Seneca

Make a Robot

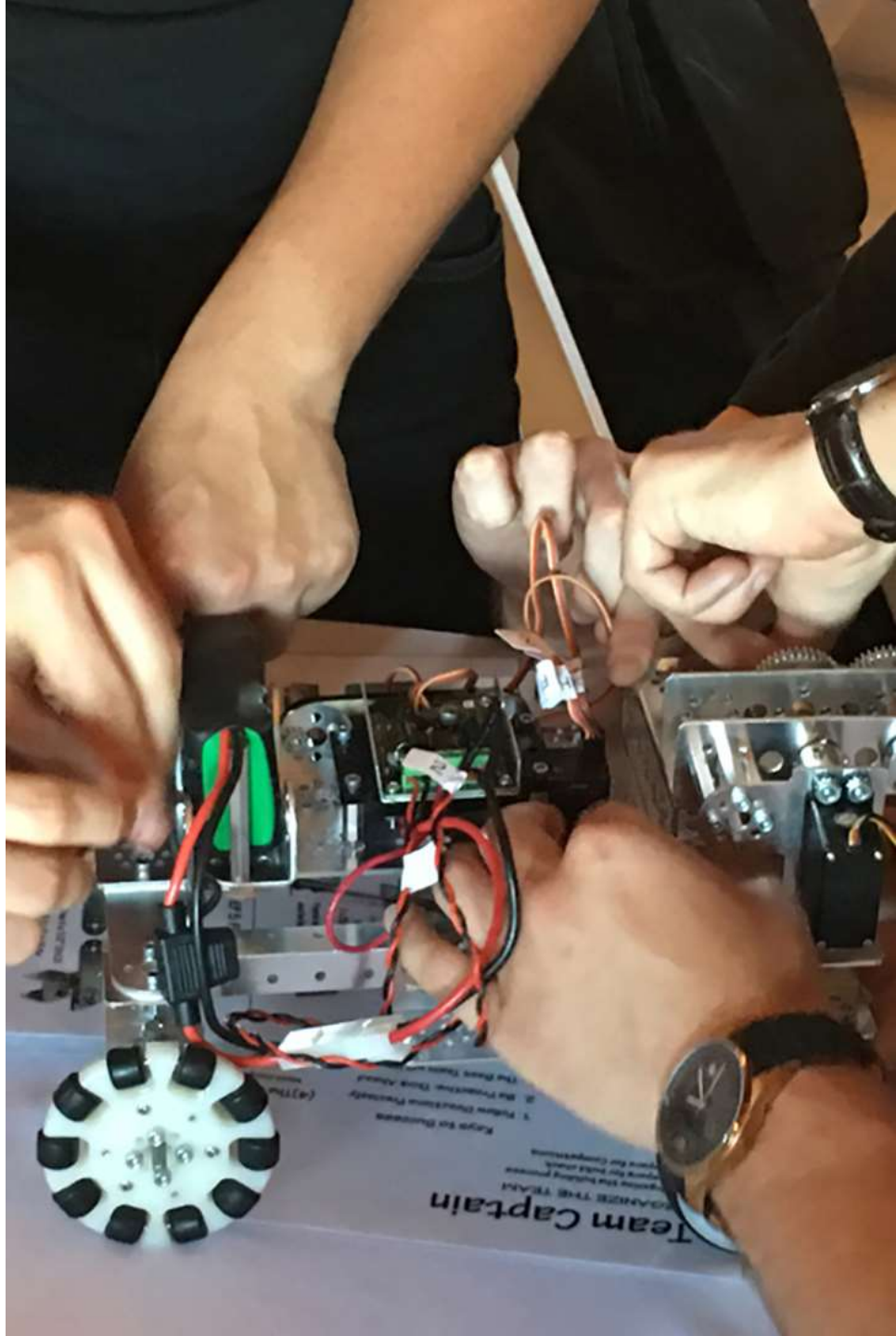
Saeid Khosravani

Summer 2022

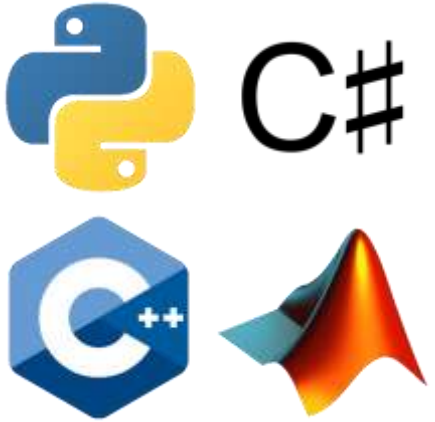
Bringing it all together

Now that you know about all the different types of robots, components to build them with, platforms to base them on, and the best operating system to start programming with, it’s time to build!

The type of robot you build is up to you and can change with your needs and applications!



Robot Programming Language



- In order for the robot to perform any task, we have to give them commands
- Just like any programming language we write the code giving different parts of the robot command to react to the feedback they get from the sensors
- An example would be moving the motor until sonar sensor senses an object in front of the robot then stop the motors
- Some of the commonly used languages are RobotC, C#, C++, Java, Python, MATLAB

What is ROS?

- Robot Operating System (ROS)
  - The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms (Ros.org)
  - Not an actual operating system…or even a Framework..
  - It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms
  - ROS is similar in some respects to 'robot frameworks', such as Player, YARP, Orocos, CARMEN, Orca, MOOS, and Microsoft Robotics Studio

GOALS

- Sharing and collaboration of packages
- Thin design
- ROS-agnostic libraries
- Language independence
- Easy testing
- Scaling
- (ROS Introduction, 2019)

ROS Languages

- Robot Operating System is mainly developed using 2 languages:
  - C++
  - Python
- Libraries to help work with other languages:
  - rosjava for Java
  - roslibjs or rosnodejs for JavaScript

ROS Libraries

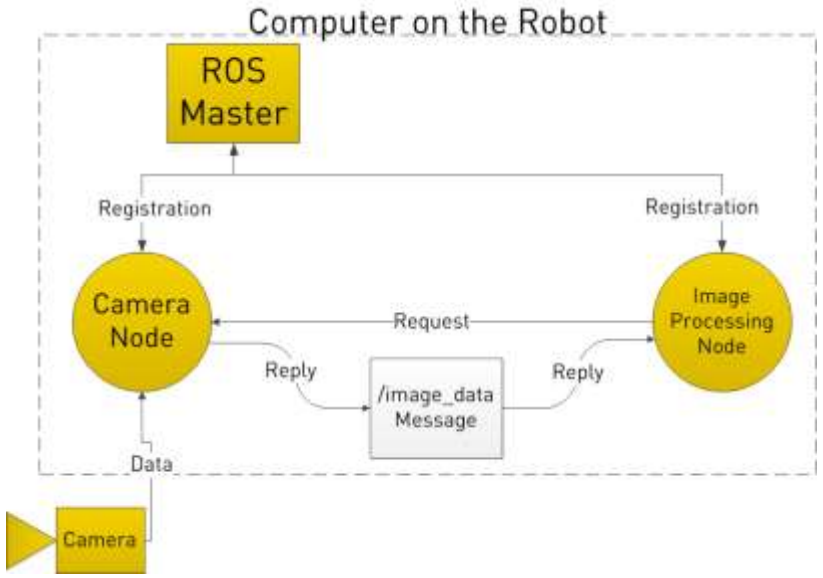
- ROS has more than just core feature and communication tools
- It contains a huge amount of ROS libraries
- Here is a list of some of the libraries:
  - Driver for the motors
  - PID control loop
  - Motion planning
  - 3D visualization
  - Camera driver

ROS Communication

- Topics are mainly used for sending data streams between nodes. For example, if you are monitoring the temperature of a motor on the robot the node monitoring the motor will send a data stream with the temperature as a result other nodes can subscribe to this topic and get the data.
- Services will allow you to create a simple synchronous client/server communication between nodes.
- Actions are based on Topics, they provide you with an asynchronous client/server architecture, where the client can send a request that takes a long time (ex: asking to move the robot to a new location). The client can asynchronously monitor the state of the server and cancel the request anytime.
- To use these communication tools, you will have to use the correct libraries in your code and define specific messages.

Example

- When all the Nodes are registered with the ROS Master then the Camera Node Publish a Topic called /image_data for example. Then the Nodes that are registered should be Subscribed to the Topic /image_data.
- Then the Camera Node receives some data from the Camera, it sends the /image_data message directly to the other nodes via TCP/IP
- Next, we need to implement a Service. A Node can register a specific service with the ROS Master, just as it registers its messages.
- In the below example, the Image Processing Node first requests /image_data, the Camera Node gathers data from the Camera, and then sends the reply.



(Clearpath Robotics, 2015)

When to use ROS

- If your robot has many different sensors and actuators. ROS will help you create a distributed system for all those components.
- If you are a beginner in robotics ROS will allow you to make a robot application quickly and when you are comfortable with ROS, you will be able to quickly write code for many different robots.
- If you are doing research in robotics ROS is perfect for that. When you use ROS, you already have all the code, communication tools, and libraries to build a robot software.
- If you are prototyping a robot, If you know ROS well enough it might only take you a few days to create a prototype and this can save you months of hard work.

ROS Community

- ROS Wiki: You can find tutorials, concept explanations, and guides for different packages.
- ROS Answers: This is where you can ask all your technical oriented questions and if you cannot find something on the Wiki, you will be using this link.
- ROS Discourse: This is a where you can talk about future developments, projects using ROS, and anything related to ROS.
- GitHub: Most of the ROS packages are available on GitHub. You will be able to browse the code and make some contributions.

ROS Components

- Practically, ROS has 5 different facets
  - Enables Modular Software Development
    - Using Reusable code packages
  - Provides a run time environment
    - Supporting near real time communication between system elements and data sharing
  - Programming Standards
    - Not like ISO, HTML, Etc..
      - It provides conventions for using ROS supported code reliably
  - Developer Tools
    - Can build computer simulations for example
  - Vibrant community Worldwide

What does this mean?

Camera

Motion Planning

ROS
Communication
Tools

Hardware Driver

...

Joystick

Hardware Supporting ROS

Arduino Portenta H7

- MCU: Dual-core Arm Cortex-M7 and Cortex-M4
- RAM: 8 MB
- Flash: 16 MB
- Peripherals: USB HS, Ethernet, WiFi/BT...

Arduino Due

- MCU: ARM Cortex-M3 AT91SAM3X8E
- RAM: 96 kB
- Flash: 512 kB

Raspberry Pi Pico RP2040

- MCU: Dual-core Arm Cortex-M0+
- RAM: 264 kB
- Flash: up to 16 MB
- Peripherals: I2C, SPI, PIO...

https://micro.ros.org/docs/overview/hardware/

Sensors - IR

1 D Finders

Infrared linear distance sensor that can be used to make low-cost robots

- https://www.terabee.com/shop/lidar-tof-range-finders/ter

2 D Finders

Sensors that can measure the distance on 2D plane, and is mainly used for navigation

http://wiki.ros.org/hls_lfcd_lds_driver

Sensors - Camera

3 D Finders and Camera

Sensors used in 3D distance measurement such as Intel's RealSense, Microsoft's Kinect, ASUS's Xtion

http://wiki.ros.org/duo3d-driver

https://www.stereolabs.com/docs/ros/

• Camera driver used for object recognition, face recognition, character recognition, etc. and various application packages

References

References:

- https://github.com/ROBOTIS-GIT
  http://wiki.ros.org/