Robotics 311 : How to build robots and make them move

Prof. Elliott Rouse

GSI Yves Nazon MS

Fall 2022

Photo credit: Brenda Ahearn / U-M CoE

ROB 311 – Lecture 14

- Today:
  - Review torque conversion
  - Discuss kinematic conversions
  - Finish lab exercises (second half+)

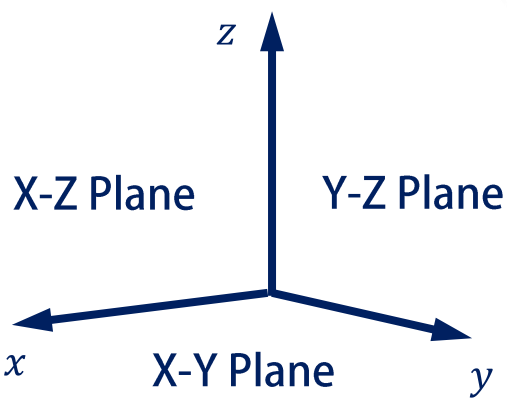
- Announcements
  - HW 3 posted, due 10/20 at class start
  - Stay tuned for HW / quiz solutions
  - No class Tuesday (10/18)! Fall study break
  - Midterm exam – 11/8

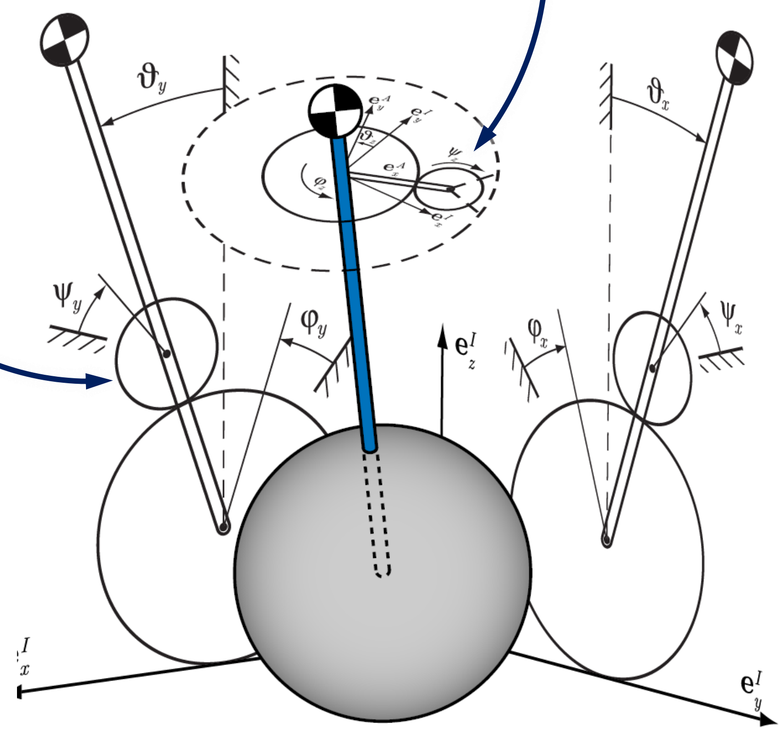
Full Planar Model

The X-Z and Y-Z planes have a single wheel beneath the center of mass

The X-Y plane contains a single wheel that rotates the ball-bot

ϑ_{axis}	Body rotation
ψ_{axis}	Wheel rotation
φ_{axis}	Ball rotation

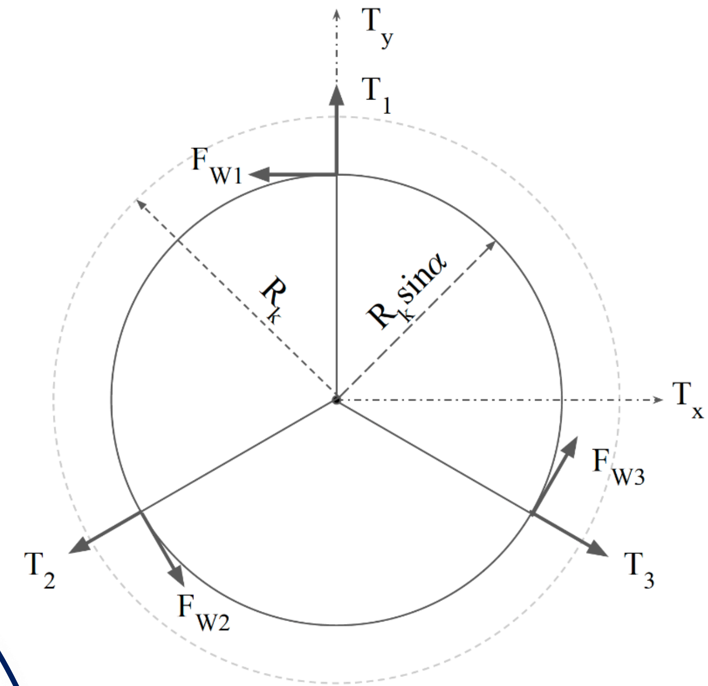




Since we want to balance and drive, the rotation of the ball (i.e. X-Y plane) is mostly irrelevant

- Each plane has two DOFs: Rotations of the chassis and ball
- X-Z and Y-Z planes are required for balance and motion—we will focus on these
- The torques and motion of the virtual wheels are different from the real wheels

Virtual and Real Wheel Contact



- Torque on the wheel - cross product

$$T_{Wi} = r_{Wi} \times F_{Wi} \;\text{ for }\; i = 1,2,3$$

$$T_{Wj} = r_{Wj} \times F_{Wj} \;\text{ for }\; j = x, y, z$$

- Virtual wheel contact points in (X-Z and Y-Z planes)

$$r_{Wx} = R_k \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad r_{Wy} = R_k \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad r_{kWz} = R_k \begin{bmatrix} 0 \\ \sin(\alpha) \\ 0 \end{bmatrix}$$

We begin with contact points then discuss forces

- Real wheel contact points

$$r_{W1} = R_k \begin{bmatrix} 0 \\ \sin(\alpha) \\ \cos(\alpha) \end{bmatrix} \qquad r_{W2} = R_k \begin{bmatrix} \frac{-\sqrt{3}}{2}\sin(\alpha) \\ -\frac{1}{2}\sin(\alpha) \\ \cos(\alpha) \end{bmatrix} \qquad r_{W3} = R_k \begin{bmatrix} \frac{\sqrt{3}}{2}\sin(\alpha) \\ -\frac{1}{2}\sin(\alpha) \\ \cos(\alpha) \end{bmatrix}$$

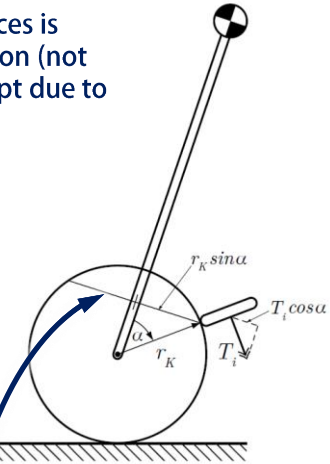
Virtual and Real Forces

- Torque on the wheel - cross product

$$T_{Wi} = r_{Wi} \times F_{Wi} \;\text{ for }\; i = 1,2,3$$

$$T_{Wj} = r_{Wj} \times F_{Wj} \;\text{ for }\; j = x, y, z$$

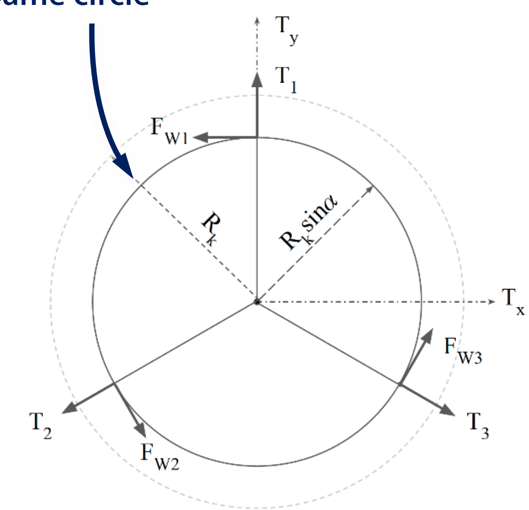
The direction of the forces is along the vector direction (not the component subscript due to the cross product)



- Virtual wheel forces

$$F_{Wx} = \frac{T_x}{r_w}\begin{bmatrix}0\\1\\0\end{bmatrix} \quad F_{Wy} = \frac{T_y}{r_w}\begin{bmatrix}1\\0\\0\end{bmatrix} \quad F_{Wz} = \frac{T_z}{r_w}\begin{bmatrix}-1\\0\\0\end{bmatrix}$$

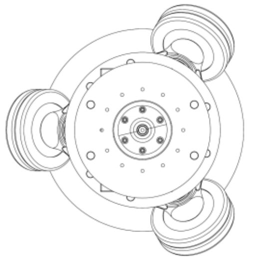
Same circle



- Real wheel forces

$$F_{W1} = \frac{T_1}{r_w}\begin{bmatrix}-1\\0\\0\end{bmatrix} \quad F_{W2} = \frac{T_2}{r_w}\begin{bmatrix}1/2\\-\sqrt{3}/2\\0\end{bmatrix} \quad F_{W3} = \frac{T_3}{r_w}\begin{bmatrix}1/2\\\sqrt{3}/2\\0\end{bmatrix}$$

Relating Virtual and Real Torques



- Torque in both coordinate systems is conserved
  - Not forces conserved—this says the torques around the ball will be equivalent

$$T_1 + T_2 + T_3 = T_x + T_y + T_z$$

- Written in terms of force and perpendicular distance

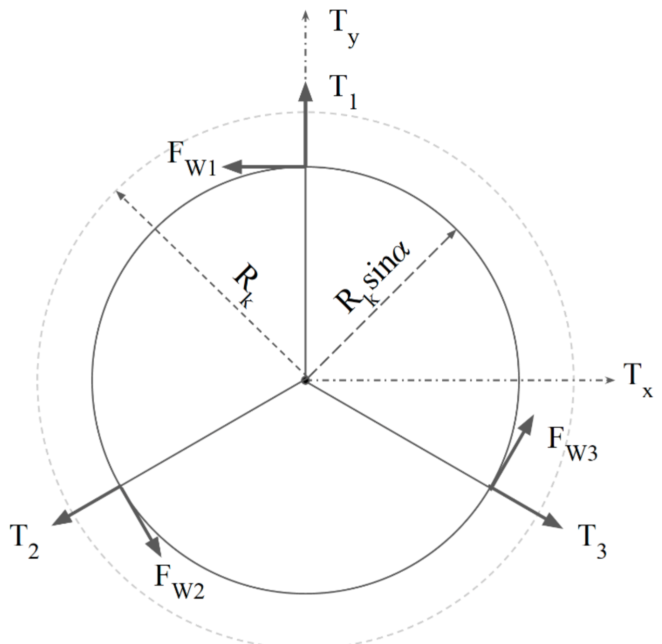
$$r_{W1} \times F_{W1} + r_{W2} \times F_{W2} + r_{W3} \times F_{W3} \ldots$$
$$= r_{Wx} \times F_{Wx} + r_{Wy} \times F_{Wy} + r_{Wz} \times F_{Wz}$$

- Solving for torques

$$T_1 = \frac{1}{3}\left(T_z - \frac{2T_y}{\cos(\alpha)}\right)$$

$$T_2 = \frac{1}{3}\left(T_z + \frac{1}{\cos(\alpha)}\left(-\sqrt{3}T_x + T_y\right)\right)$$

$$T_3 = \frac{1}{3}\left(T_z + \frac{1}{\cos(\alpha)}\left(\sqrt{3}T_x + T_y\right)\right)$$



Determined by solving conservation of torque using eqns. from previous slides

Relating Virtual and Real Torques

α $\pi/4$ or 45°

β $\pi/2$ or 90°

- Conversion equations provided in ETH Zurich dissertation
- Solved for real motor torques

These are the same equations as the previous slide, but written generally for any α and β

$$T_1 = \frac{1}{3} \cdot \left(T_z + \frac{2}{\cos\alpha} \cdot (T_x \cdot \cos\beta - T_y \cdot \sin\beta) \right)$$

$$T_2 = \frac{1}{3} \cdot \left(T_z + \frac{1}{\cos\alpha} \cdot \left(\sin\beta \cdot (-\sqrt{3}T_x + T_y) - \cos\beta \cdot (T_x + \sqrt{3}T_y) \right) \right)$$

$$T_3 = \frac{1}{3} \cdot \left(T_z + \frac{1}{\cos\alpha} \cdot \left(\sin\beta \cdot (\sqrt{3}T_x + T_y) + \cos\beta \cdot (-T_x + \sqrt{3}T_y) \right) \right)$$

- Solved for virtual motor torques

$$T_x = \cos\alpha \cdot \left(T_1 \cdot \cos\beta - T_2 \cdot \sin(\beta + \frac{\pi}{6}) + T_3 \cdot \sin(\beta - \frac{\pi}{6}) \right)$$

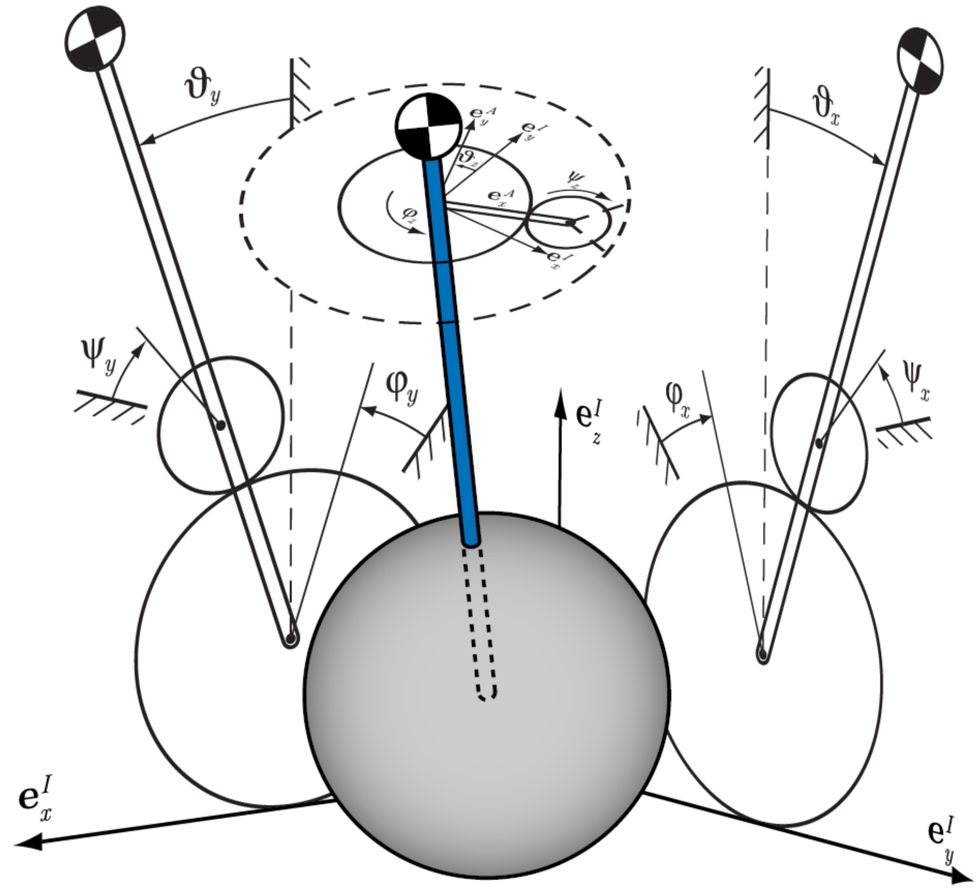
$$T_y = \cos\alpha \cdot \left(-T_1 \cdot \sin\beta - T_2 \cdot \cos(\beta + \frac{\pi}{6}) + T_3 \cdot \cos(\beta - \frac{\pi}{6}) \right)$$

$$T_z = T_1 + T_2 + T_3$$

Fankhauser and Gwerder et al, 2010

Ball-Bot Kinematics

- We now can describe torque from the wheels to full planar model
  - Go from x and y components to the 3 motor torques
- Can we develop a similar analysis for velocity transformation?
- The real and virtual wheels are the same size
- Do the real wheels and virtual wheels have the same velocity?
  - No, wheel orientation affects velocity
  - Wheels only spin with the component of linear velocity perpendicular to wheel axis



Ball-Bot Kinematics

- Planar model has two DOFs - φ_{axis} and ϑ_{axis}
- Lets look at the velocities of points A, B, and C

$v_{A-Ccomponent}$	Velocity component of point A, B, or C
φ_{axis}	Ball angle
ϑ_{axis}	Body angle
ψ_{axis}	Wheel angle
y_k	Linear position of ball center
	Time derivative denoted by dot operator

$$v_{Ay} = 0$$

$$v_{Az} = 0$$

$$v_{By} = \dot{y}_k + \dot{\varphi}_x R_k \cos(\vartheta_x)$$

$$v_{Bz} = -\dot{\varphi}_x R_k \sin(\vartheta_x)$$

$$v_{Cy} = \dot{y}_k + \big((R_k + R_w)\dot{\vartheta}_x + \dot{\psi}_x R_w\big)\cos(\vartheta_x)$$

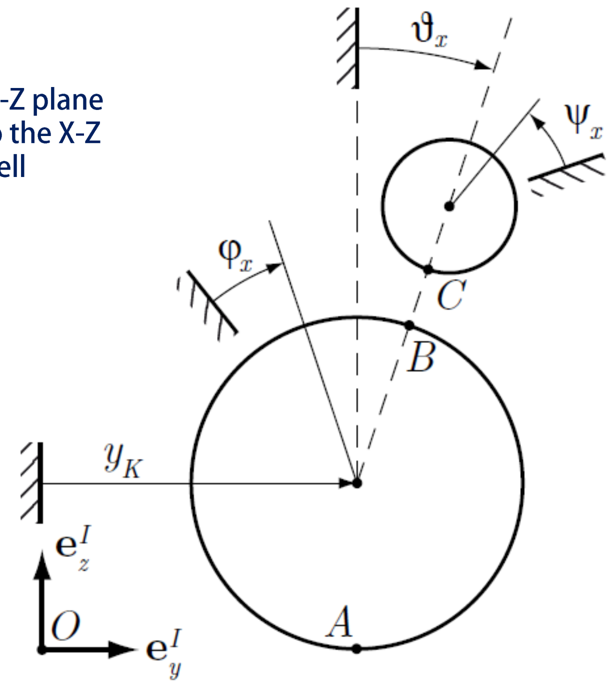
$$v_{Cz} = -\big((R_k + R_w)\dot{\vartheta}_x + \dot{\psi}_x R_w\big)\sin(\vartheta_x)$$

$$v_{By} = v_{Cy}$$

$$v_{Bz} = v_{Cz}$$

Y-Z Plane

Derived for the Y-Z plane but applicable to the X-Z plane as well



Ball-Bot Kinematics

$v_{A-Ccomponent}$	Velocity component of point A, B, or C
φ_{axis}	Ball angle
ϑ_{axis}	Body angle
ψ_{axis}	Wheel angle
y_k	Linear position of ball center
	Time derivative denoted by dot operator

- We can set the velocities equal at points A and B

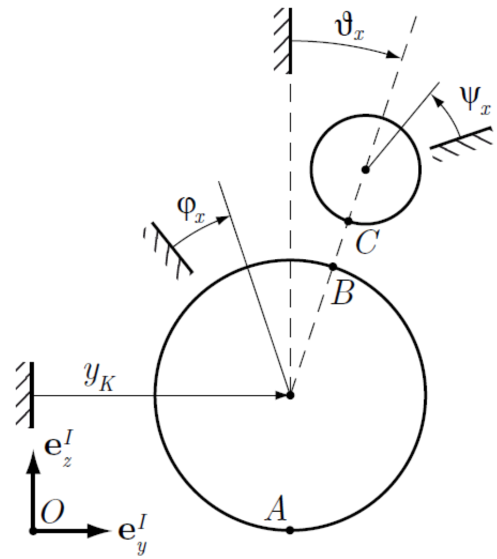
$$v_{By} = v_{Cy}$$

$$v_{Bz} = v_{Cz}$$

$$v_y = \cancel{\dot{y}_k} + \dot{\varphi}_x R_k \cancel{\cos}(\vartheta_x) = \cancel{\dot{y}_k} + \big((R_k + R_w)\dot{\vartheta}_x + \dot{\psi}_x R_w\big)\cancel{\cos}(\vartheta_x)$$

$$v_z = -\dot{\varphi}_x R_k \cancel{\sin}(\vartheta_x) = -\big((R_k + R_w)\dot{\vartheta}_x + \dot{\psi}_x R_w\big)\cancel{\sin}(\vartheta_x)$$

$$\dot{\varphi}_x R_k = (R_k + R_w)\dot{\vartheta}_x + \dot{\psi}_x R_w$$



- We want to know how fast the wheels need to spin for known / induced ball and body rotations

Ball-Bot Kinematics

- Planar model has two DOFs - φ_{axis} and ϑ_{axis}
- Lets look at the velocities of points A, B, and C

$v_{A-Ccomponent}$	Velocity component of point A, B, or C
φ_{axis}	Ball angle
ϑ_{axis}	Body angle
ψ_{axis}	Wheel angle
y_k	Linear position of ball center
	Time derivative denoted by dot operator

$$\dot{\varphi}_x R_k = (R_k + R_w)\dot{\vartheta}_x + \dot{\psi}_x R_w$$

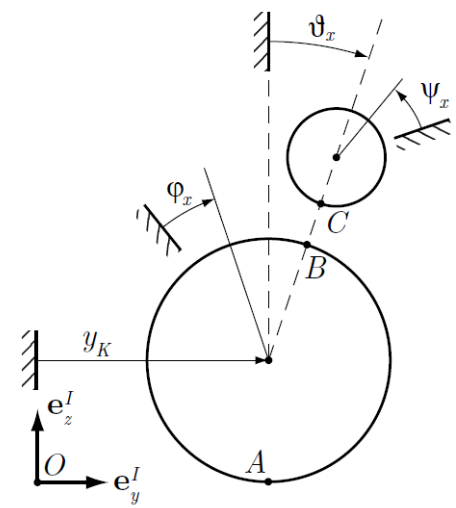
$$\dot{\psi}_x = \frac{R_k}{R_w}\dot{\varphi}_x - \frac{R_k + R_w}{R_w}\dot{\vartheta}_x$$

$$\dot{\psi}_x = \frac{R_k}{R_w}(\dot{\varphi}_x - \dot{\vartheta}_x) - \dot{\vartheta}_x$$

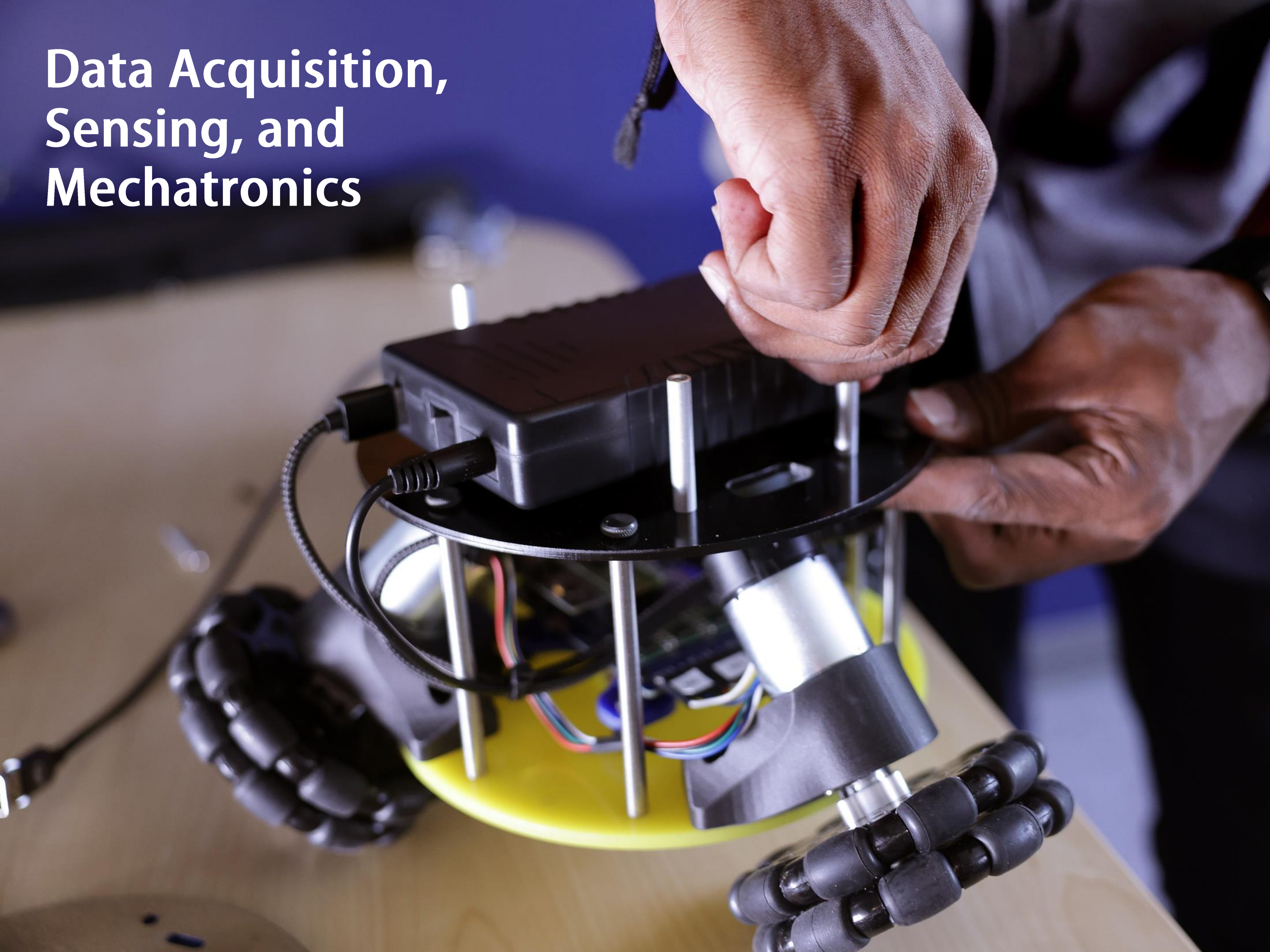
Virtual wheel angular velocity

Ball angular velocity

Body angular velocity



Data Acquisition, Sensing, and Mechatronics



Data Acquisition

- A key aspect of developing a robotic system is sensing the robot state / environment
- For the ball bot, we will want to know
  - Lean angle
  - Ball rotation (obtained from motor rotation)
- In the coming slides, we will learn how use sensors for data acquisition
- Two types of data acquisition
  - Analog: continuous voltages read by a computer and interpreted from the voltage value
  - Digital: information is sent over a communication bus in the form of 1s and 0s that contain the sensor data that has been digitized
- Our ball-bot design only uses digital sensors but we will quickly review analog sensing
  - Previously more common but becoming less common

Data Acquisition

- Typically, sensors output voltage, which needs to be acquired by a digital computer for analysis and control
- Accomplished using a data acquisition system

Physical system / process → Sensor transducer → Signal conditioning → Analog to digital (A2D) converter → Computer / RPi

- Analog to digital (A2D) converter: digitizes voltage to be read by a computer
- Three key attributes
  - Bits – describes how many 'bins' can the voltage be separated into
  - Sample rate (Hz) – how fast the loop runs / A2D converter is sampled
  - Input range (V) – the total range of voltages able to be sampled

Resolution → $$\frac{\Delta V}{bit} = \frac{V_{range}}{2^{bits} - 1}$$

Analog Data Acquisition

- To calculate the voltage, we sample the A2D system, which will return an integer number of bits
- These bits need to be converted back to voltage
- After converting to volts, it needs to be converted to the original units
  - This is accomplished using the sensors calibration curve or ‘sensitivity’

$$V_{sensor} = V_{min} + \left(b \cdot \frac{V_{range}}{2^{bits} - 1} \right)$$

Number of bits returned from your A2D card

- Example – your 16 bit DAQ system records a value of 11564, with an input range of -10 to 10 V.
- What is the voltage?

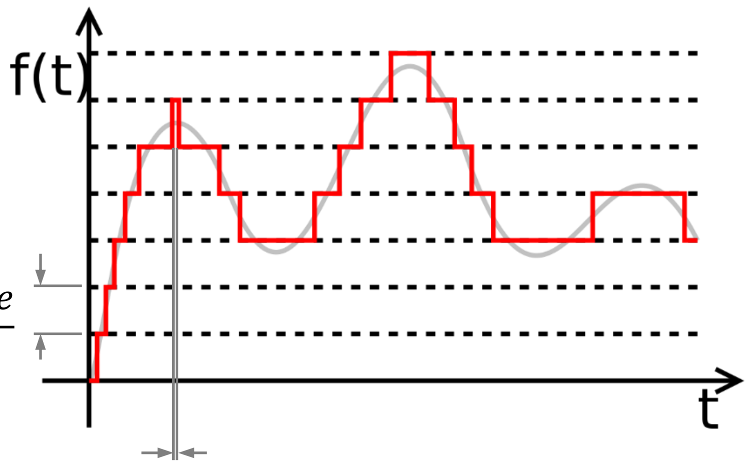
$$V = -10 + \left(11564 \cdot \frac{20}{65535} \right) = -6.47\ V$$

USB data acquisition card

Analog Data Acquisition

- Signals are sampled periodically, at a frequency governed by the sample rate
- This causes some 'quantization' of the analog signal
- The sample rate is a key factor of a data acquisition system
- It governs the frequency content of what can be measured
- Higher sample rates can sense higher frequencies
- Nyquist criterion:

$$\frac{input\ range}{2^{bits}-1}$$



Maximum measurable frequency →

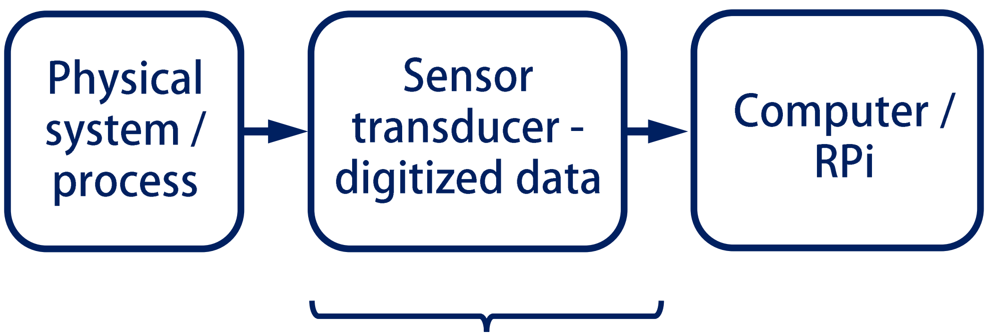
$$f_{max} = \frac{1}{2} F_s$$

$$\Delta t = \frac{1}{F_s} = \frac{1}{sample\ rate}$$

- The highest frequency that can be measured is half the sample rate
- Sample rate / loop rate must be set carefully to make sure the relevant frequencies can be recorded

Digital Data Acquisition

- Digital sensors are becoming increasingly common
- Data are transmitted using digital communication (USB, I^2C, etc.)
- No analog voltage – concepts of resolution change for these sensors
    - Governed directly from the sensor, not the A2D card
- Nyquist frequency still applies
- Sampling still applies



Digital sensors communicate directly with acquisition computer

Digital Data Acquisition

- Digital sensors are defined by their bit-level resolution
  - For example, 14-bit encoder (rotation sensor)

$$\frac{°}{bit} = \frac{360°}{2^{14} - 1} = 0.0220$$

- Digital communication busses
  - USB
  - Inter-Integrated Circuit communication (I^2C)
  - Serial Peripheral Interface (SPI)
  - RS-232 (old school, serial port)

$\left.\begin{array}{l} I^2C \\ SPI \end{array}\right\}$ Most common (for now)

- A program is used to request and interpret information from the sensor
- Information on communication and interpretation is provided in the sensor datasheet
- This program is called a 'driver' or 'API' and is usually written in C or Python

ROB 311 – Lab 7

- There is some spread in the ball-bot readiness between teams (this is okay)
- We will break here to allow some of the class to finish Lab 7
- Git
  - Revisit forking repositories
  - Authentication
  - Saving data
- Python
  - Looping and indexing
  - Importing packages
  - Creating functions
  - Saving data