

COMP0204: Introduction to Programming for Robotics and AI

Lecture 10: Robotics Programming and Applications Overview

Guest Lecture by: Dr Narsimlu Kemsaram

MEng Robotics and AI

UCL Computer Science

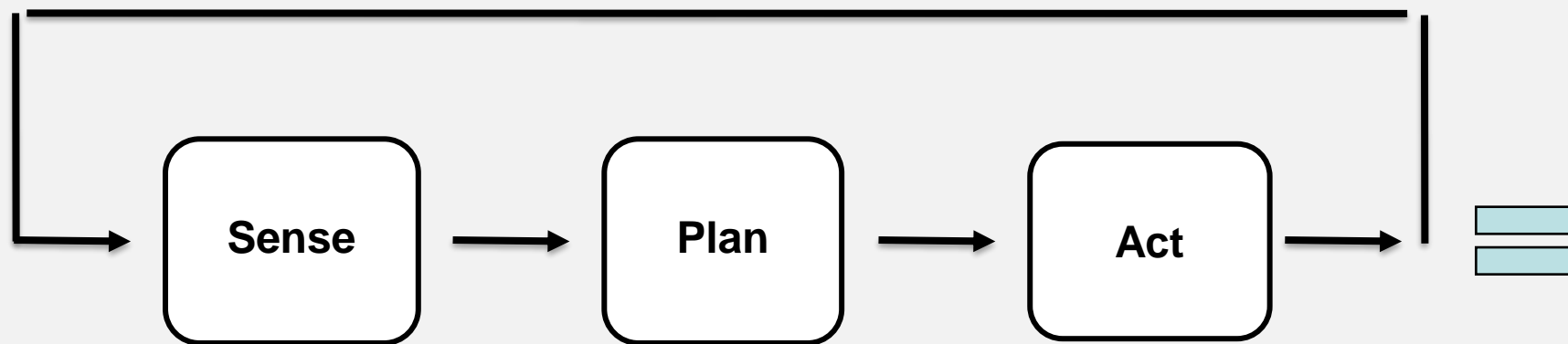
Today's lecture

- Introduction to Robotics
- Robotics in Action – Practical Applications
- Introduction to Arduino IDE
- Robotics Programming with Mona Robot
- Simple Concurrent Programming: Robot Sensing and Control

Introduction to Robotics

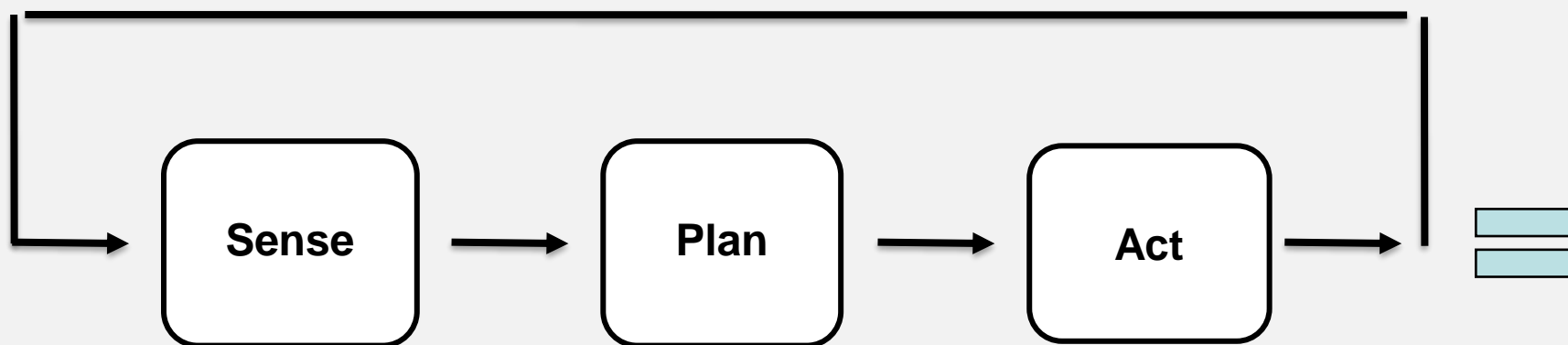
What is a Robot?

- A goal-oriented **machine** that generally can **sense**, **plan** and **act** **autonomously**.



Required Sensors and Algorithms for Robot?

- **Sensors:** Camera, LiDAR, etc.
- **Algorithms:** Mapping, Path Planning, Control, etc.
- **Hardware:** Arduino, Raspberry Pi, Jetson Nano, etc.
- **Software:** ???



Required Software Components for Robotics?

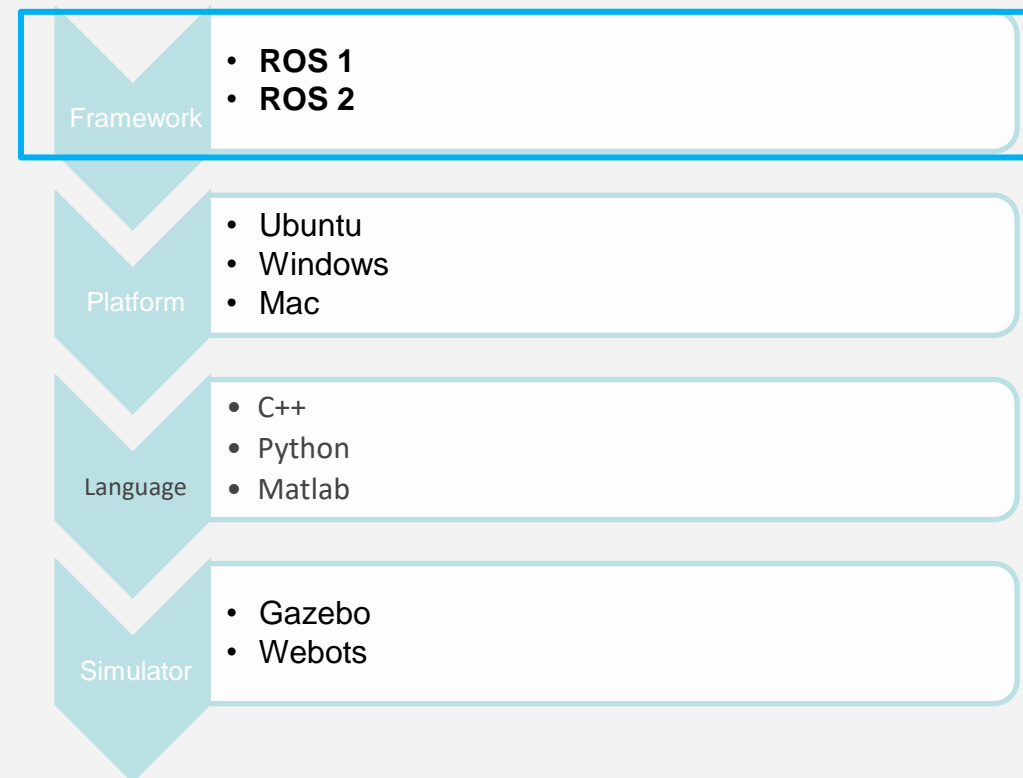
- **Required Software Components for Robotics:**

- **Framework**

- **Platform**

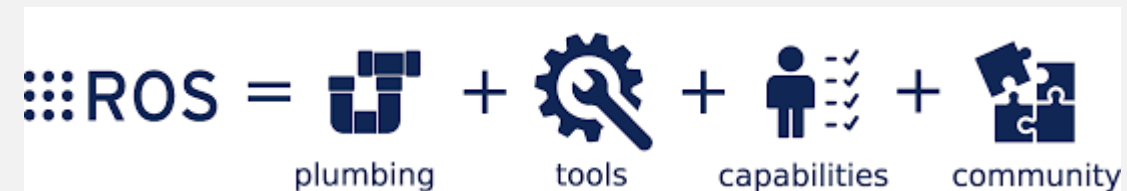
- **Language**

- **Simulator**



What is ROS?

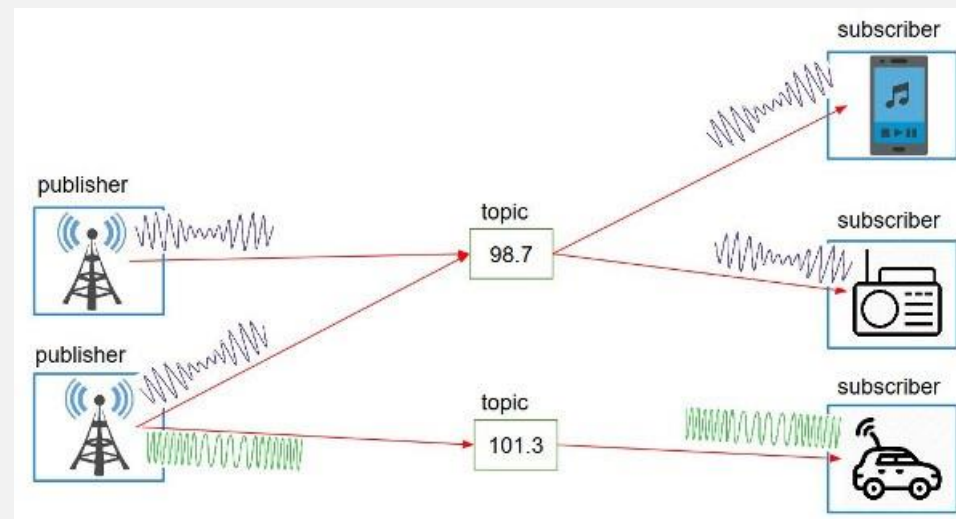
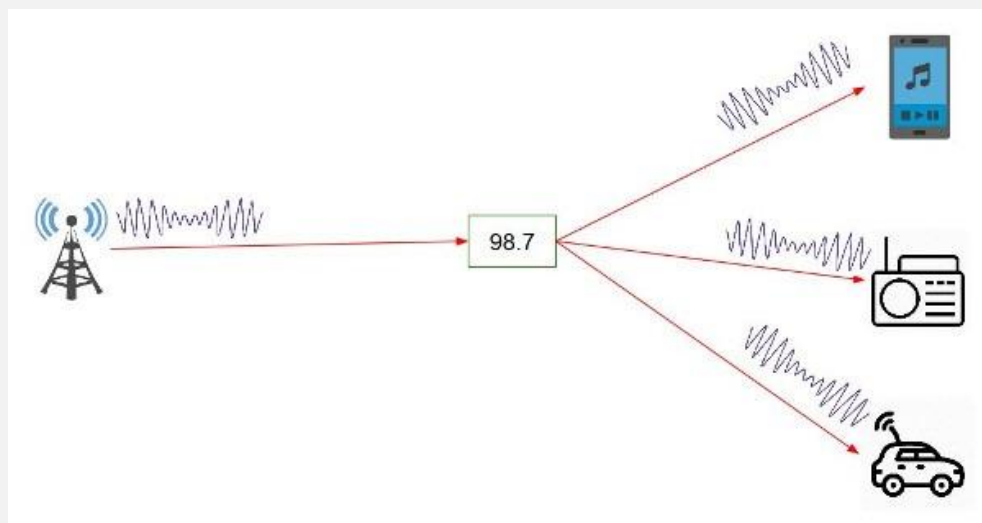
- **ROS = Robot Operating System**
- **ROS** is a collection of **tools**, **software libraries**, and **documentation**.
- The term "**Operating System**" is a misnomer: **ROS** is more like a **framework or middleware** plus a **developer community (eco system)**.



 **ROS.org**

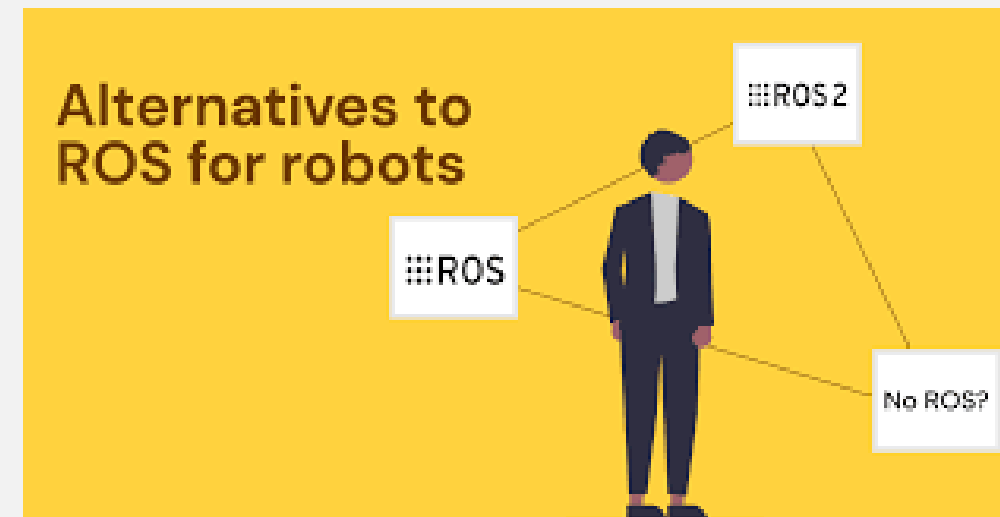
- <https://www.ros.org/>

What is ROS Nodes? ROS Topics?



History of ROS?

- Originally developed in **2007** at the **Stanford Artificial Intelligence Laboratory**.
- Since **2013** managed by **Open-Source Robotics Foundation (OSRF)**.
- Today used by many **Robots, Universities** and **Companies**.
- **De facto Standard** for Robot Programming.



ROS1 to ROS2?

- Since **ROS1** was started in **2007**, a lot has **changed in the robotics** and ROS community.
- In **2020**, **ROS2** was started.
- The goal of the **ROS 2** project is to adapt to these **changes**, leveraging what is great about ROS 1 and **improving** what isn't.



ROS 2 Distribution Releases

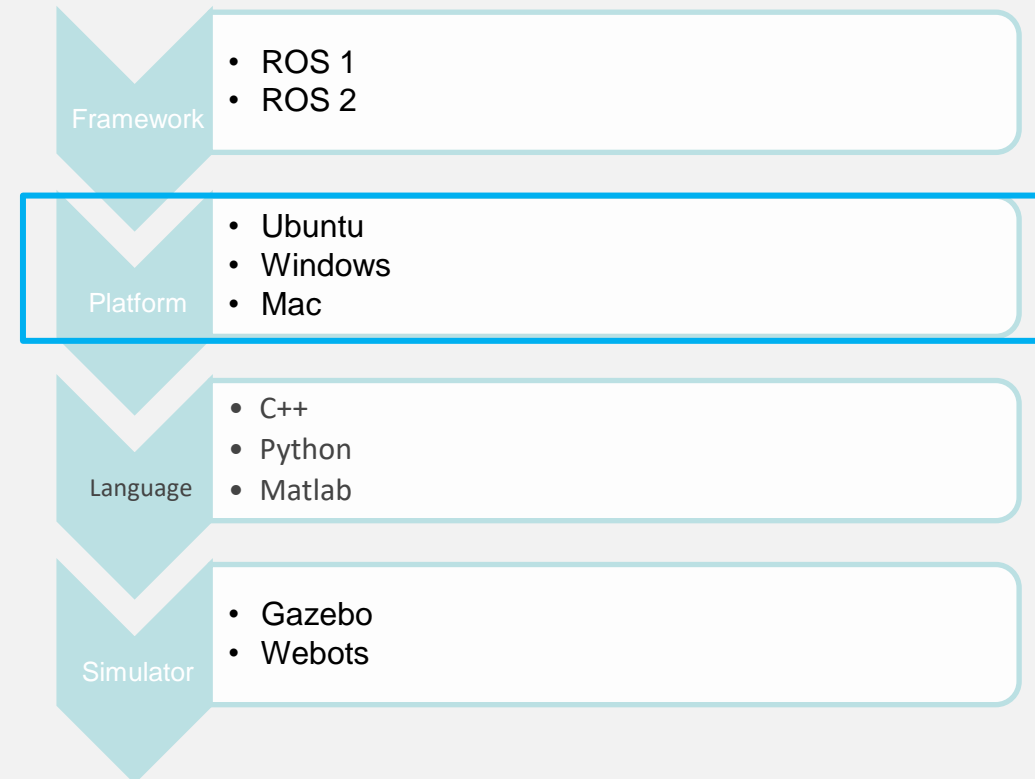
Distribution	Release date	Poster	EOL date	Support duration
Rolling Ridley ^{[94][95]} (rolling release with latest features)	progressing since June 2020		N/A	N/A
Jazzy Jalisco ^[2]	May 2024	t.b.d.	EST. May 2029	5 years
Iron Irwini	23 May 2023 ^[96]		November 2024	1.5 years
Humble Hawksbill	23 May 2022 ^[97]		May 2027	5 years
Galactic Geochelone	23 May 2021 ^[98]		December 2022	1.5 years
Foxy Fitzroy	5 June 2020 ^[99]		June 2023	3 years

Old version
Older version, still maintained
Latest version
Future release

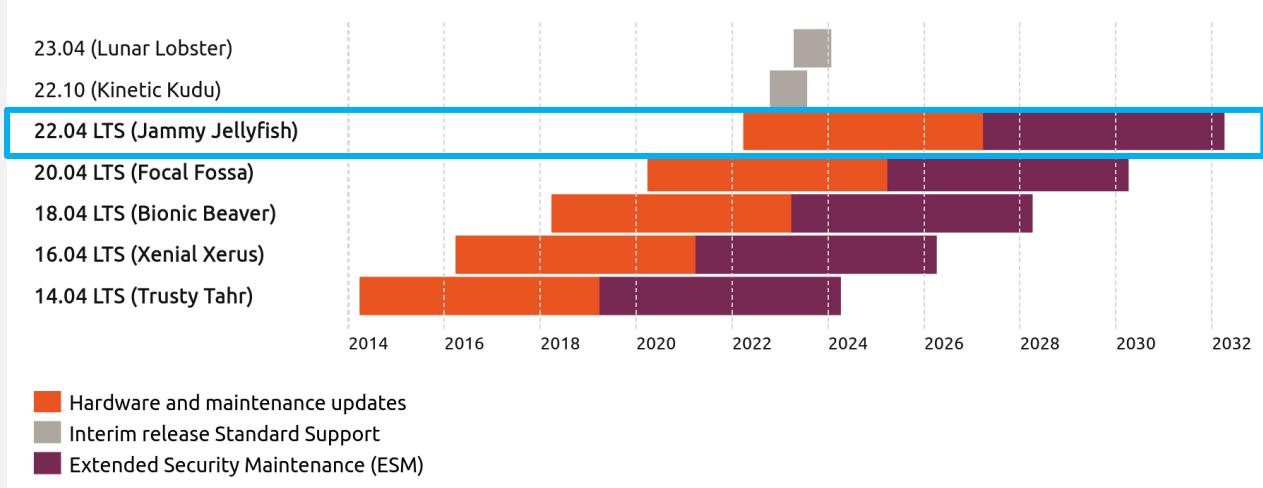
Required Software Components for Robotics?

- **Required Software Components for Robotics:**

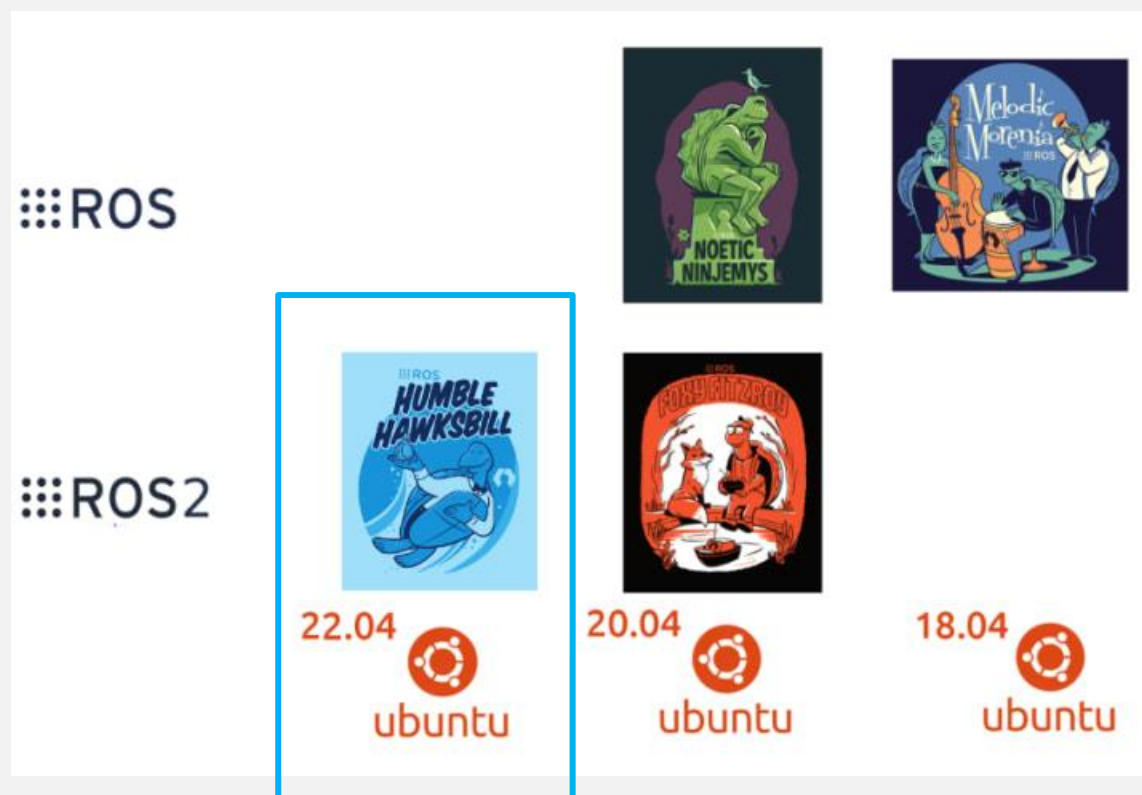
- Framework
- **Platform**
- Language
- Simulator



Platform for Robotics: Ubuntu or Windows or Mac



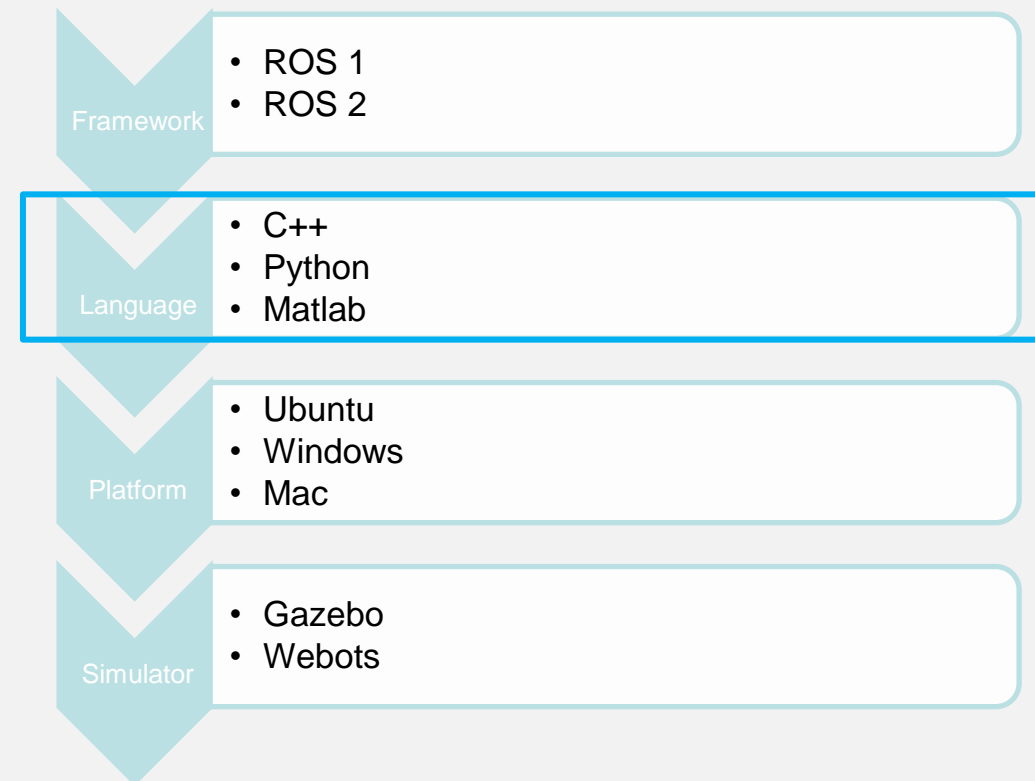
Platform for Robotics: Ubuntu or Windows or Mac



Required Software Components for Robotics?

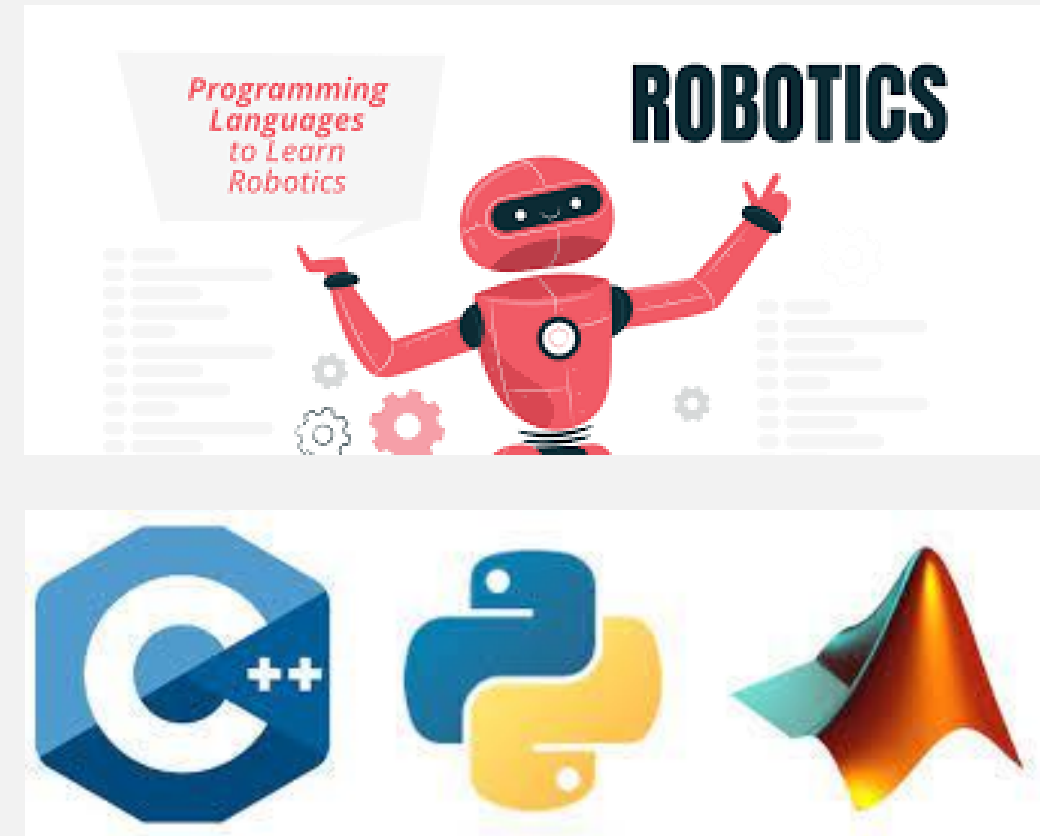
- **Required Software Components for Robotics:**

- Framework
- **Language**
- Platform
- Simulator



Programming Language for Robotics

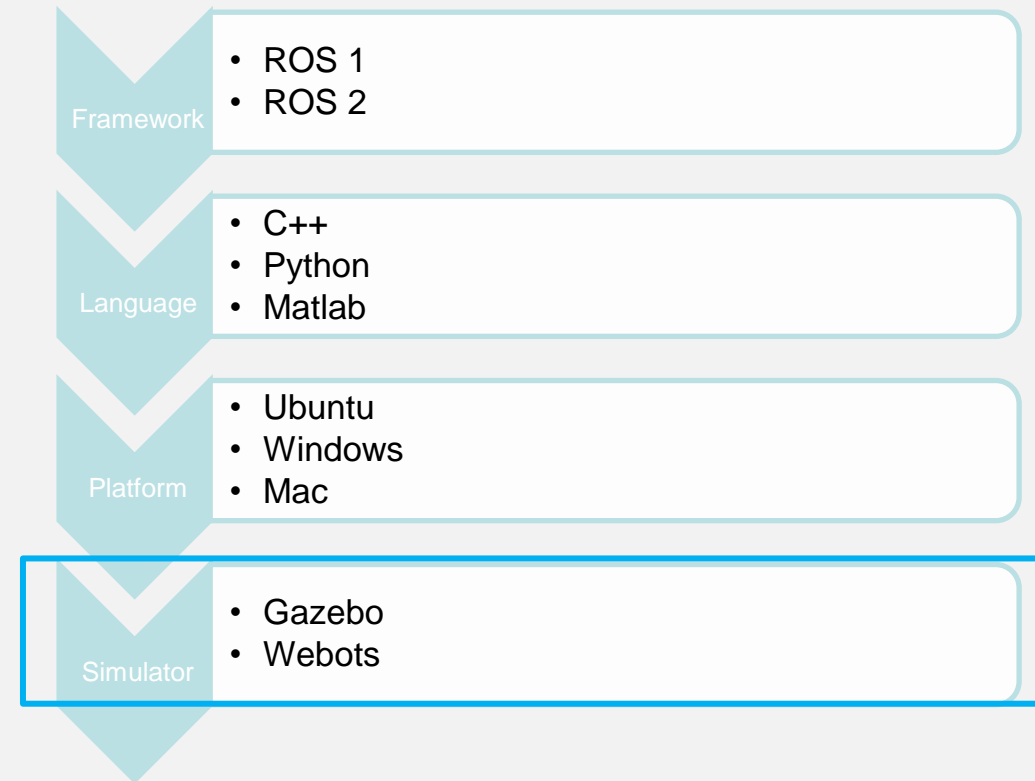
- As such, robotics require a **programming language** that is **versatile, efficient, and easy to use**.
- **C++, Python, and MATLAB** are the most popular programming languages used in robotics, each with their own strengths and weaknesses.



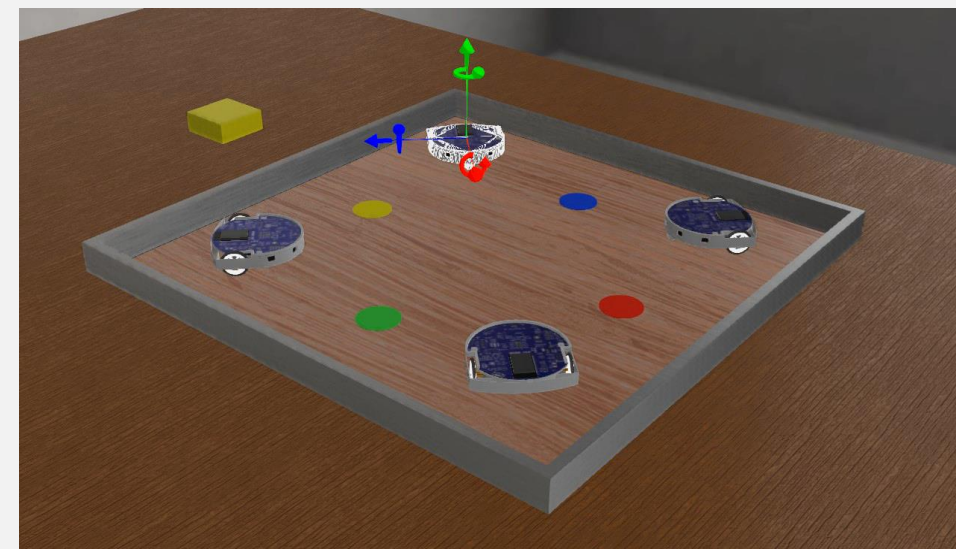
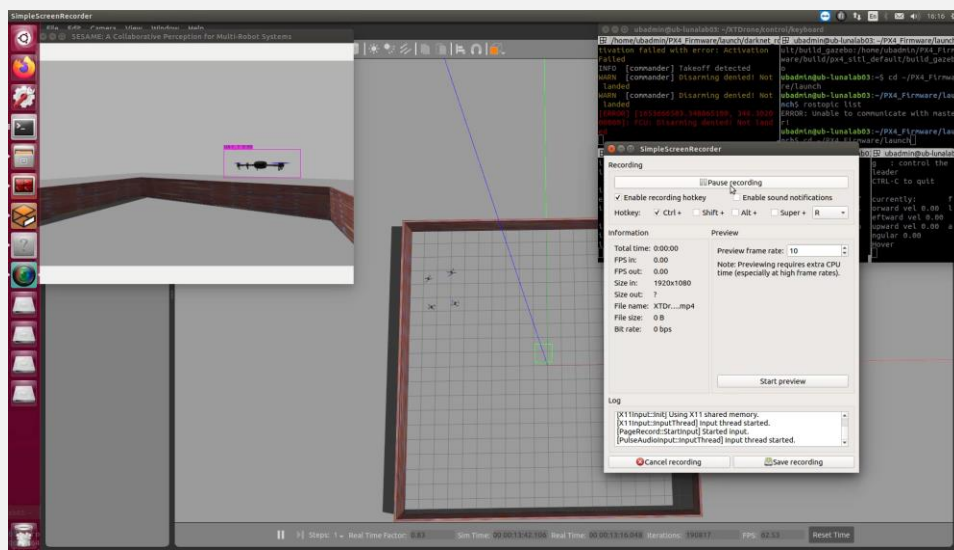
Required Software Components for Robotics?

- **Required Software Components for Robotics:**

- Framework
- Language
- Platform
- Simulator



Simulator for Robotics: Gazebo or Webots



Webots
robot simulation

Example Robots



Robot Model	Controller/Computer	Sensors	Actuators	Price (€)
BulbRobot (2.2.1)	Raspberry Pi 3 B	2 obstacle avoidance IR; 5 line tracking IR; 1 Ultrasonic; 1 IR receiver; 1 joystick; 1 camera;	2 DC motors; 4 RGB LEDs; 1 servo-motor; 1 buzzer;	122 ^(a)
Mona (2.2.2)	ATmega 328	5 proximity IR; 2 magnetic encoders;	2 DC motors;	118 ^(b)
Khepera IV (2.2.3)	Gumstix Overo FireSTORM COM; dsPIC33FJ64 GS608;	12 IR; 1 IMU; 2 microphones; 1 camera;	2 DC motors; 3 RGB LEDs; 1 speaker;	3240 ^(b3)
Thymio (2.2.4)	Microchip PIC24F	7 proximity IR; 2 line tracking IR; 1 accelerometer; 1 thermistor; 1 microphone;	2 DC motors; 39 LEDs; 1 speaker;	151.75 ^(c)
E-puck (2.2.5)	Microchip dsPIC30F6014A	8 proximity IR; 1 accelerometer; 1 gyroscope (later versions); 3 microphones; 1 camera;	2 stepper motors; 1 speaker; 9 red LEDs; Set of green LEDs;	878 ^(b)
E-puck2 (2.2.5)	STM32F407	8 proximity IR; 1 ToF; 1 IMU with magnetometer; 4 microphones; 1 camera; 1 IR receiver; 1 rotary switch;	2 stepper motors; 1 speaker; 5 red LEDs; 4 RGB LEDs; Set of green LEDs;	878 ^(b)
Robobo (2.2.6)	PIC32 based; Phone processor; ^(d)	4 motor encoders; 8 IR; Phone sensors; ^(d)	2 movement motors; 2 phone support motors; 7 RGB LEDs; Phone actuators; ^(d)	399 ²⁴

- <https://www.ros.org/robots/>

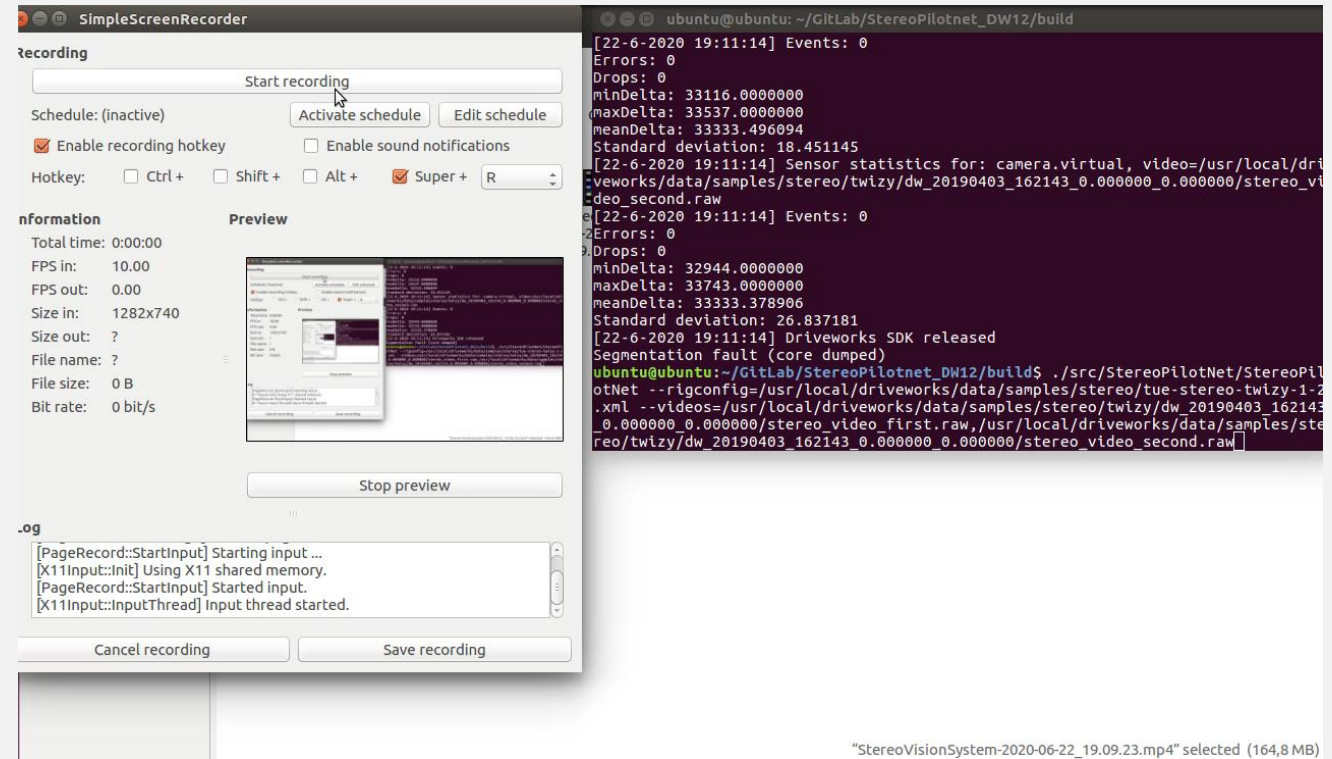
Robotics in Action – Practical Applications

Robotics in Action – Practical Applications

- Autonomous Ground Vehicles (AGVs)
- Unmanned Aerial Vehicles (UAVs)
- Mars Exploration Rovers (MERs)
- Robotics in Medical

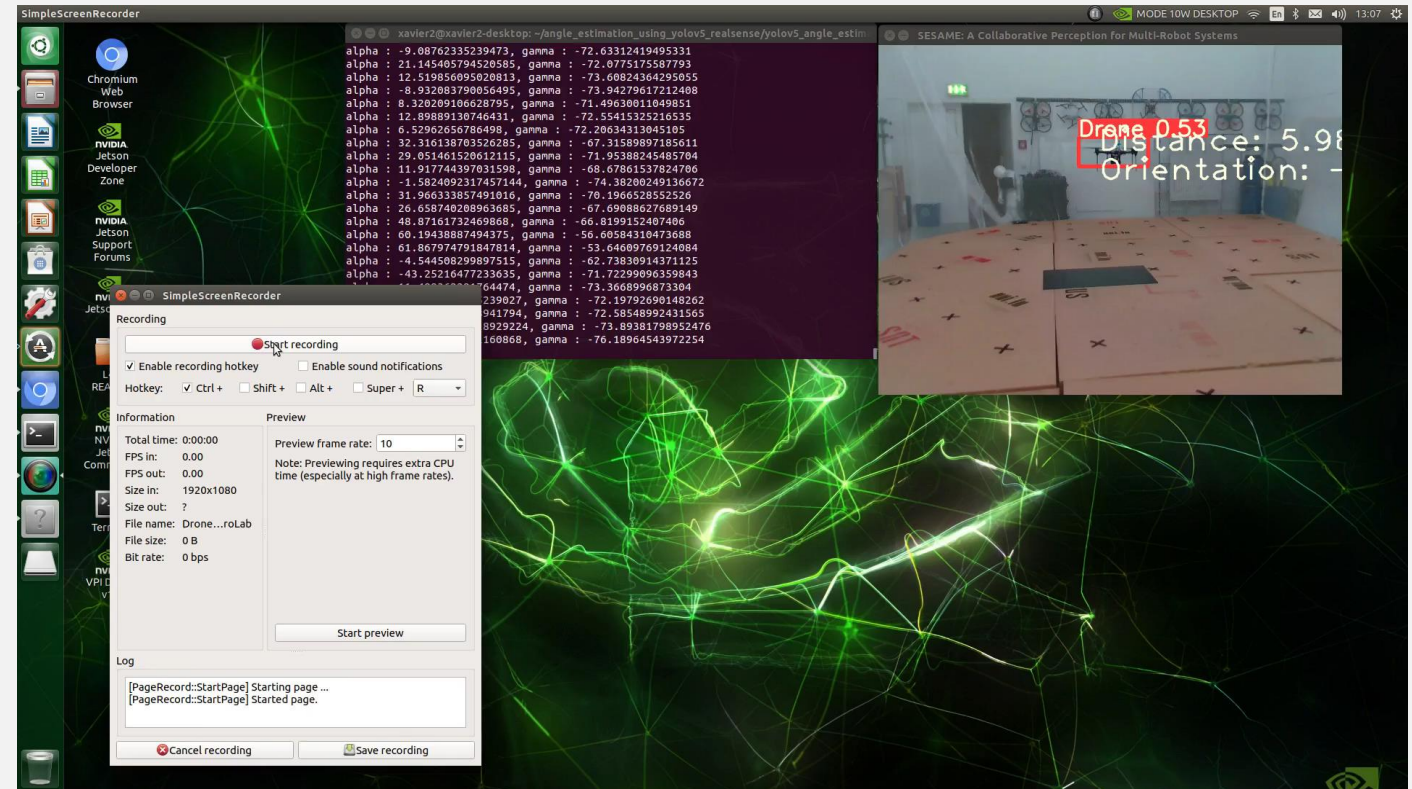
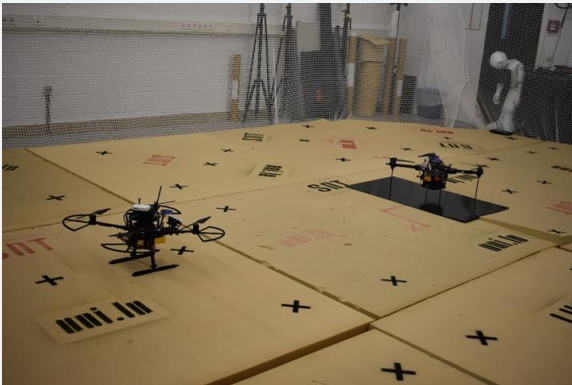


Autonomous Ground Vehicles (AGVs)



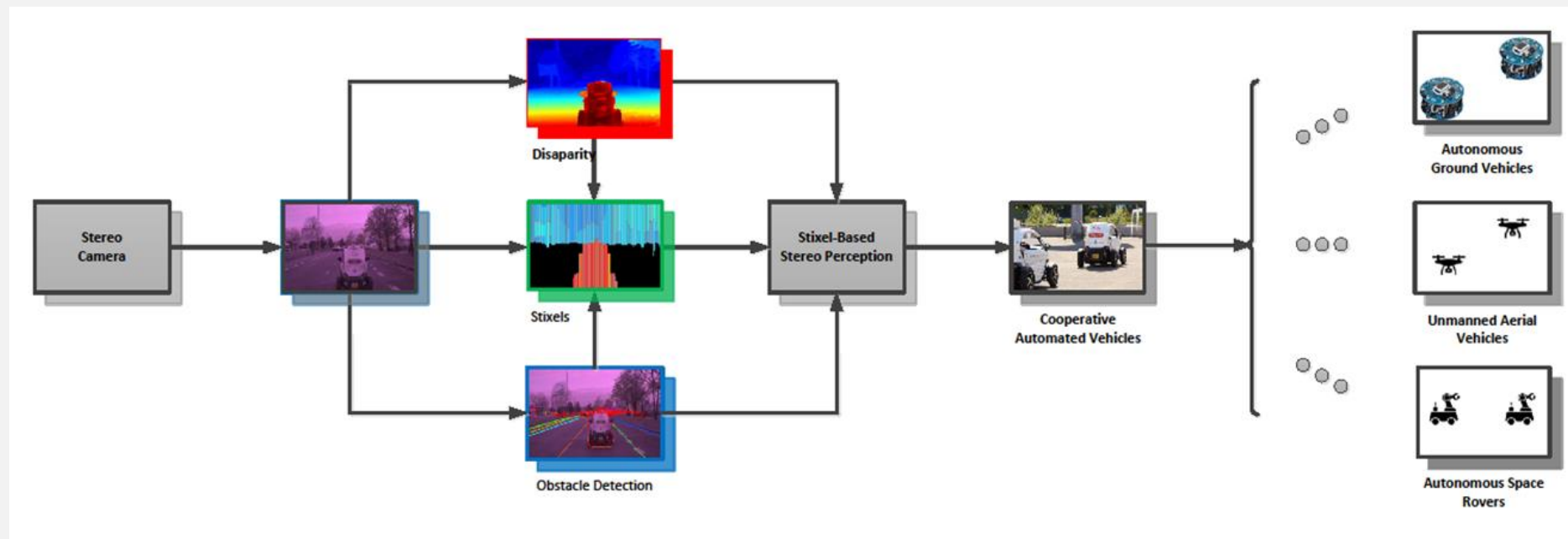
- Narsimlu Kemsaram, Anweshan Das, and Gijs Dubbelman. **A Stereo Perception Framework for Autonomous Vehicles.** In 2020 IEEE 91st Vehicular Technology Conference (VTC), Antwerp, Belgium, 25-28 May 2020.

Unmanned Aerial Vehicles (UAVs)



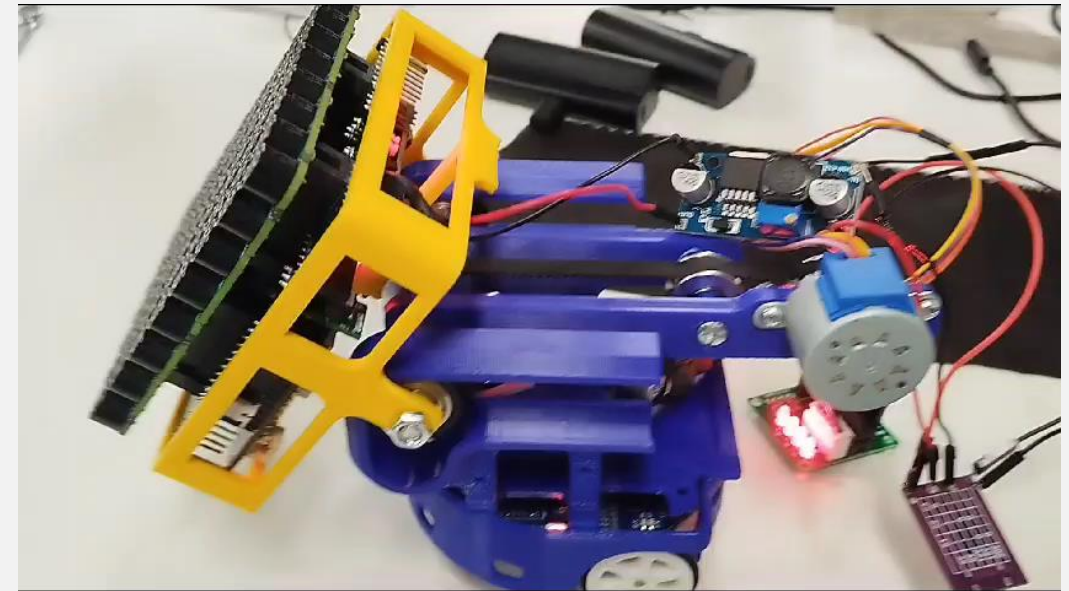
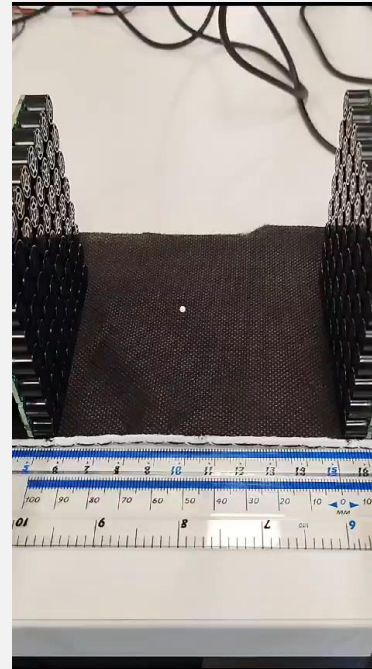
- Narsimlu Kemsaram, Antoine Richard, Sumit Goski, Junlin Song, Abhishek Bera, Miguel Olivares-Mendez. **An Onboard Perception Framework for Multi-UAV Systems** (Submission In Progress).

Mars Exploration Rovers (MERs)



- Narsimlu Kemsaram, Anweshan Das, and Gijs Dubbelman. **A Stixel-Based Stereo Perception for Multi-Robot Systems** at the AIAA Aviation 2023 Forum Conference, 12-16 June 2023, San Diego, CA, USA.

Swarm Robotics



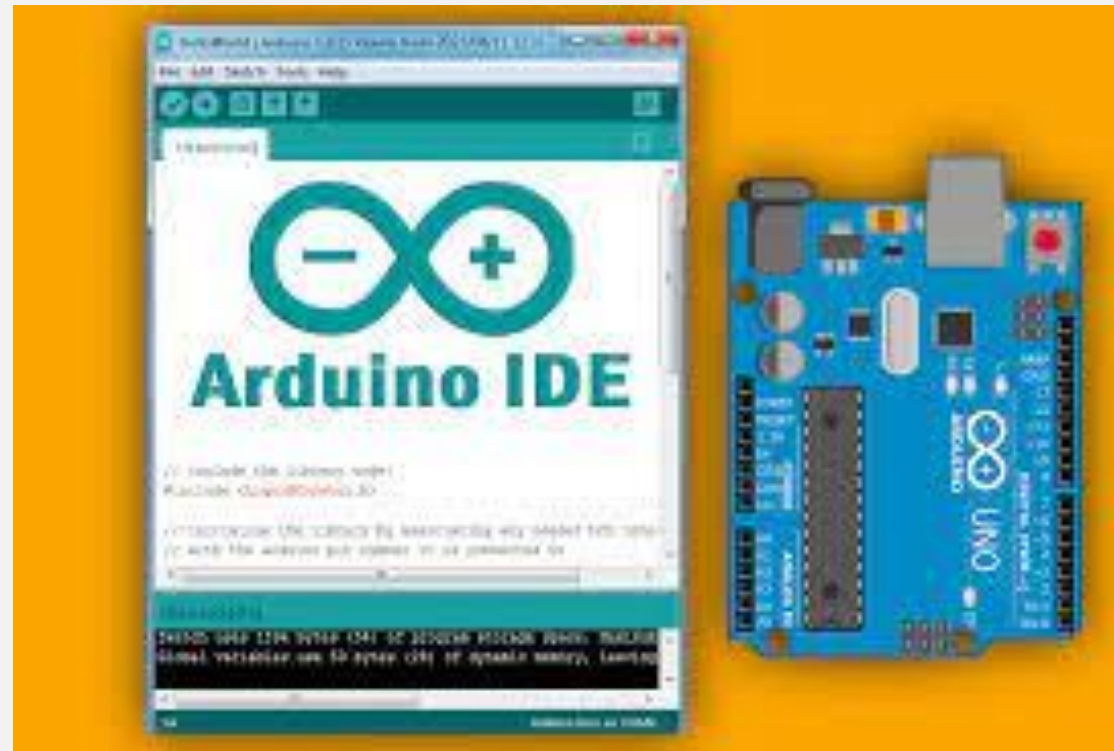
- **AcoustoBots: Acoustic, self-actuated, multi-modal bots for tangible tabletop interactions (In Progress).**

Robotics in Medical



Introduction to Arduino IDE

Robotics Programming: Arduino IDE



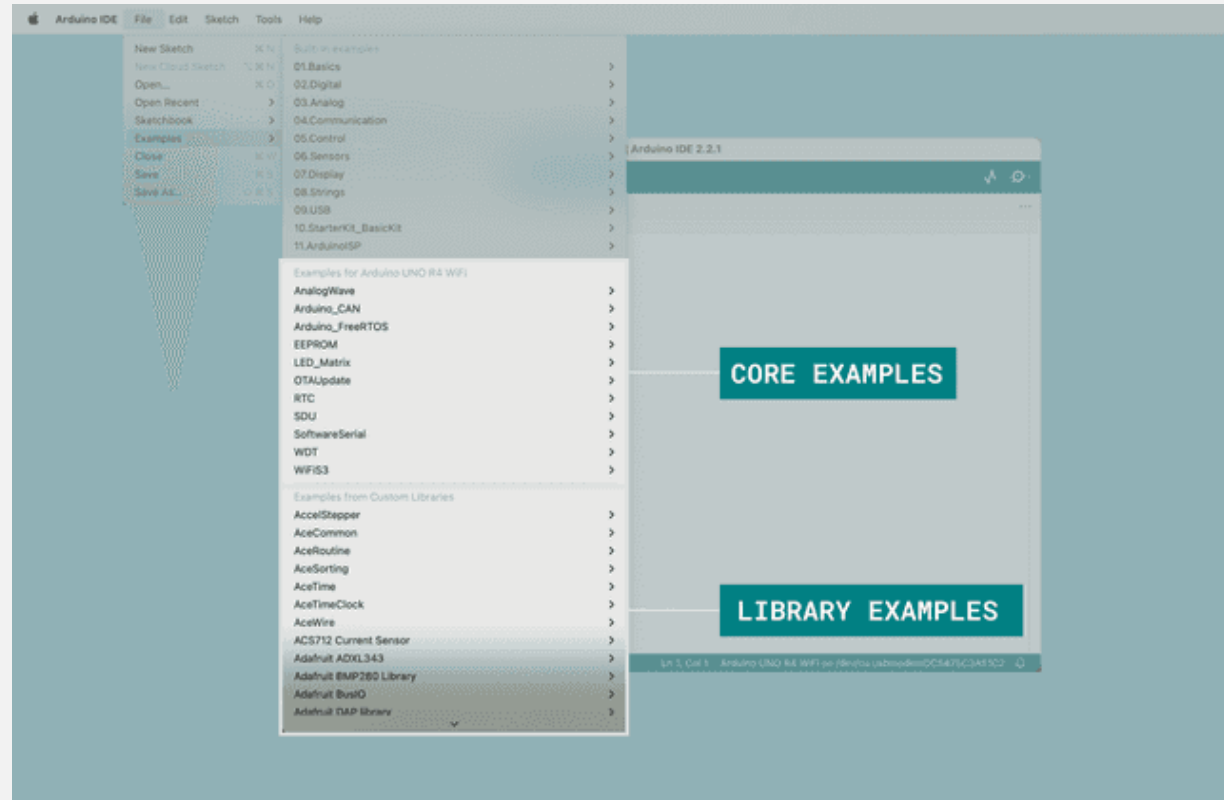
- <https://www.arduino.cc/en/software>.

Robotics Programming: Arduino IDE



- <https://www.arduino.cc/en/software>.

Robotics Programming: Arduino IDE



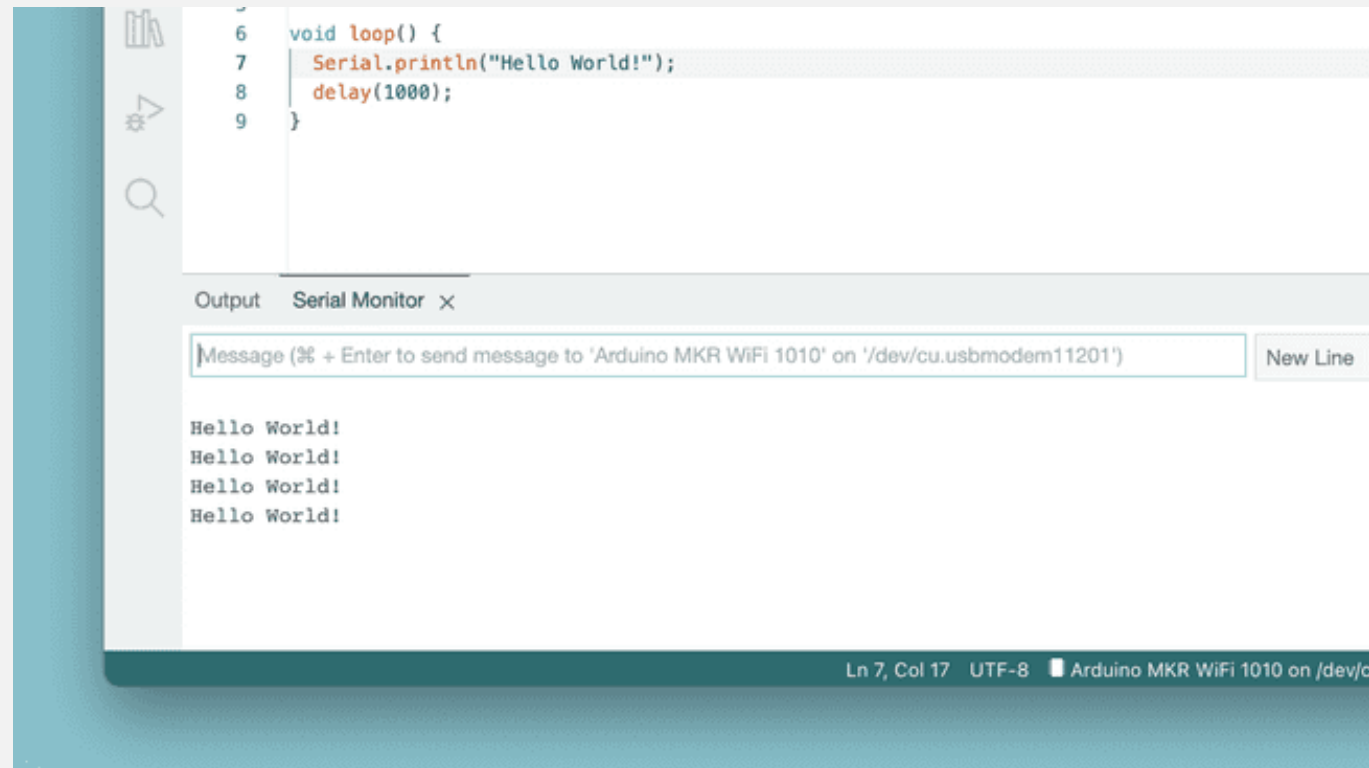
- <https://www.arduino.cc/en/software>.

Robotics Programming: Arduino IDE



- <https://www.arduino.cc/en/software>.

Robotics Programming: Arduino IDE – Serial Monitor



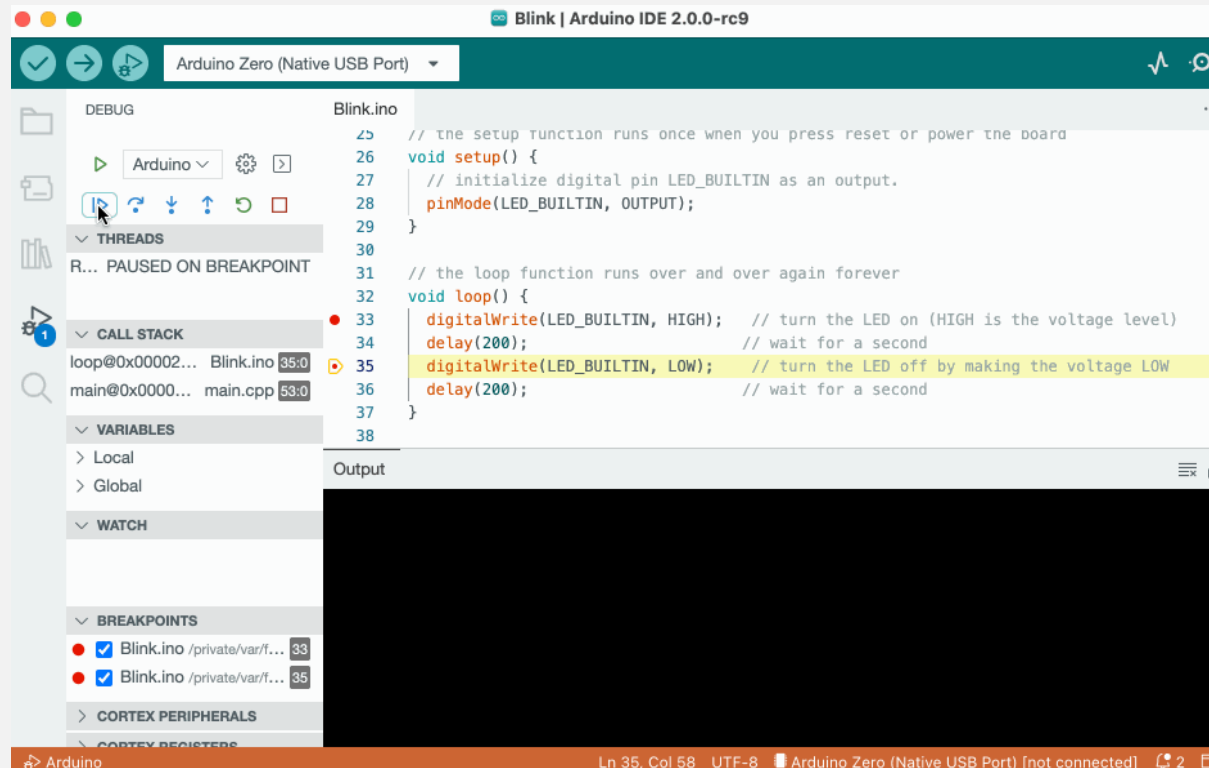
- <https://www.arduino.cc/en/software>.

Robotics Programming: Arduino IDE – Serial Plotter



- <https://www.arduino.cc/en/software>.

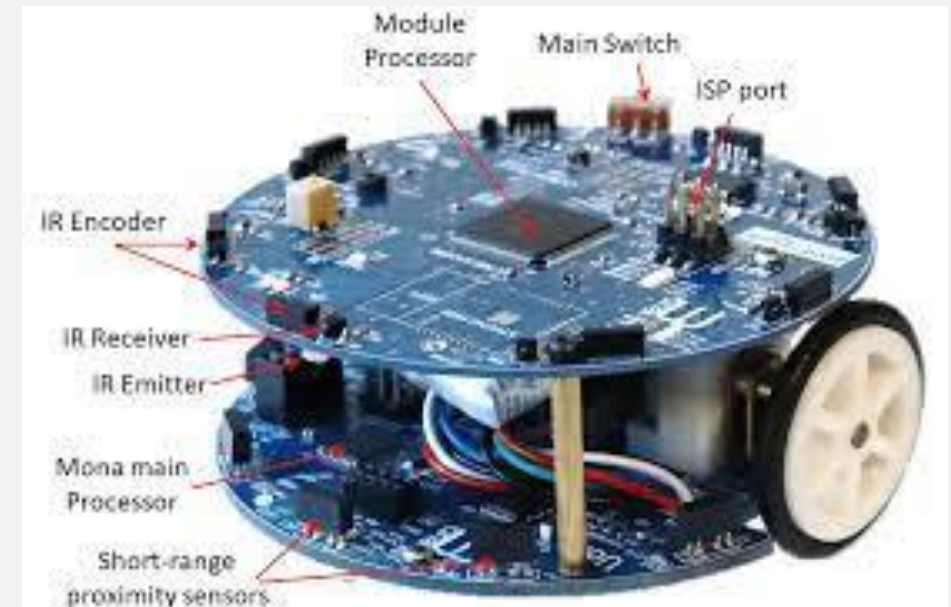
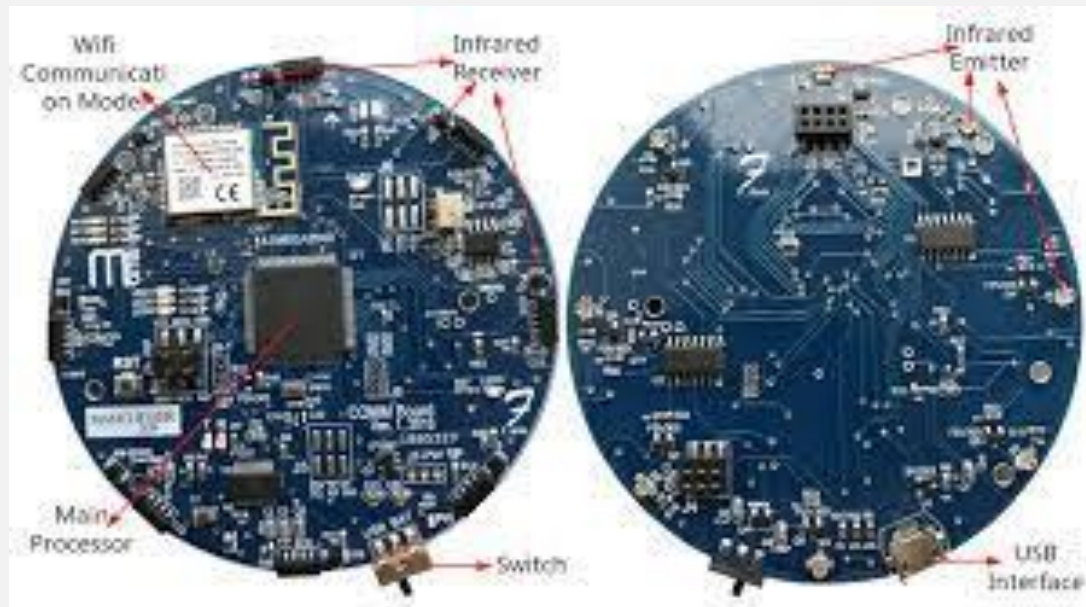
Robotics Programming: Arduino IDE – Debugging



- <https://www.arduino.cc/en/software>.

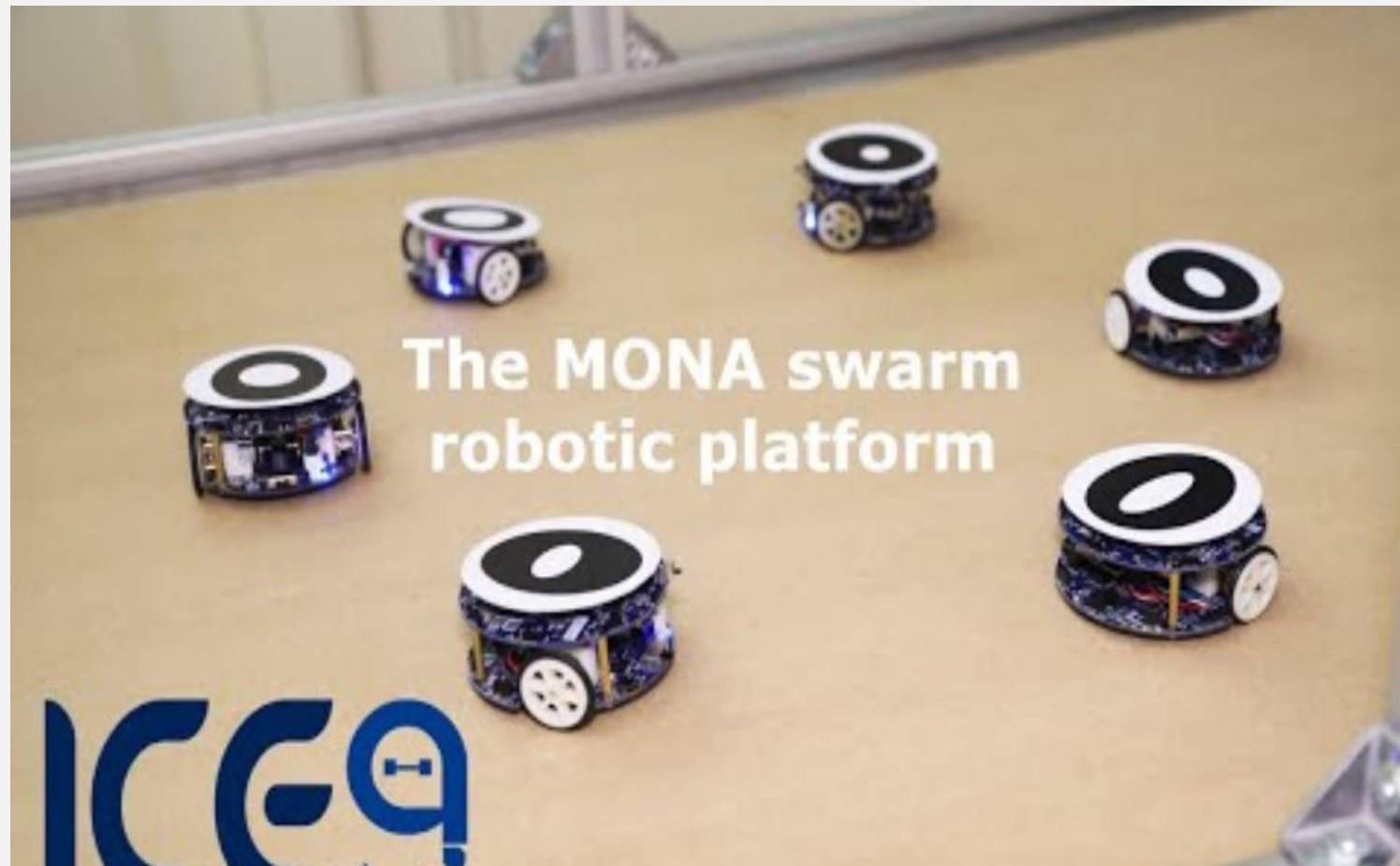
Robotics Programming: Mona Robot

Robotics Programming: Mona Robot



- <https://github.com/MonaRobot/Mona-Platform>.

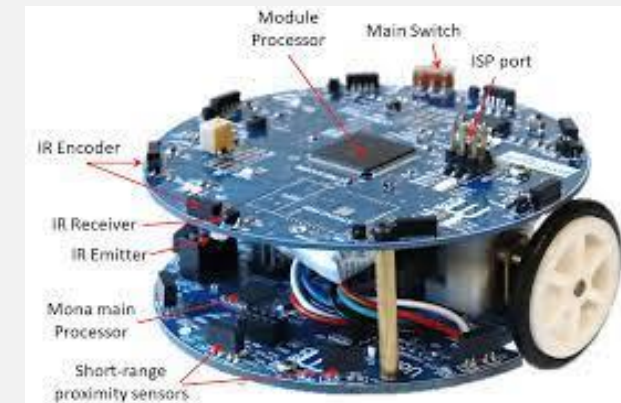
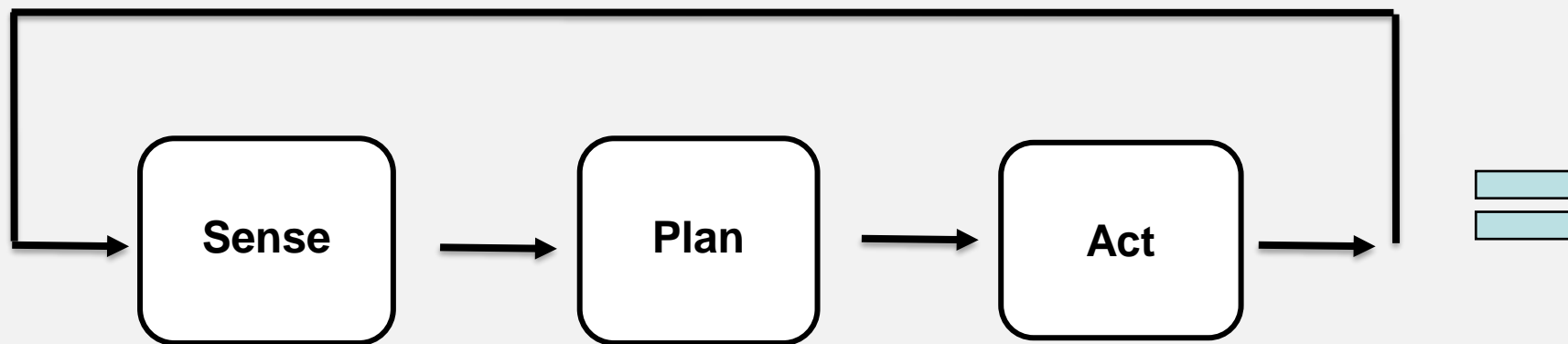
Robotics Programming: Mona Robot



Robotics Programming: Sense, Plan, and Control

Robotics Programming: Sense, Plan, and Control

- A goal-oriented **machine** that generally can **sense**, **plan** and **act** **autonomously**.



Robotics Programming: Pre-requisites

- **Arduino IDE**
- **Mona Robot**
- **ESP32 Libraries**
- **Arduino Collision Avoidance Code**

Robotics Programming: Sense, Plan, and Control

Simple_collision_avoider_led | Arduino IDE 2.2.1

File Edit Sketch Tools Help

ESP32 Wrover Module

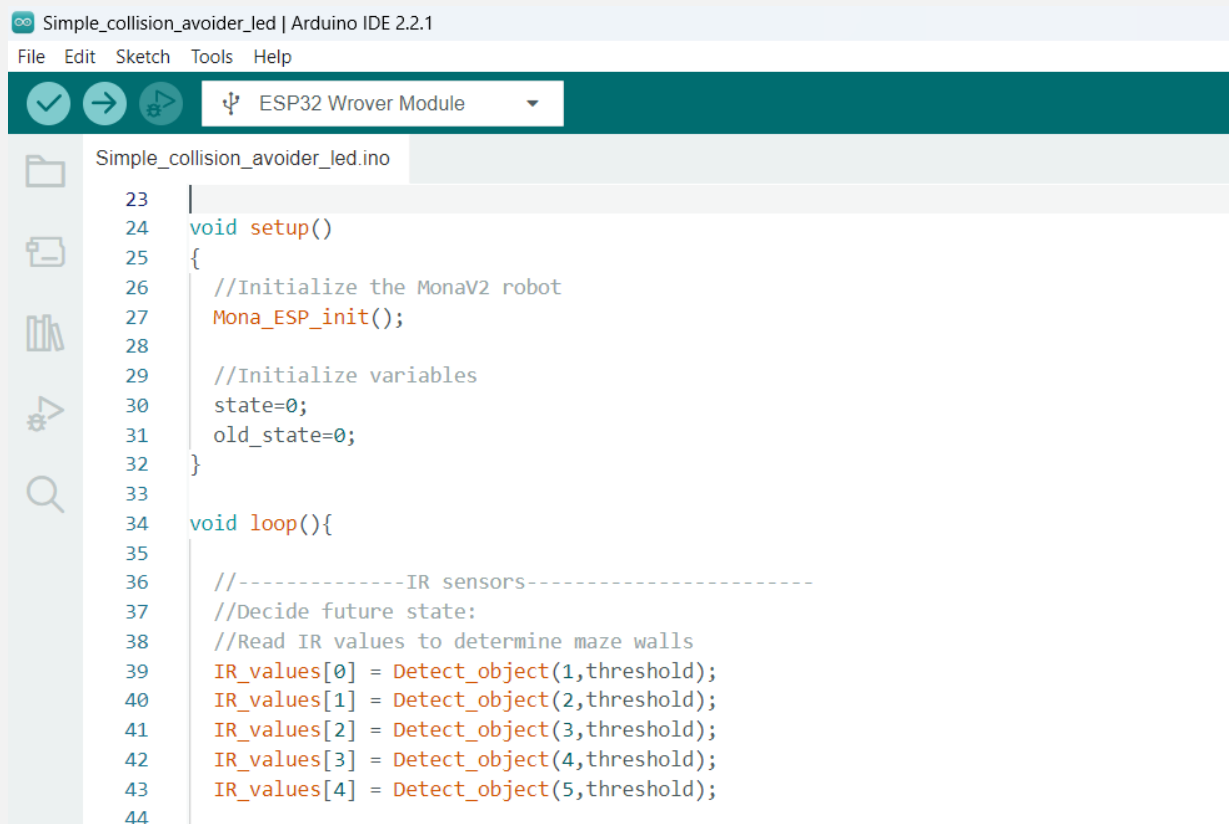
Simple_collision_avoider_led.ino

```

1  /*
2   Simple_collision_avoider.ino - Usage of the libraries Example
3   Using the Mona_ESP library in C style.
4   Created by Bart Garcia, December 2020.
5   bart.garcia.nathan@gmail.com
6   Released into the public domain.
7  */
8  //Include the Mona_ESP library
9  #include <Wire.h>
10 #include "Mona_ESP_lib.h"
11
12 //Variables
13 bool IR_values[5] = {false, false, false, false, false};
14
15 //Threshold value used to determine a detection on the IR sensors.
16 //Reduce the value for a earlier detection, increase it if there
17 //false detections.
18 int threshold = 35; // 0 white close obstacle -- 1023 no obstacle
19
20 //State Machine Variable
21 // 0 -move forward , 1 - forward obstacle , 2 - right proximity , 3 - left proximity
22 int state, old_state;
23

```

Robotics Programming: Sense, Plan, and Control



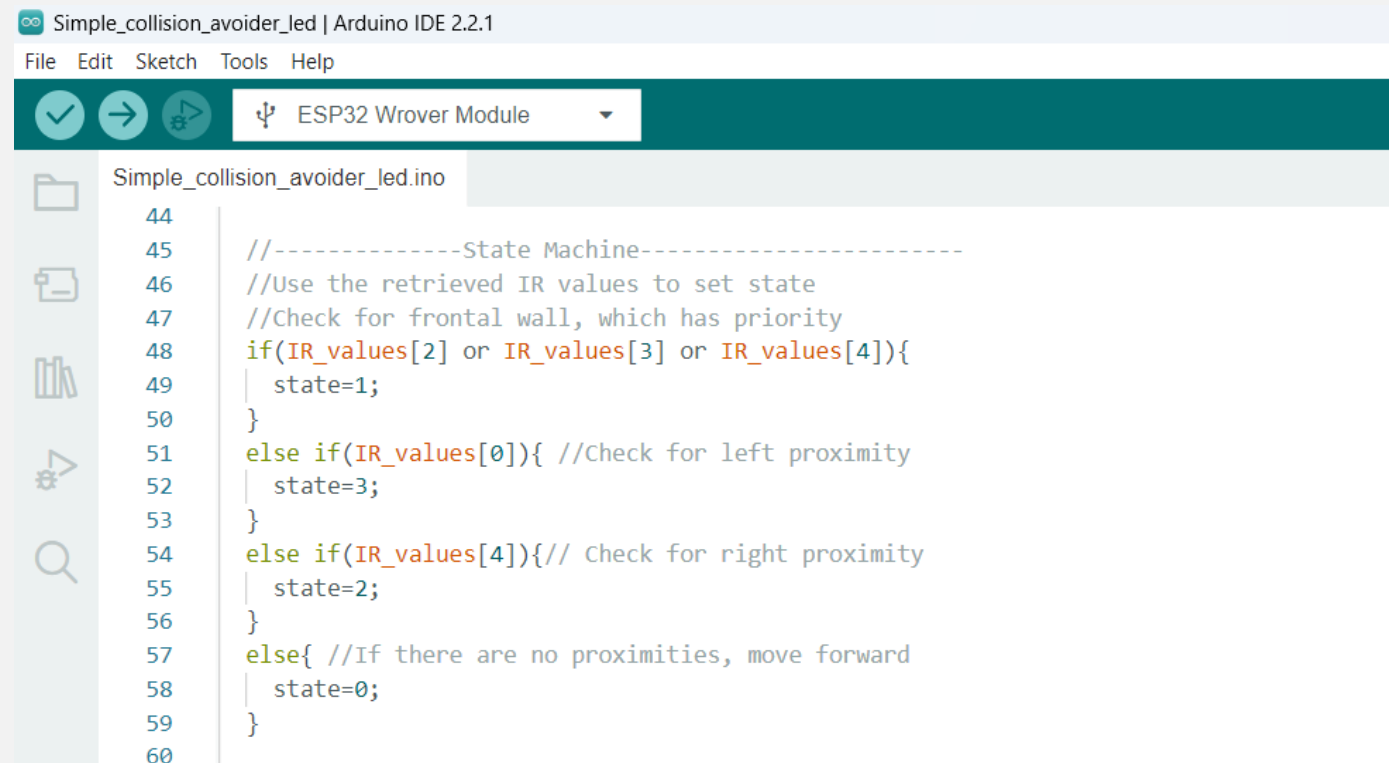
```

Simple_collision_avoider_led | Arduino IDE 2.2.1
File Edit Sketch Tools Help
ESP32 Wrover Module

Simple_collision_avoider_led.ino
23
24 void setup()
25 {
26     //Initialize the MonaV2 robot
27     Mona_ESP_init();
28
29     //Initialize variables
30     state=0;
31     old_state=0;
32 }
33
34 void loop(){
35
36     //-----IR sensors-----
37     //Decide future state:
38     //Read IR values to determine maze walls
39     IR_values[0] = Detect_object(1,threshold);
40     IR_values[1] = Detect_object(2,threshold);
41     IR_values[2] = Detect_object(3,threshold);
42     IR_values[3] = Detect_object(4,threshold);
43     IR_values[4] = Detect_object(5,threshold);
44

```

Robotics Programming: Sense, Plan, and Control



```
Simple_collision_avoider_led | Arduino IDE 2.2.1
File Edit Sketch Tools Help
ESP32 Wrover Module

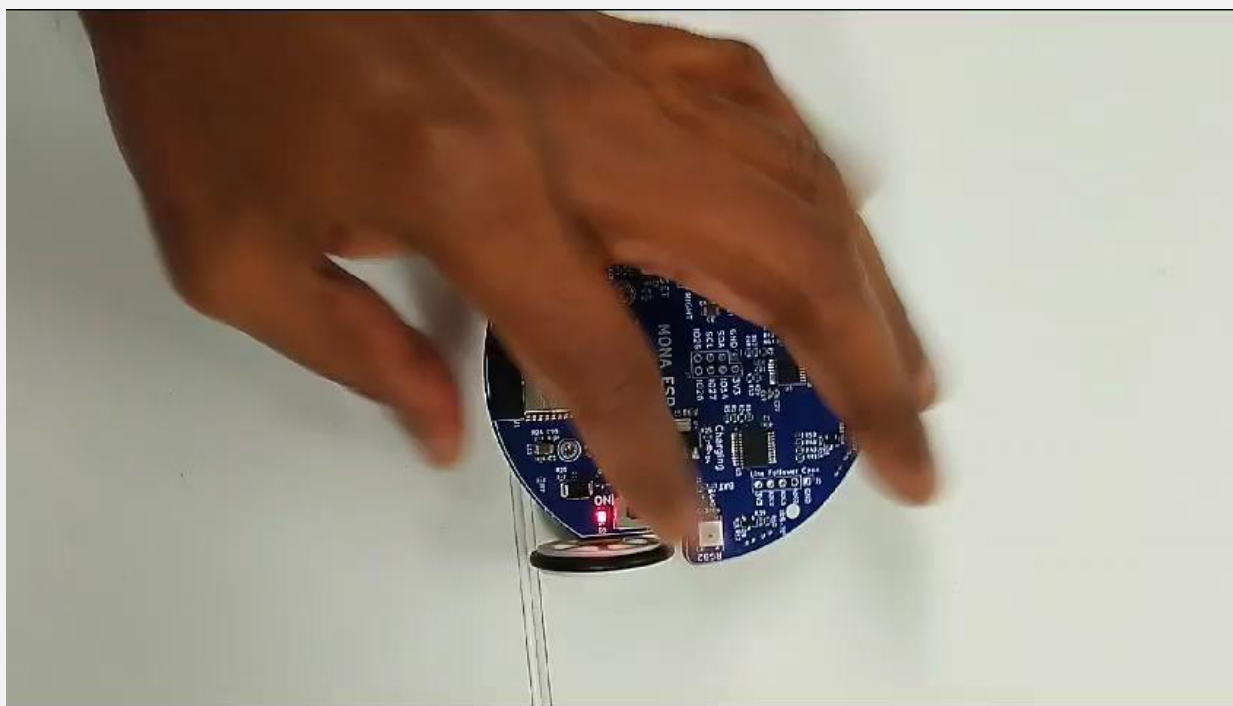
Simple_collision_avoider_led.ino
44
45 //-----State Machine-----
46 //Use the retrieved IR values to set state
47 //Check for frontal wall, which has priority
48 if(IR_values[2] or IR_values[3] or IR_values[4]){
49 | state=1;
50 | }
51 else if(IR_values[0]){ //Check for left proximity
52 | state=3;
53 | }
54 else if(IR_values[4]){ // Check for right proximity
55 | state=2;
56 | }
57 else{ //If there are no proximities, move forward
58 | state=0;
59 | }
60
```

Robotics Programming: Sense, Plan, and Control

```
Simple_collision_avoider_led | Arduino IDE 2.2.1
File Edit Sketch Tools Help
ESP32 Wrover Module

Simple_collision_avoider_led.ino
61 //-----Motors-----
62 //Set motors movement based on the state machine value.
63 if(state == 0){
64     // Start moving Forward
65     Motors_forward(150);
66     // Turn Off LEDs to show that the robot in state=0
67     Set_LED(1,0,0,0);
68     Set_LED(2,0,0,0);
69 }
70 if(state == 1){
71     Set_LED(1,0,0,0);
72     Set_LED(2,0,0,0);
73     //Spin to the left
74     Motors_spin_left(100);
75     // Turn LEDs to show that the robot in state=1
76     Set_LED(1,20,0,0);
77 }
78 if(state == 2){
79     Set_LED(1,0,0,0);
80     Set_LED(2,0,0,0);
81     //Spin to the left
82     Motors_spin_left(100);
83     // Turn LEDs to show that the robot in state=2
84     Set_LED(1,20,0,0);
85 }
86 if(state == 3){
87     Set_LED(1,0,0,0);
88     Set_LED(2,0,0,0);
89     //Spin to the right
90     Motors_spin_right(100);
91     // Turn LEDs to show that the robot in state=3
92     Set_LED(2,20,0,0);
93 }
```

Robotics Programming: Sense, Plan, and Control



Robotics Programming: Sense, Plan, and Control

Demo: Deploy Code onto the Mona Robot!

Thank you!