Robotics!

Jennefer Maldonado

Agenda

1. Introduction
2. Use Cases
3. Raspberry Pi
4. Python
5. Putting it All Together!

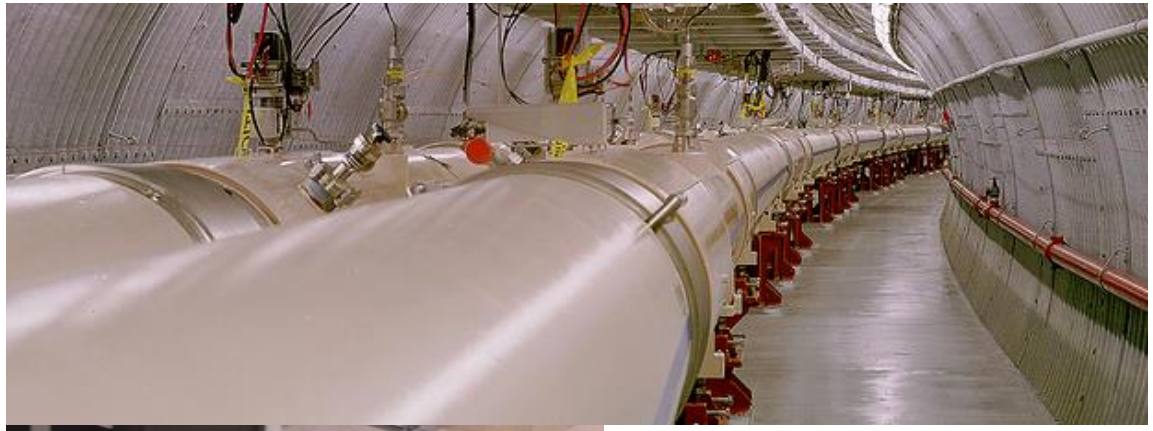
Introduction

- Robotics is interdisciplinary and can be applied to many projects
- Hardware
  - Electrical Engineering
- Software
- Domain Science

Use Cases

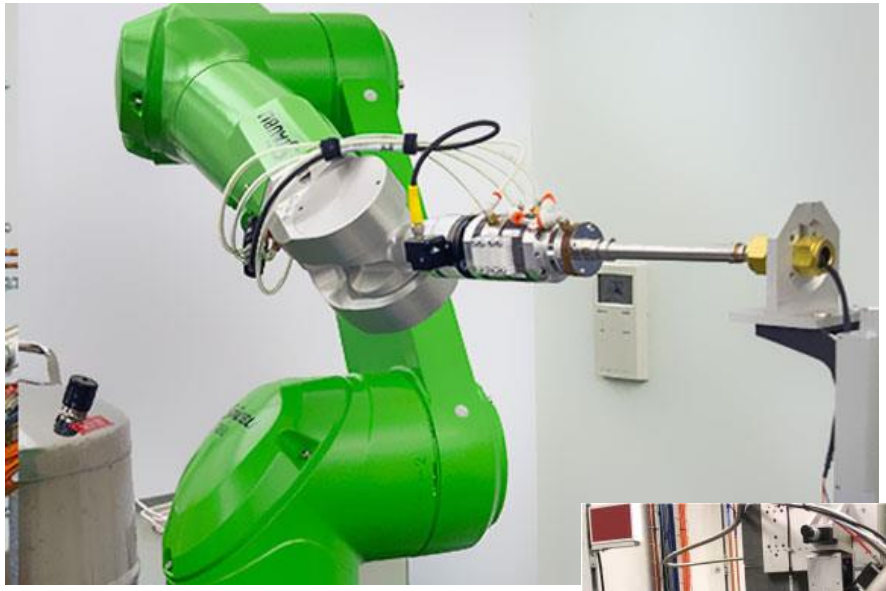
Relativistic Heavy Ion Collider

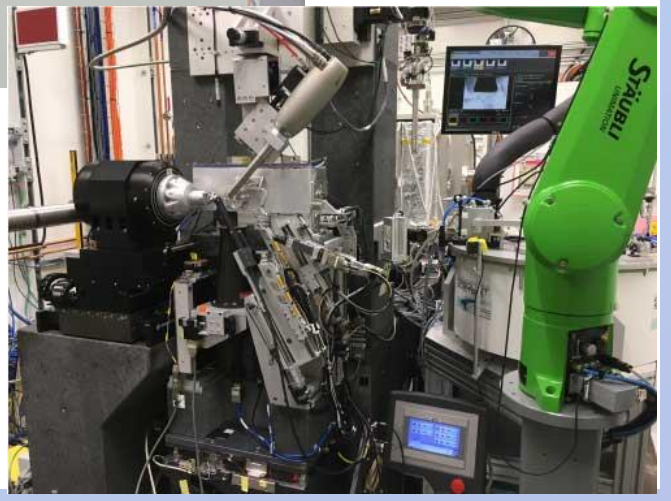
- RHIC's mission is to collide heavy ions and determine what makes up protons and neutrons
- Anytime you accelerate a charged particle it emits radiation
- C-AD wants to limit the amount of radiation people are exposed to
- Robots for measurements, sweeps, installations etc.



National Synchrotron Light Source II

- Beamlines at NSLS-II need to change samples often
- Needed a quick and efficient way to do so
- Robot arms grab samples and exchange them to speed up data collection



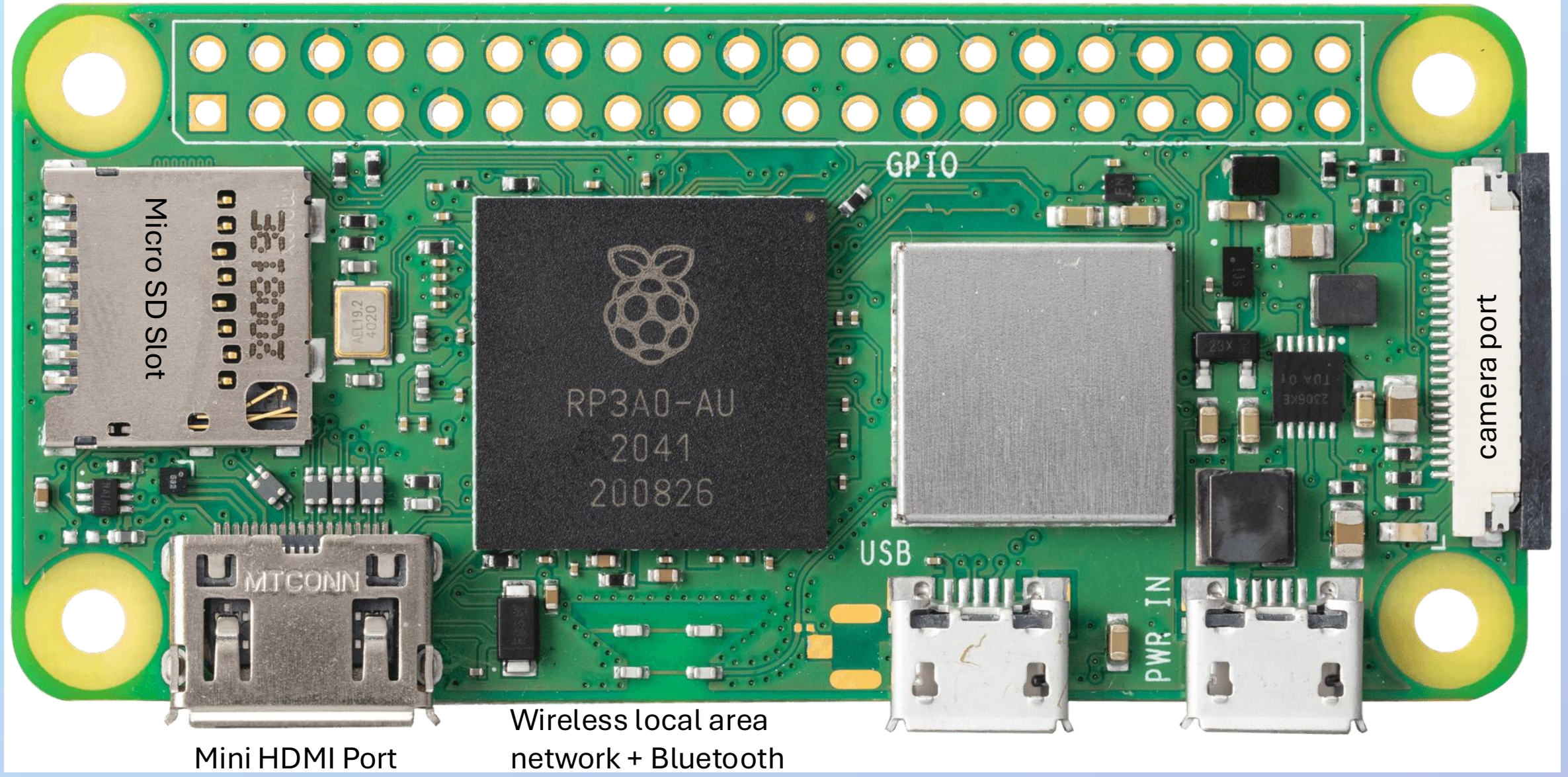


Raspberry Pi

Raspberry Pi: An Introduction

- Small single board computers
- Intended to be used for teaching basic computer science
- Became way more popular than expected
- Used for robotics, home and industry automation, and hobbies

general-purpose input/output



Mini HDMI Port

Wireless local area network + Bluetooth


wheels
battery box
breadboard
“third wheel”
direct current motors
resistors
motor controller board
distance sensor
line follower sensor
jumper cables

Important Terms

Breadboard

- Prototype board for electronic circuits
- Doesn’t require soldering and are easily reusable

Resistors

- Two terminal electrical component that implements resistance as a circuit element
- Are used to reduce current flow, adjust signal levels, and divide voltages

Distance Sensor

- devices that generate or sense ultrasound energy
- In our robots the two “speakers” emit a high-frequency sound
- this bounces off any nearby solid objects
- Listens for return echo and calculates the distance between the two

Python

- Python is a programming language used complete data analysis, automate repetitive tasks, and build software & websites
- On our example we will use python to tell the robot cars to move!

Important Terms

- **IP Address:** number assigned to a device connected to a computer network that uses Internet Protocol for communication
- **SSH:** network protocol for operating network services securely
- Allows for remote login and command-line execution

Getting Started

Instructions: part 1

1. Plug in portable chargers and switch battery packs to “on”
2. ssh pipi@<ip address>
3. password is picar
4. cd EduKit3-master/‘CamJam EduKit 3- GPIO Zero’/Code

Each Pi has a custom IP Address, please note the address you are using. It is located on the paper handout

Hello World!

- Run python 1-helloworld.py

What do you see?

Let’s test those motors!

```
# CamJam EduKit 3 - Robotics


import time  # Import the Time library
from gpiozero import CamJamKitRobot  # Import the CamJam GPIO Zero Library

robot = CamJamKitRobot()

# Turn the motors on
robot.forward()
time.sleep(1)  # Pause for 1 second

robot.backward()
# Turn the motors off
robot.stop()
```

Make sure your robot is in a good location not blocked by anything.
How do we run this code?

python 3-motors.py

Let's Turn Around

```
import time  # Import the Time library
from gpiozero import CamJamKitRobot  # Import the CamJam GPIO Zero Library

robot = CamJamKitRobot()

robot.forward()
time.sleep(1)  # Pause for 1 second

robot.backward()
time.sleep(1)

robot.left()
time.sleep(0.5)  # Pause for half a second

robot.right()
time.sleep(0.5)

robot.stop()
```

What would happen if we remove the time.sleep command?
Try it!

Speeding Up & Slowing Down

```
# CamJam EduKit 3 - Robotics
# Worksheet 7 - Controlling the motors with PWM

import time  # Import the Time library
from gpiozero import CamJamKitRobot  # Import the CamJam GPIO Zero Library

robot = CamJamKitRobot()

# Set the relative speeds of the two motors, between 0.0 and 1.0
motorspeed = 0.3

motorforward = (motorspeed, motorspeed)
motorbackward = (-motorspeed, -motorspeed)
motorleft = (motorspeed, 0)
motorright = (0, motorspeed)

robot.value = motorforward
time.sleep(1)

robot.value = motorbackward
time.sleep(1)  # Pause for 1 second

robot.value = motorleft
time.sleep(1)  # Pause for 1 second

robot.value = motorright
time.sleep(1)  # Pause for 1 second

robot.stop()
```

Change the *motorspeed* to a few different values.
What is a good speed to keep the robot at?

Does your robot move in a straight line?

```
# CamJam EduKit 3 - Robotics
# Worksheet 7 - Controlling the motors with PWM

from gpiozero import CamJamKitRobot  # Import the CamJam GPIO Zero Library
import time  # Import the Time library

robot = CamJamKitRobot()

# Set the relative speeds of the two motors, between 0.0 and 1.0
leftmotorspeed = 0.5
rightmotorspeed = 0.5

motorforward = (leftmotorspeed, rightmotorspeed)
motorbackward = (-leftmotorspeed, -rightmotorspeed)
motorleft = (leftmotorspeed, 0)
motorright = (0, rightmotorspeed)

robot.value = motorforward
time.sleep(1)

robot.value = motorbackward
time.sleep(1)  # Pause for 1 second

robot.value = motorleft
time.sleep(1)  # Pause for 1 second

robot.value = motorright
time.sleep(1)  # Pause for 1 second

robot.stop()
```

Vary the value of the left and right motor speed until the robot moves in a straight path

Tip: Try adding more forward commands to watch the robot move in a straight (or not) line

What happens if you remove the time.sleep commands? Can you make the robot turn 90 degrees? (either left or right)

Avoiding Obstacles

- In the file, 9-avoidance.py there is code to activate the distance sensor and have the robot avoid obstacles
- Run this first and then change the distance variables
  - *hownear* – how close the robot can get to items
  - *reversetime* – how far back should the robot move
  - *turntime* – how long should the robot turn for to move away from the obstacle

Avoiding Obstacles

- If the robot does not turn properly try to adjust the left and right motor speed
- Think about the direction each wheel needs to turn to make the robot turn left and right. Do the wheels turn the same way?
- **Challenge:** Instead of reversing and going right, make your robot turn left, measure the distance to the next object then turn right and do the same. Drive in the direction of the obstacle furthest away

Web Control

- python3 webbot.py –i <ip address>
- in the web browser go to <ip address>:5000

Special Thanks To...

- Kyle Kulmatycski
- Jennipher Day
- Daniel Wynne
- Mitch Schneider
- Pete Zimmerman