M4F processor (floating point operations)
- SIMD operations
- 384KB RAM for data
- Fast control loop closure



ambiqmicro

Apollo3 Blue MCU Datasheet

Ultra-Low Power Apollo MCU Family

Features

Ultra-low supply current:
- 6 µA/MHz executing from FLASH or RAM at 3.3 V
- 1 µA deep sleep mode (BLE in shutdown) with RTC at 3.3 V

High-performance ARM Cortex-M4 Processor:
- 48 MHz nominal clock frequency, with 96 MHz high performance TurboSPOT™ Mode
- Floating point unit
- Memory protection unit
- Wake-up interrupt controller with 32 interrupts

Integrated Bluetooth[1] 5 low-energy module:
- RF sensitivity: -93 dBm (typical)
- TX: 3 mA @ 0 dBm, RX: 3 mA
- TX peak output power: 4.0 dBm (max)

Ultra-low power memory:
- Up to 1 MB of flash memory for code/data
- Up to 384 KB of low latency/leakage SRAM for code/data
- 16 KB 2-way Associative/Direct-Mapped Cache

Ultra-low power interface for on- and off-chip sensors:

- 3.37 x 3.25 mm (<0.35mm thk pkg) 66-pin WLCSP with 37 GPIO
- 3.37 x 3.25 mm (<0.30mm thk pkg) 66-pin WLCSP with 37 GPIO (Apollo3 Blue Thin MCU)

Applications

- Voice-on-SPOT™ compatible for always-listening keyword detect, audio command recognition and voice assistant integration in battery-powered devices including:
  - Bluetooth headsets, earbuds, and truly wireless earbuds
  - Remote and Gaming Controls
  - Smart home
- Wearables including smart watches and fitness/activity trackers
- Hearing aids, Digital Health Monitoring and Sensing Devices
- Smart Home Automation, Security and Lighting control

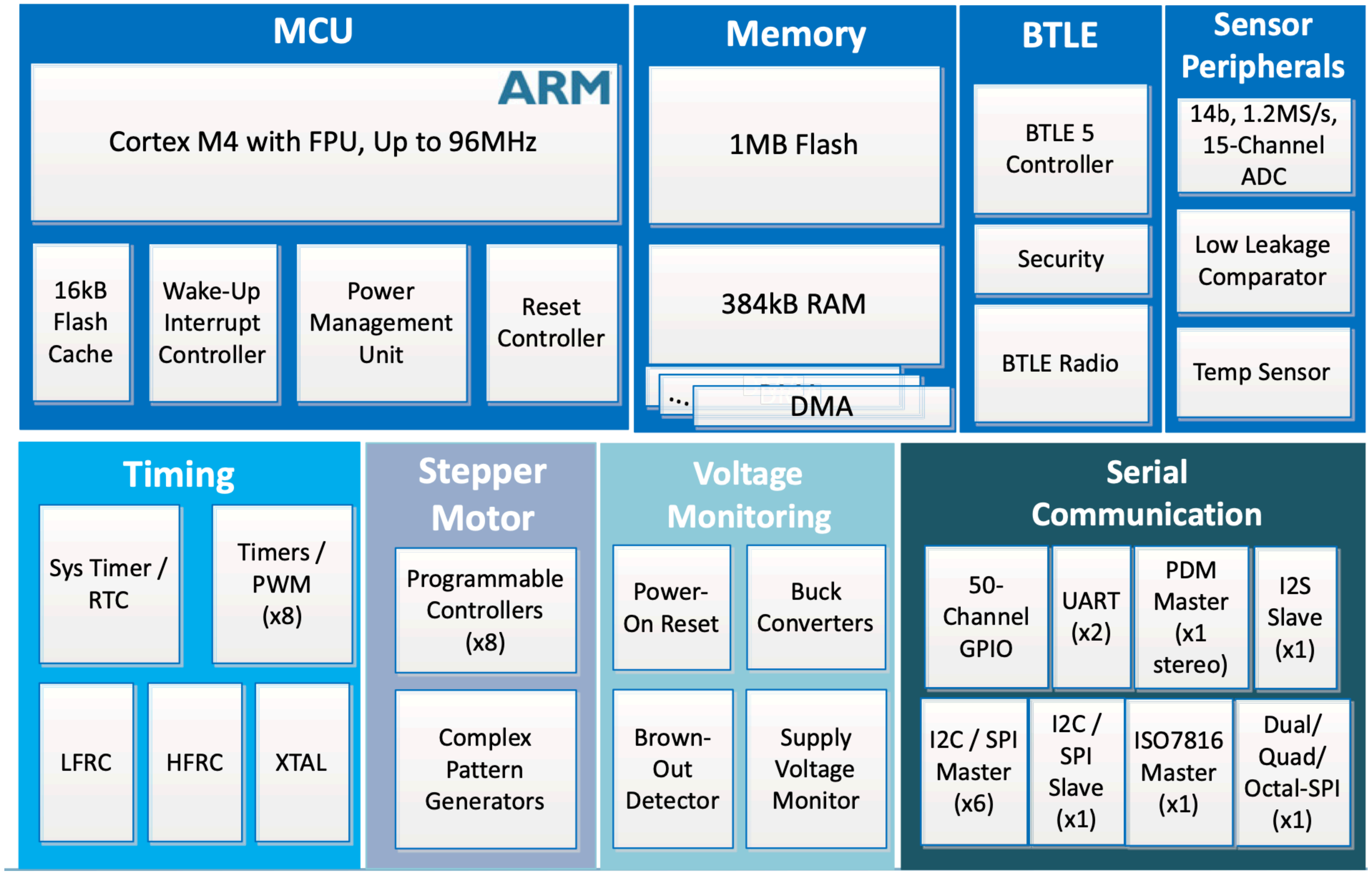
Description

The Apollo MCU Family is an ultra-low power, highly integrated microcontroller platform based on Ambiq Micro's patented Sub-threshold Power Optimized Technology (SPOT™) and designed

https://learn.sparkfun.com/tutorials/designing-with-the-sparkfun-artemis

Lab 1: Artemis and Bluetooth

Apollo3 Blue MCU



https://cdn.sparkfun.com/assets/1/5/c/6/7/Apollo3-Blue-MCU-Datasheet_v0_15_0.pdf

Lab 1: Artemis and Bluetooth

Course Software Development

- C (arduino) to program the real robot
- Python (Jupyter)
  - Offboard processing/ debugging/ send and receive data to/from real robot
  - Simulator (program your virtual robot
- Robot Commands in Lab 1B:
  - We give you a few commands to make sure you can communicate. You will be making many, many robot commands over this semester. The tasks ensure that you know how they work and that you can make your own.

Lab 1: Artemis and Bluetooth

Data Types

- Bluetooth: char
- ToF: unsigned int
- Serial.print: strings
- IMU: float
- PID: double
- millis(): unsigned long
- if-statements: bool

Lab 1: Artemis and Bluetooth

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}: 32 bits
  - Long_{64bit}: 64 bits

You can specify the length:
- int16_t
- uint32_t

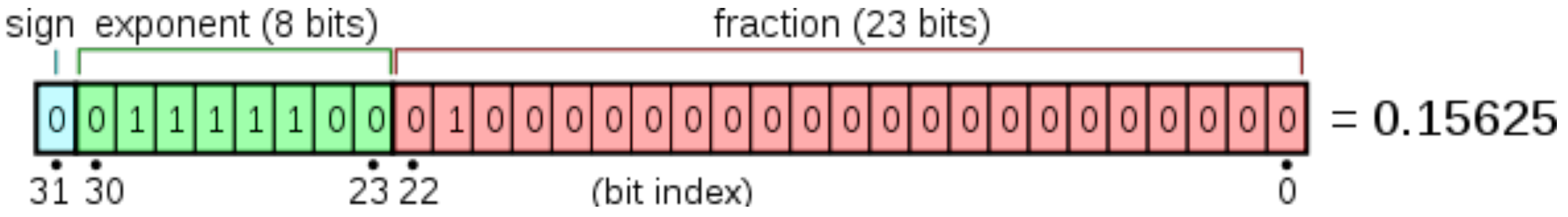
Bool:

Range:
- Signed char_{32bit}:
- Unsigned char_{32bit}:
- Int_{32bit}:

Lab 1: Artemis and Bluetooth

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 ~ 3.4028235×10^{38}



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

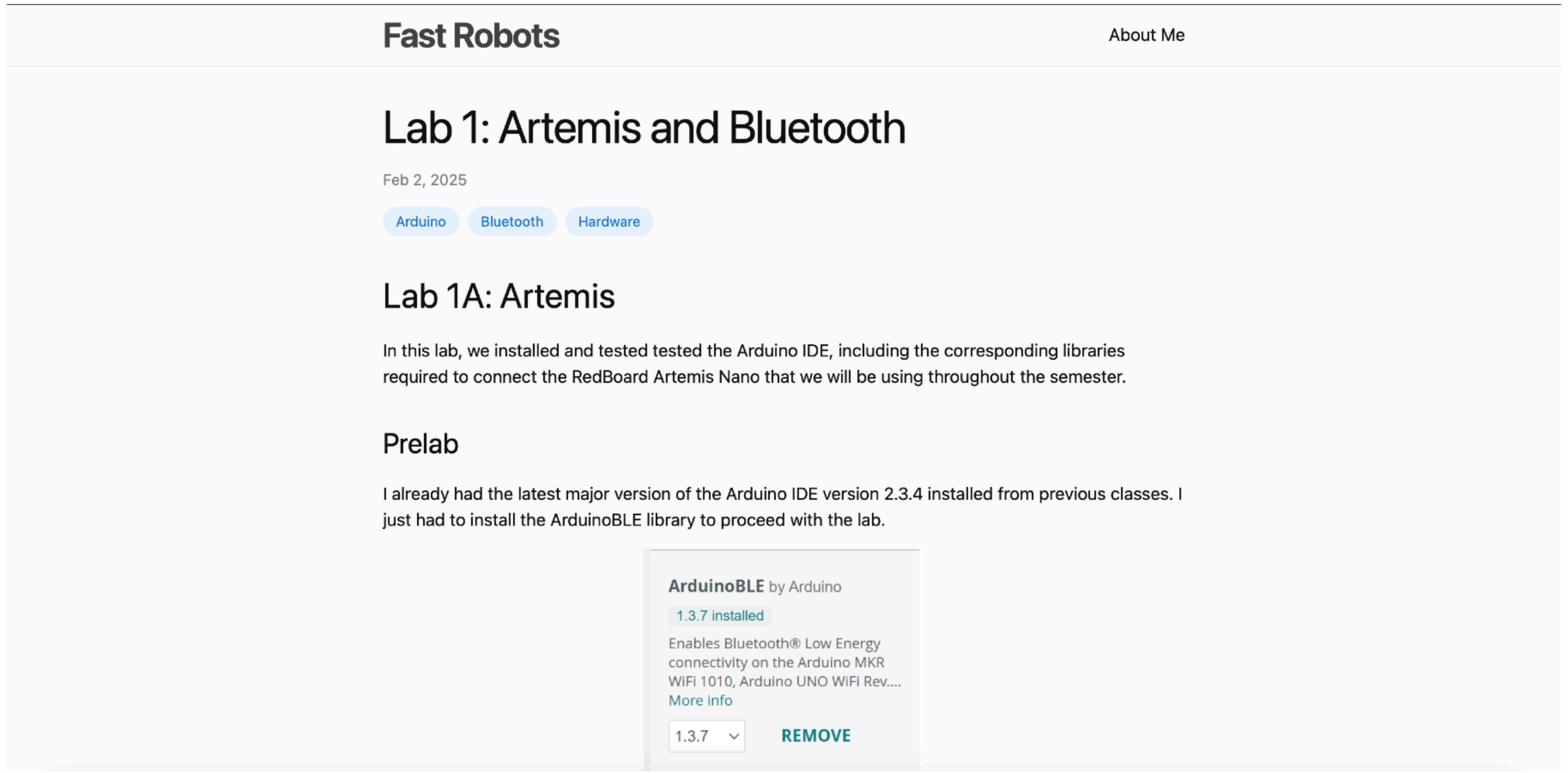
Lab 1: Artemis and Bluetooth

Data Types

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

Lab 1: Artemis and Bluetooth

Summary



- Example write up from last year
    - Comprehensive,
    - Concise,
    - Visually Appealing
- **Important note:**
    - Lab 1 is split over two weeks and is very easy to allow for students to move into the class during Add/Drop
    - The next labs are **significantly more** time consuming, start early.

Class Action Items

- If you want to drop the class, please let me know **ASAP** and return your lab kits!
- For students who are moving off of the waitlist, I will schedule open hours this weekend for you to pick up your kit!
- **January 30th, midnight**: Make a GitHub repository and build your Github page
    - Include: name, photo, a small introduction, and the class number
    - Share **the page link** in the canvas assignment

See you next week!