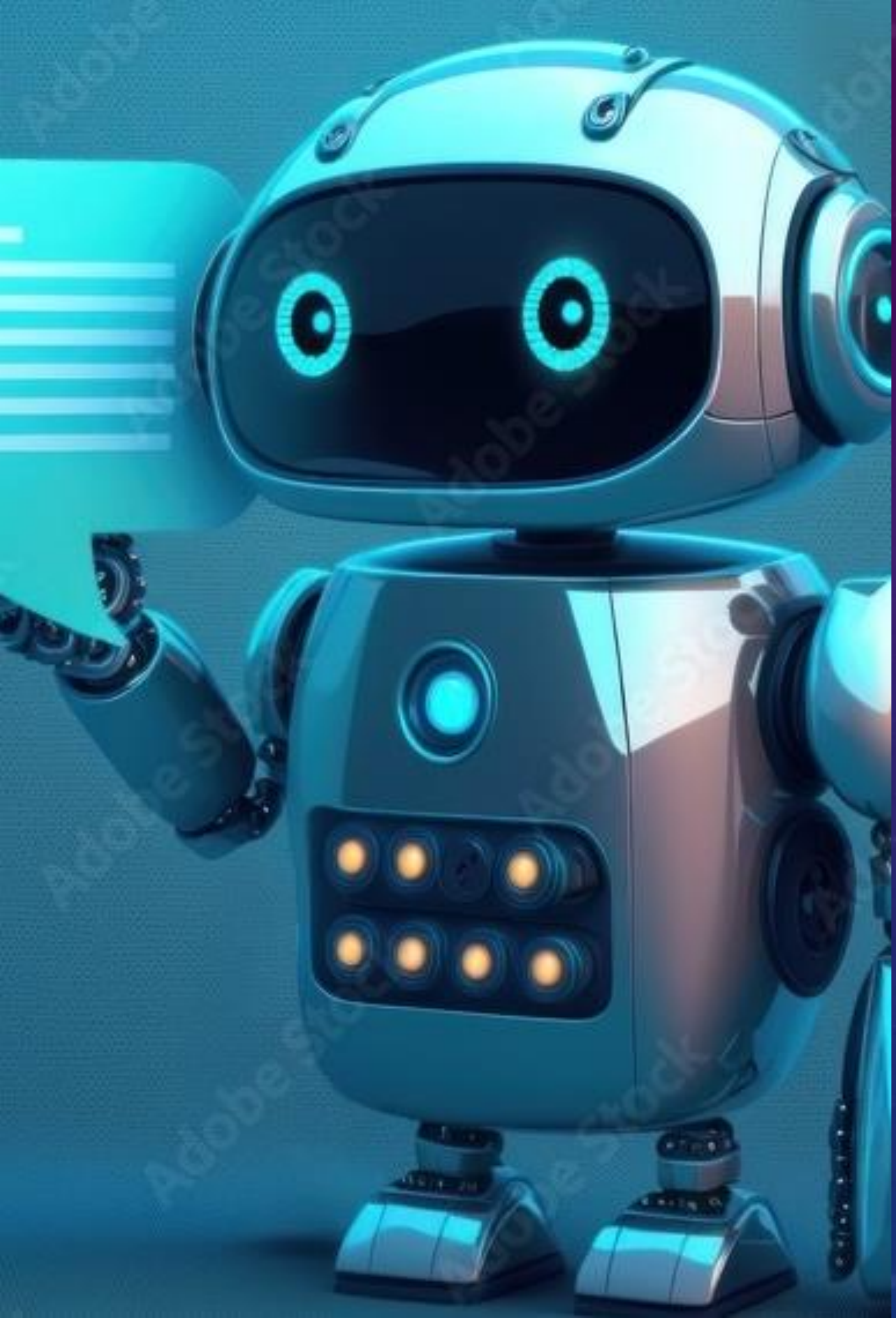
INTRODUCTION TO ROBOTICS

AGENDA

- Types of robots
- Robotics Applications
- Technologies used in Robotics
- Linux installation
- ROS Installation



• SOFTWARE ROBOTS

also known as software bots, are computer programs designed to automate repetitive and rule-based tasks, mimicking human actions within digital systems. They excel at tasks like data entry, form filling, and report generation, reducing errors and increasing operational efficiency. Software robots can integrate with various applications, scale up easily, and offer cost savings through labor reduction.

These bots are often part of Robotic Process Automation (RPA), where they play a pivotal role in streamlining business processes and freeing up human resources for more complex, value-added tasks. RPA tools facilitate the configuration and management of software robots, making automation accessible and adaptable to a wide range of industries and functions.

COMMON TYPES OF HW ROBOTS

- **Automated Guided Vehicles (AGVs)**
- **Autonomous Mobile Robots (AMRs)**
- **Underwater Robots (ROVs and AUVs)**
- **Articulated Robots**
- **Humanoid**
- **Cobots**
- Hybrid

NON-FIXED VS FIXED

Mobile	Stationary
•AMRs •AGVs •Humanoids •Hybrids	•Articulated robots •Cobots

ROBOTICS APPLICATIONS

- Industrial
- Farming and Agriculture
- Logistics
- Retail and Hospitality
- Smart Cities
- Health care

ROBOTS IN ROS

- examples of robots that are compatible with ROS (Robot Operating System)

  Turtlebot3 , Husky, Franka emika and Kobuki and there a lot more.

A SMART CITY PROJECT WITH HUSKY

- 2 x depth cameras
- 1 x 3D LiDAR
- 1 x IMU
- 1 x u-blox 7 GPS

- API: ROS, C++ and python

  for more info visit:
  https://www.generationrobots.com/blog/en/a-smart-city-project-with-husky/

LOGISTICS ROBOTS

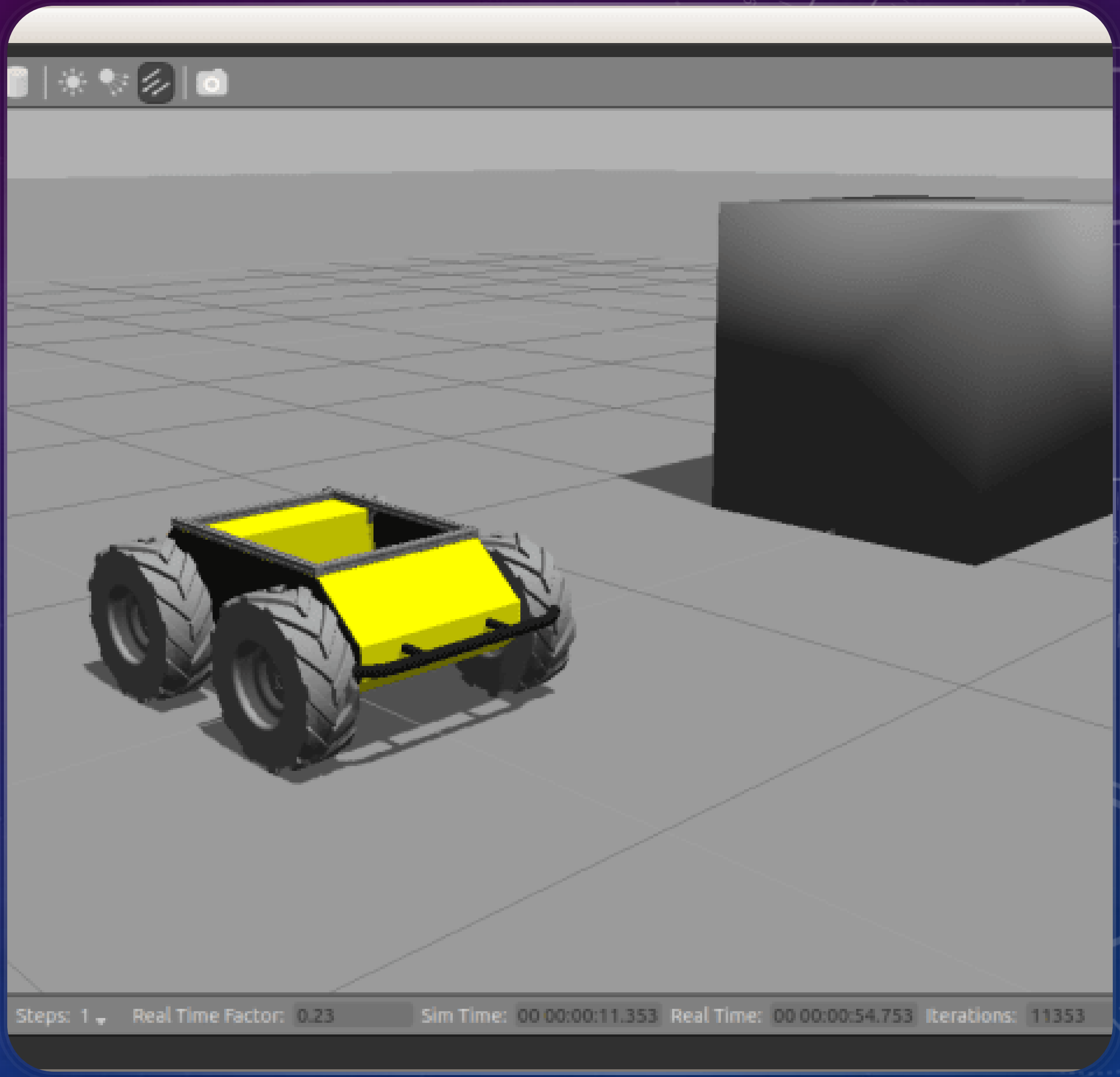
- These robots play essential roles in streamlining and optimizing the movement of goods within warehouses, distribution centers, and transportation hubs. Here are several types of robots used in logistics more than 2 billion dollars' worth robots used in 2022.
- Used by Companies which has large warehouses such as Amazon.

INDUSTRIAL ROBOTS

- INDUSTRIAL ROBOTS PLAY KEY ROLE IN INDUSTRY 4.0 TRANSFORMATION
- KUKA IS A FAMOUS MANUFACTURE FOR SUCH ROBOTS AND THESE ROBOTS CAN BE SIMULATED AND CONTROLLED/AUTOMATED/PROGRAMMED BY ROS
- IT MAY INVOLVE INDUSTRIAL INTERNET OF THINGS (IIOT) PROTOCOLS TO CONTROL OR MONITOR THESE ROBOTS.

TOOLS NEEDED TO MODEL A ROBOT

- **Robotics Simulation Software:**
- Robotics simulation software allows you to create a virtual environment in which you can model the robot's physical characteristics, sensors, and behaviors. Popular robotics simulation platforms include:
- **Gazebo:** An open-source robotics simulator that can simulate the dynamics of a wide range of robots and environments.
- **Webots:** A versatile robotics simulation platform that supports various robot models and programming languages.
- **V-REP (CoppeliaSim):** A multi-platform simulation software with a user-friendly interface and support for modeling complex robotic systems.



PROGRAMMING ENVIRONMENT

- you'll need a programming environment to develop the robot's control algorithms. Common frameworks used in robotics modeling include:

- **ROS (Robot Operating System):** A flexible framework for writing robot software that provides communication tools and libraries for various robotic tasks.
- ROS has two interface programming languages:

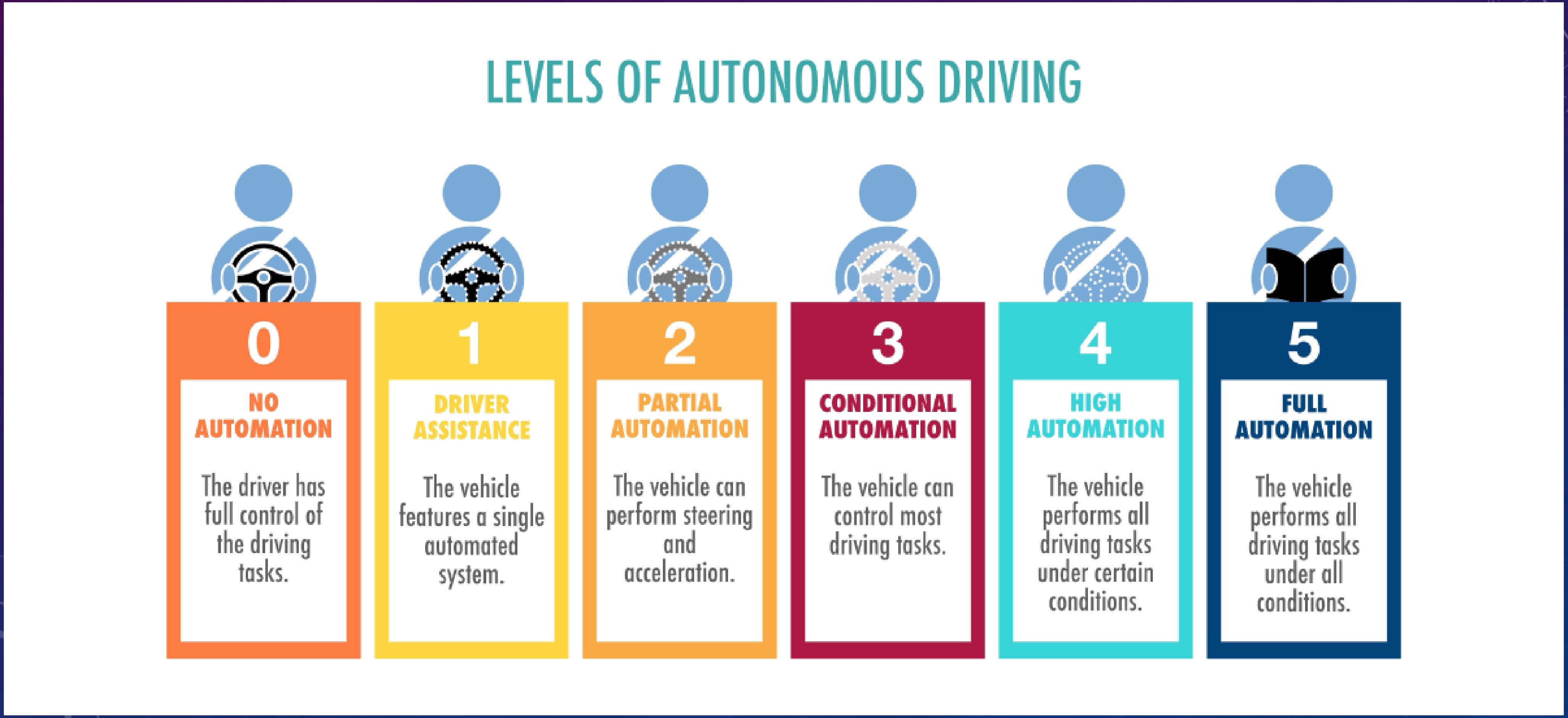
- **C++:** Often used for high-performance robot control software.

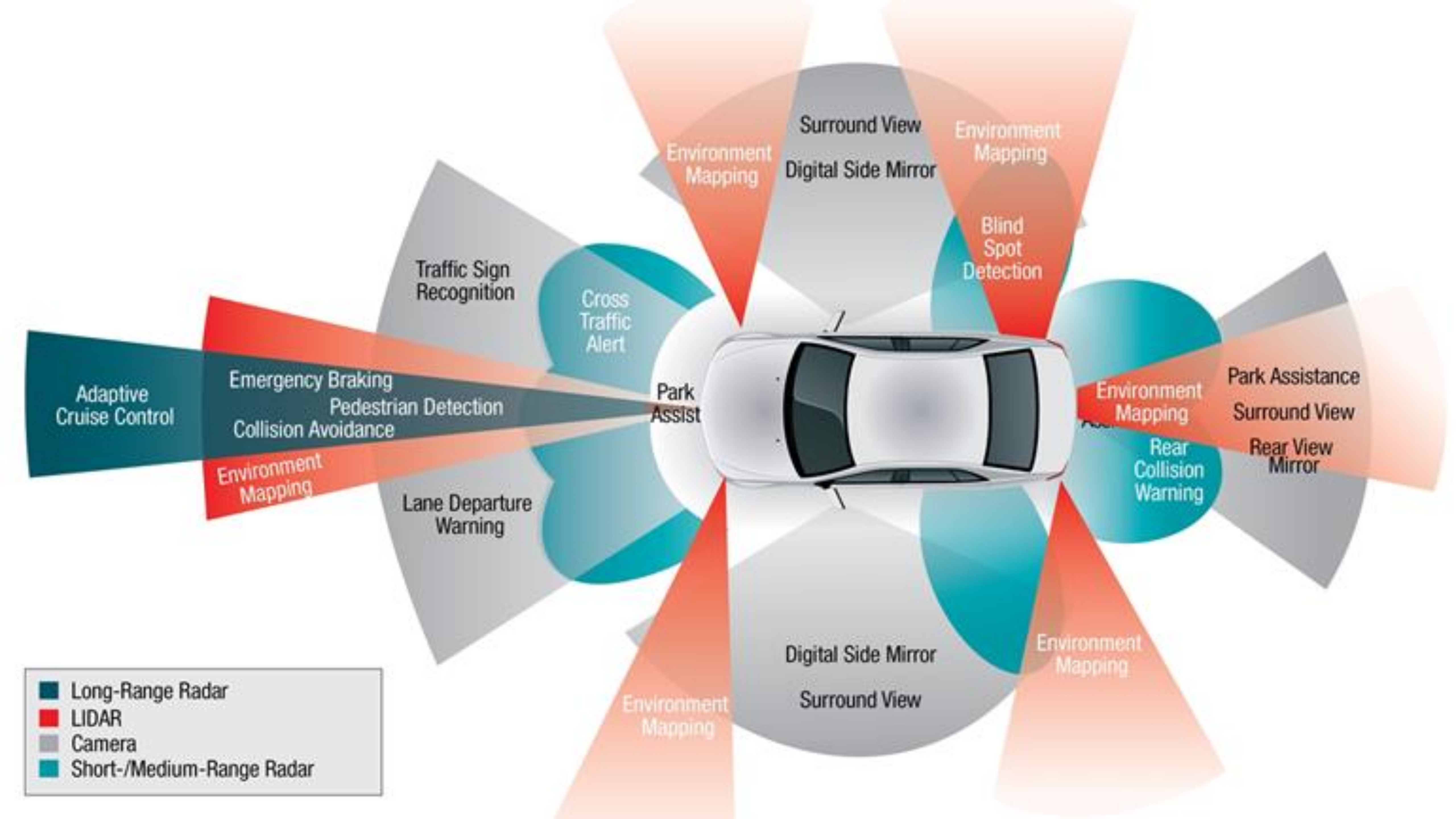
- **Python:** A popular programming language for developing robot control algorithms and scripts.

SELF-DRIVING CAR AS A ROBOT EXAMPLE

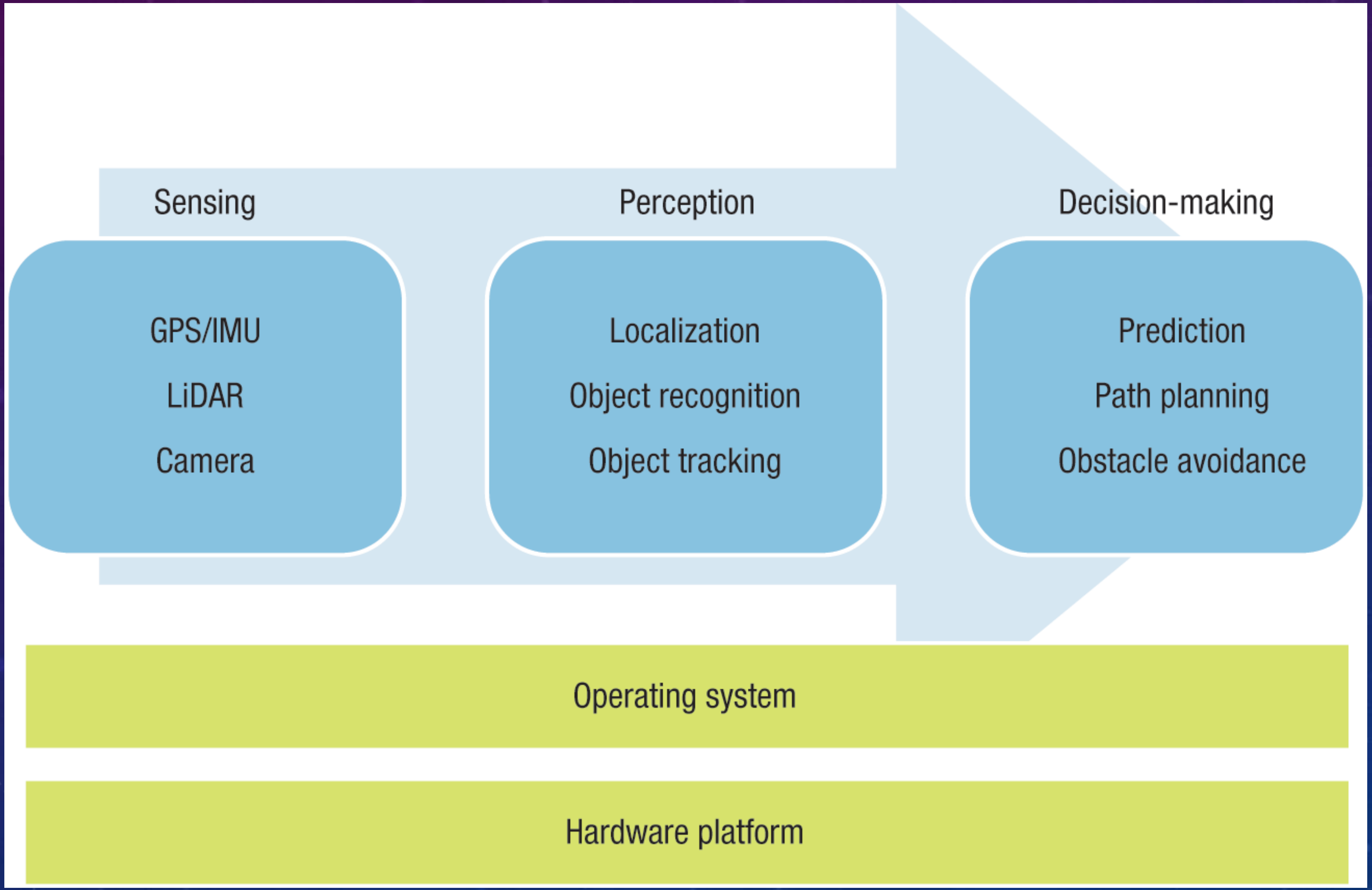
- Self-driving car is a robot which is able to navigate and steer autonomously and the level of human interaction depends on level of autonomy.

AUTONOMOUS VEHICLE STACK




Surround View
Digital Side Mirror
Environment Mapping
Environment Mapping
Blind Spot Detection
Traffic Sign Recognition
Cross Traffic Alert
Park Assist
Adaptive Cruise Control
Emergency Braking
Pedestrian Detection
Collision Avoidance
Environment Mapping
Lane Departure Warning
Environment Mapping
Park Assistance
Surround View
Rear View Mirror
Rear Collision Warning
Digital Side Mirror
Surround View
Environment Mapping
Environment Mapping
Long-Range Radar
LIDAR
Camera
Short-/Medium-Range Radar

AUTONOMOUS DRIVING STACK

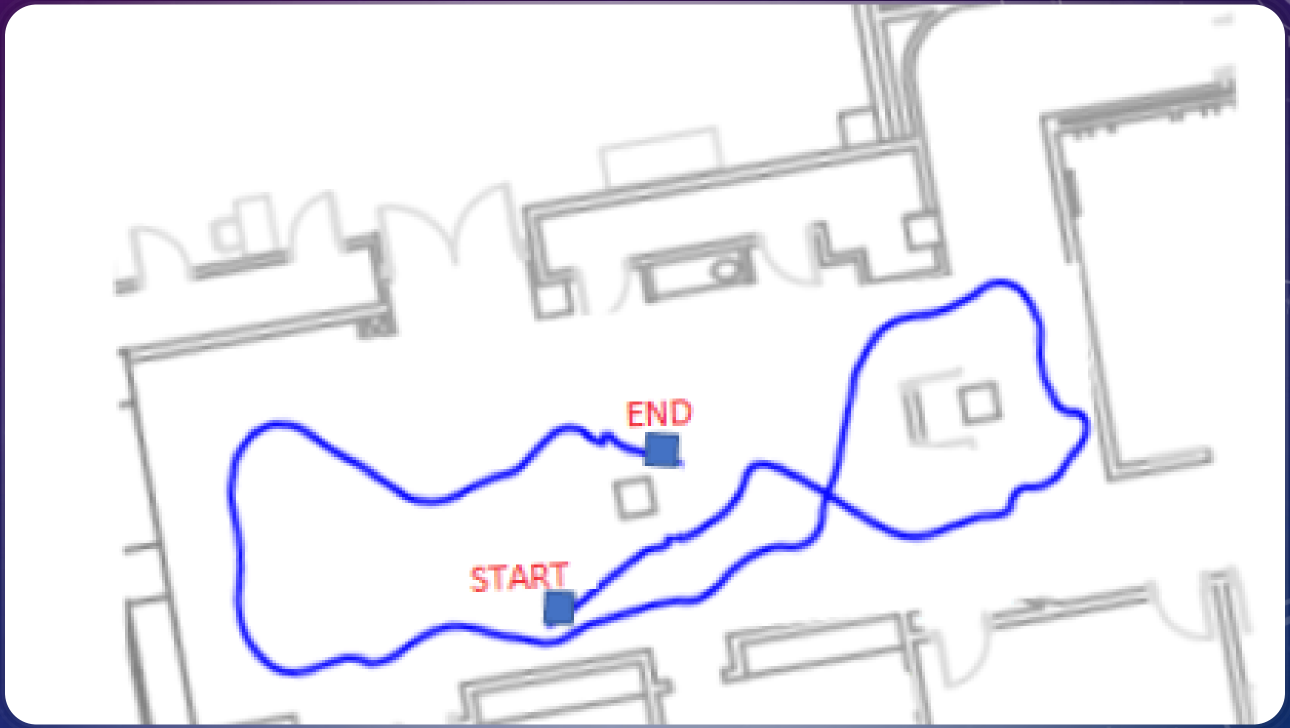


3D DEPTH PERCEPTION

- **3D Depth Perception:** Some robots use stereo cameras or depth-sensing cameras like LiDAR (Light Detection and Ranging) to perceive the depth of objects in their environment. This 3D depth perception enables precise distance measurements and helps robots create accurate 3D maps of their surroundings.

LOCALIZATION AND MAPPING

- Localization refers to the process of determining a robot's position (often represented as a set of coordinates) within its environment. It answers the question: "Where is the robot?" Accurate localization is crucial for robots to move safely, avoid obstacles, and reach their intended destinations. There are various methods for robot localization:

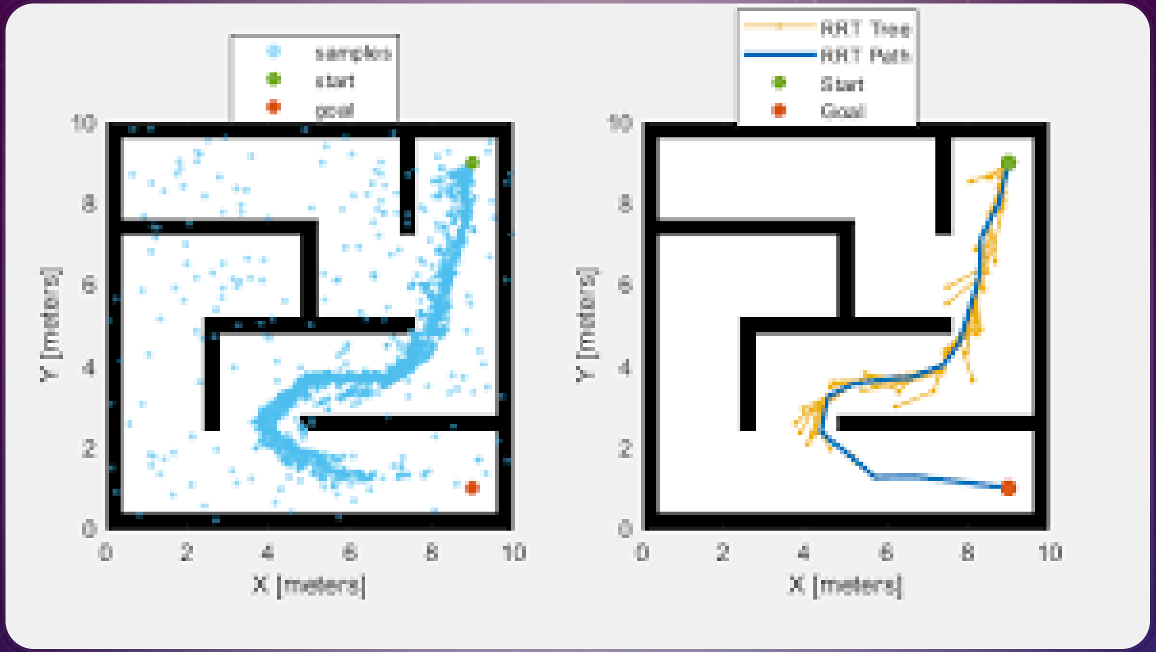


LOCALIZATION METHODOLOGY

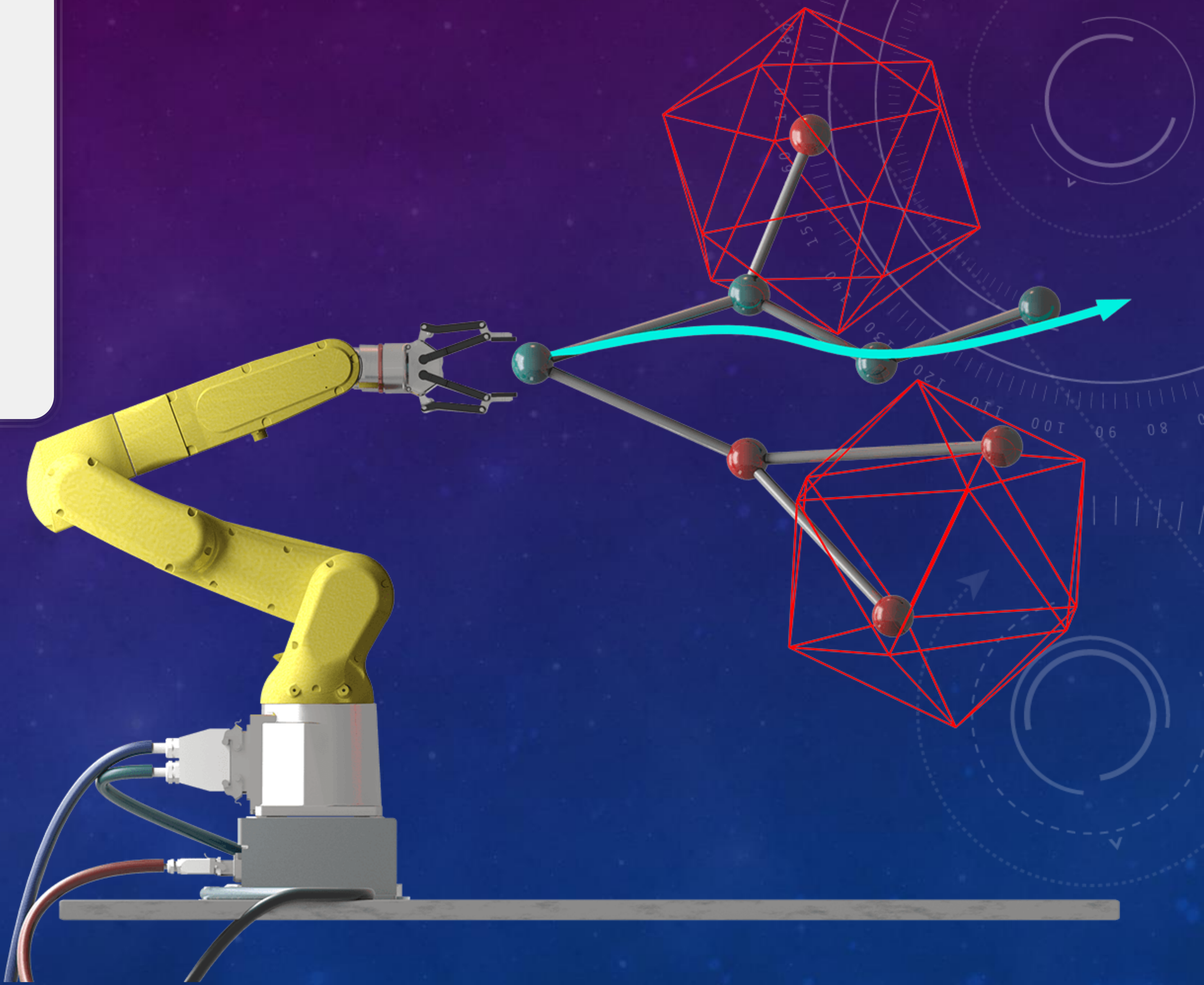
Odometry: Odometry uses information from wheel encoders or sensors to estimate the robot's position based on the distance and direction it has traveled. Over time, errors can accumulate in odometry-based localization, so it is often used in conjunction with other methods.

Sensor Fusion: This approach combines data from multiple sensors, such as GPS, IMU (Inertial Measurement Unit), lidar, and cameras, to improve localization accuracy. Sensor fusion algorithms, like Kalman filters and particle filters, are commonly used to integrate sensor data and estimate the robot's pose (position and orientation).

SLAM (Simultaneous Localization and Mapping): SLAM is a technique that allows a robot to simultaneously build a map of its environment and determine its own location within that map. It is particularly useful in unknown or dynamically changing environments. SLAM algorithms use sensor data, such as lidar or depth cameras, to create a map and estimate the robot's pose.



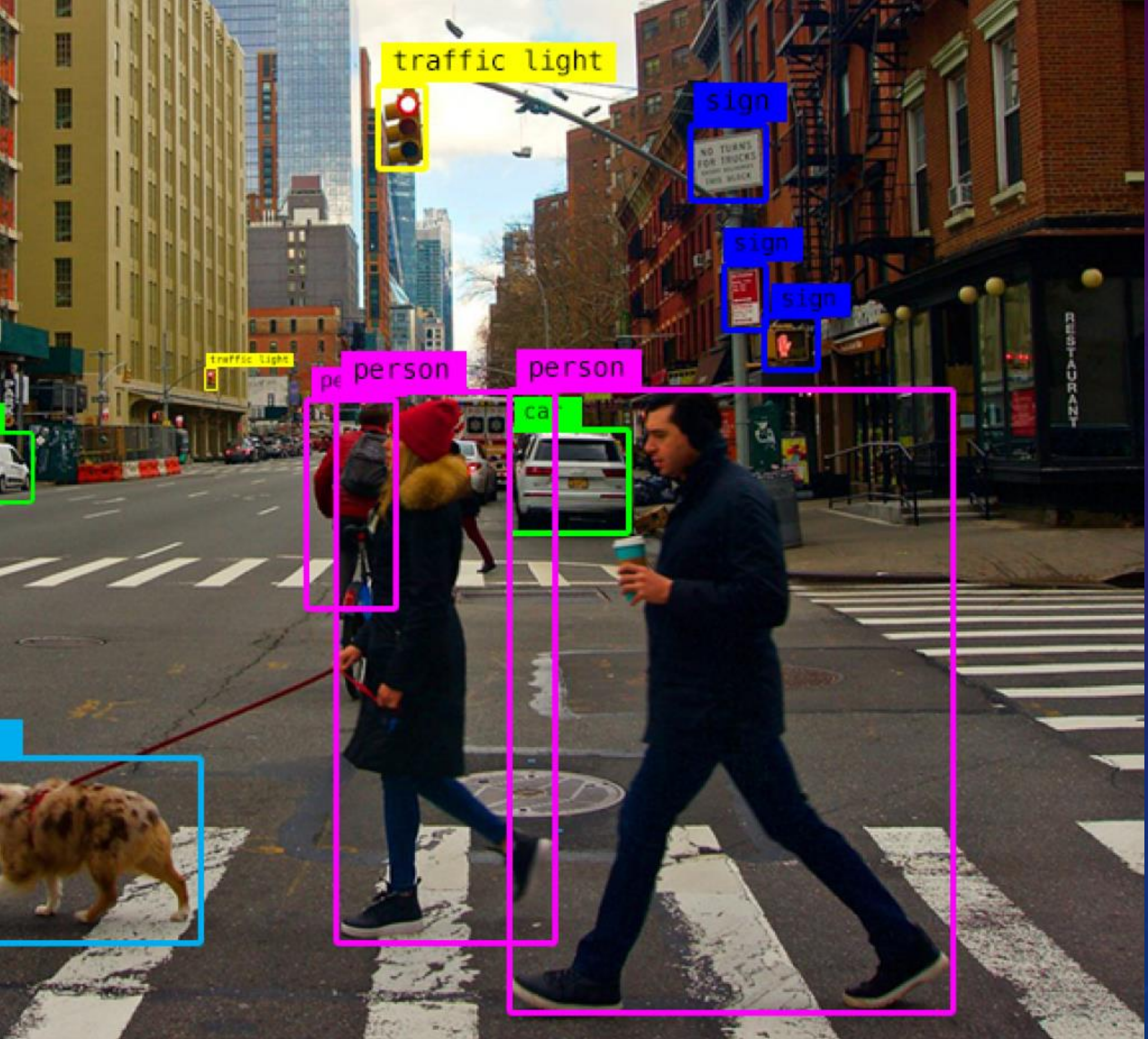
MOTION PLANNING



- **Motion Planning and Control Libraries:** You'll need libraries and tools for motion planning and control to simulate how the robot navigates and interacts with its environment. ROS provides various packages for these purposes, such as MoveIt! for motion planning.

VISION FOR ROBOTS

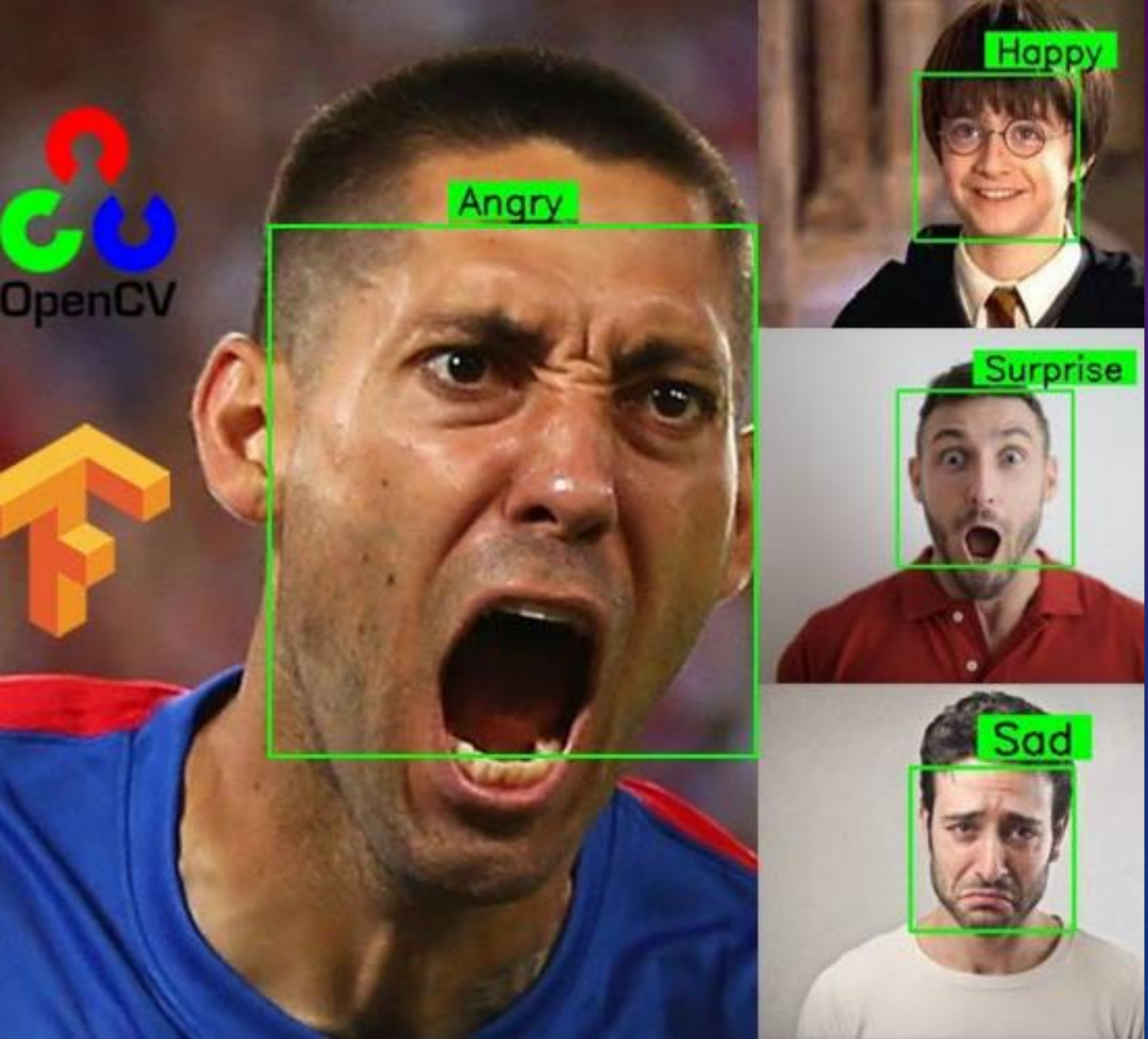
- Many applications involves computer vision for Robotics such as :
  - **Visual Perception:** Robots equipped with cameras can capture images or video of their environment. Computer vision algorithms are then used to process this visual data, extracting meaningful information about the surroundings. Visual perception enables robots to "see" and interpret the world in a way similar to how humans do.



OBSTACLE DETECTION AND AVOIDANCE

Object Detection and Recognition: One of the primary uses of cameras on robots is to detect and recognize objects. This includes identifying and classifying objects based on their shape, color, texture, or other visual features. Object detection and recognition are essential for tasks such as pick-and-place operations in manufacturing, autonomous navigation, and even assisting in search and rescue missions.

- Cameras on robots can be used to detect obstacles in their path. This is crucial for autonomous navigation, as robots can analyze the visual data to plan safe routes and avoid collisions. This capability is particularly important in applications like self-driving cars and warehouse robots.



GESTURE AND EMOTION RECOGNITION

- In human-robot interaction scenarios, cameras can be used to capture and interpret gestures, facial expressions, and emotions. This allows robots to respond to human cues and signals, making interactions more natural and intuitive.

TECHNOLOGIES USED IN VISION

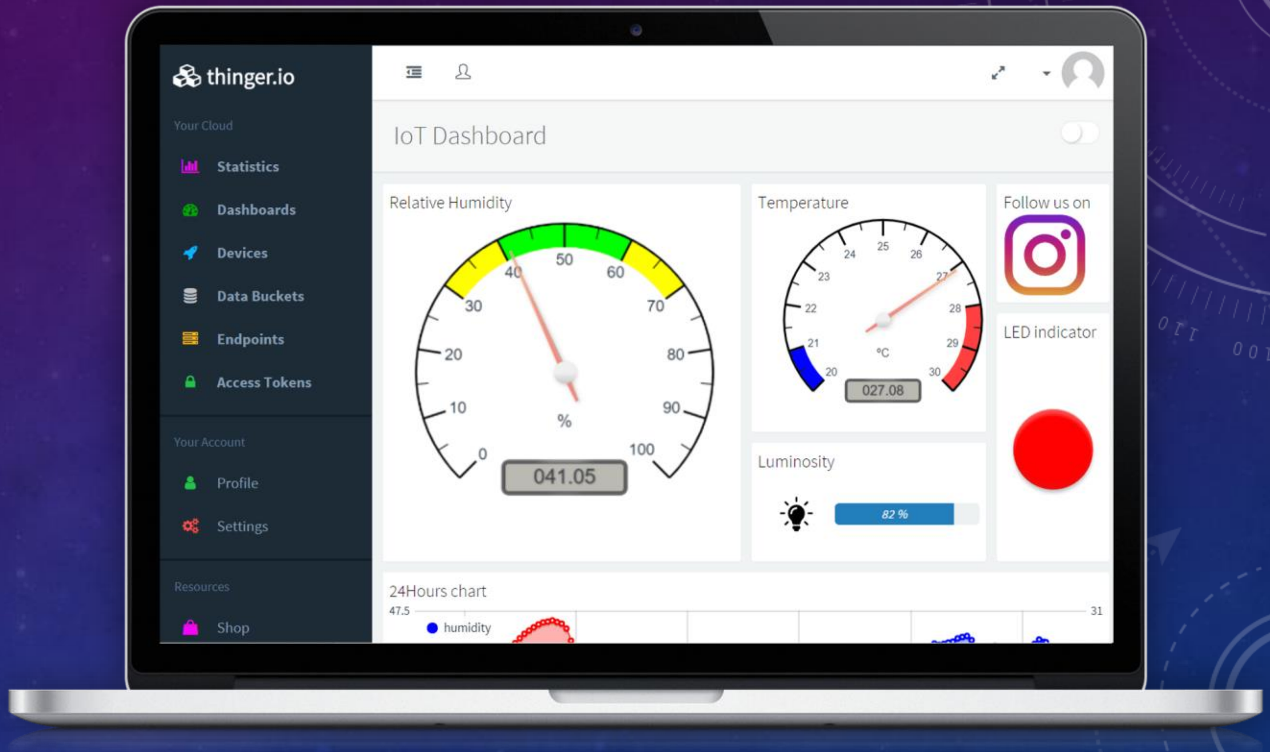
- OpenCV is a library for computer vision made by intel and its widely used nowadays in various and many Computer vision and Robotics Applications and it has many programming languages interfaces such as C++, Java and Python.
  C++ is used when you are targeting high-performance applications used in industry.

ROBOTS AND INTERNET OF THINGS

- IoT is used in Robotics to accomplish one or both of the following tasks :
    1) Control
    2) Monitor

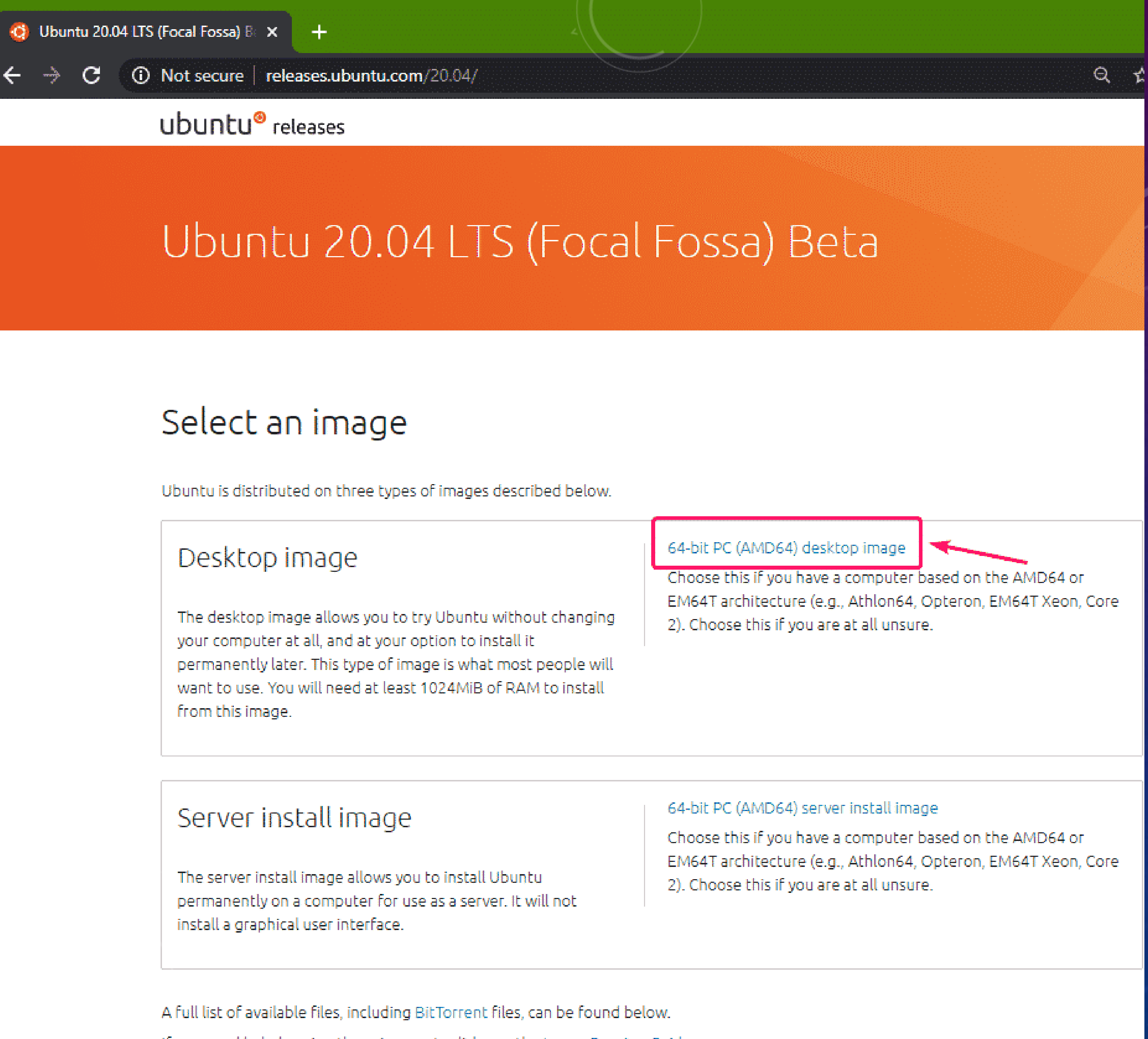
    Could be like giving a set point as a destination to reach for an autonomous mobile robot

    Or could be these AMR is equipped by sensors to monitor the environment for example a robot to monitor the weather status and sends sensor readings to a remote server.



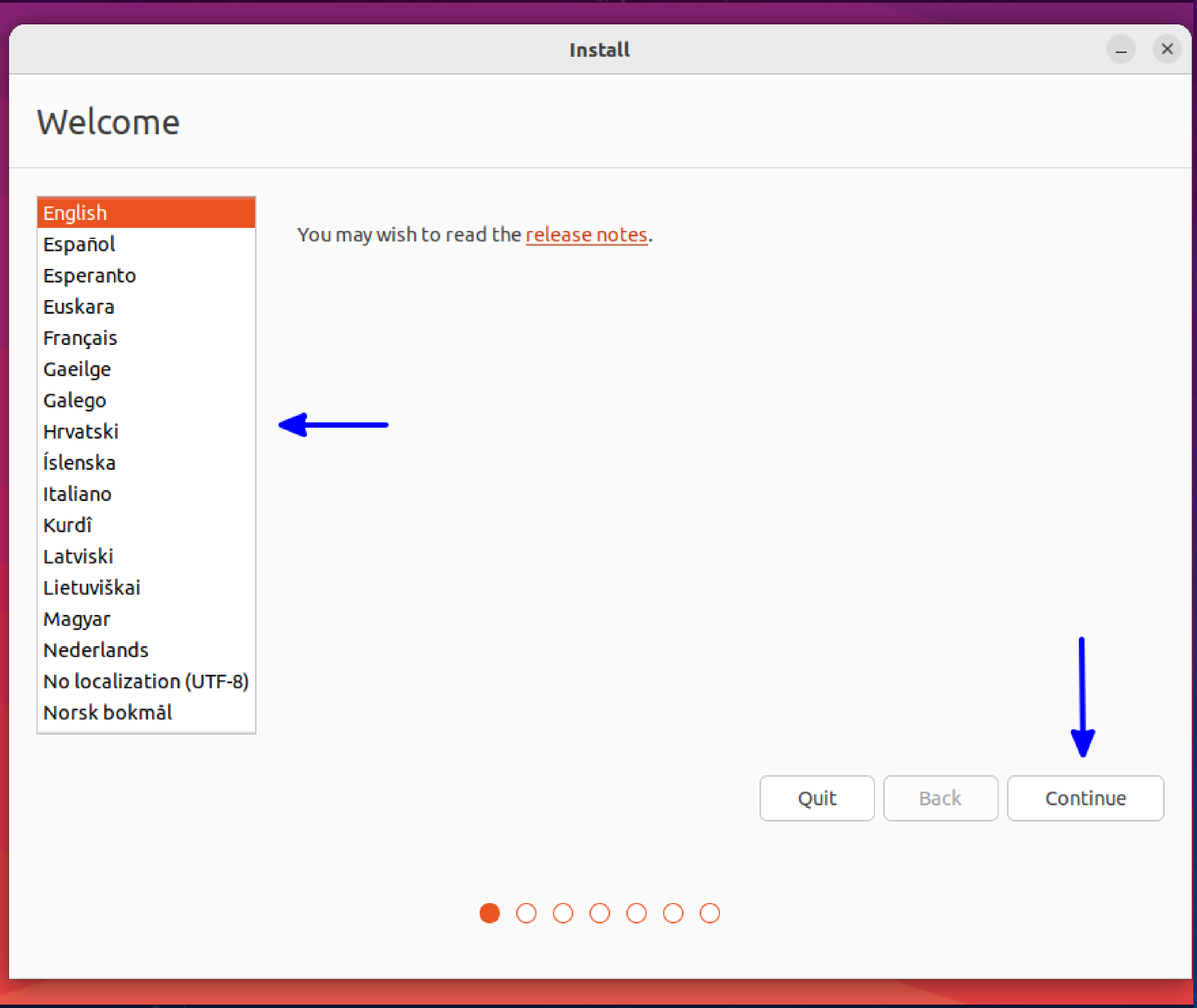
LET'S GET STARTED

INSTALL UBUNTU AND ROS



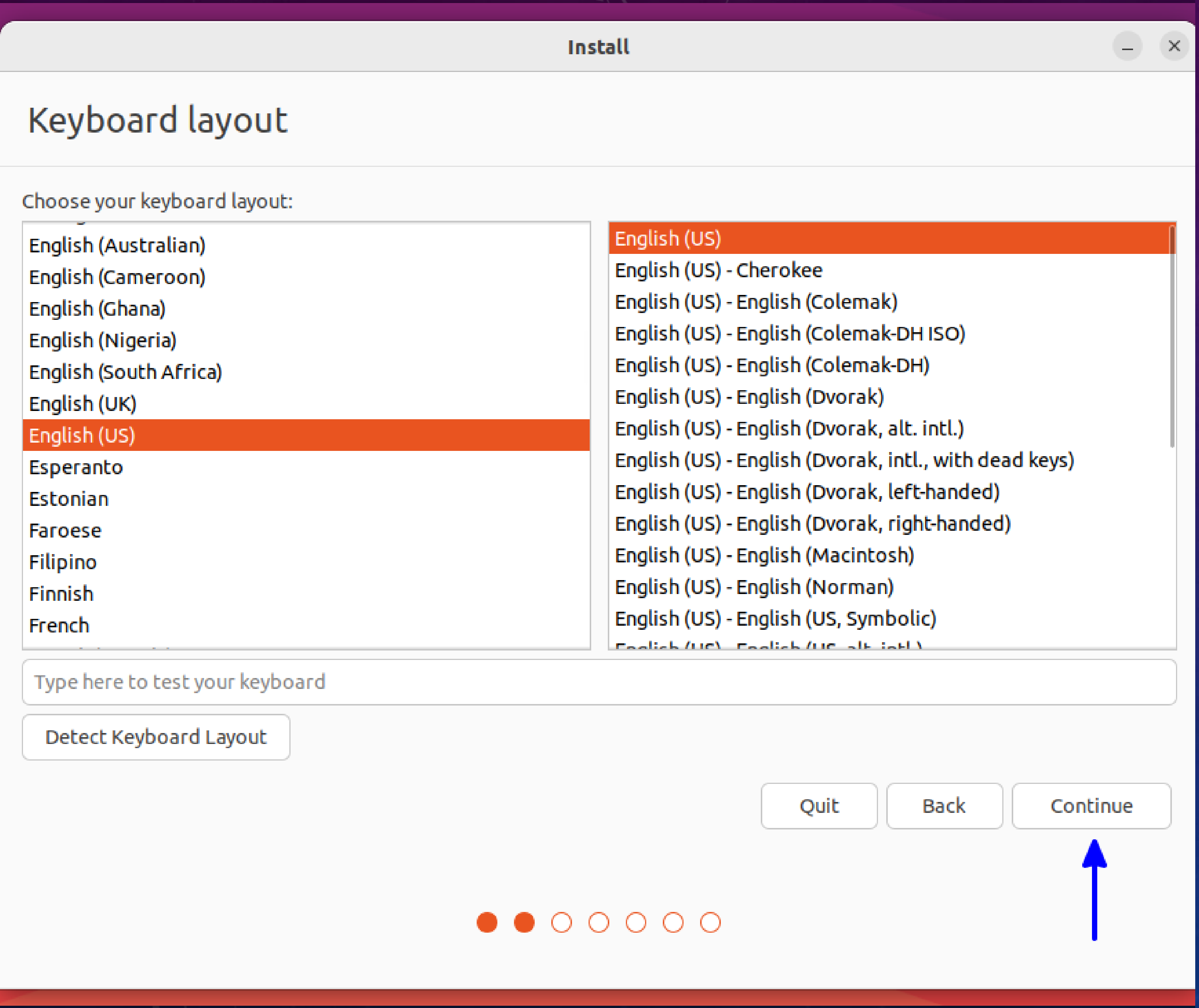
DOWNLOAD UBUNTU 20.04

SELECT THE 20.04 RELEASE



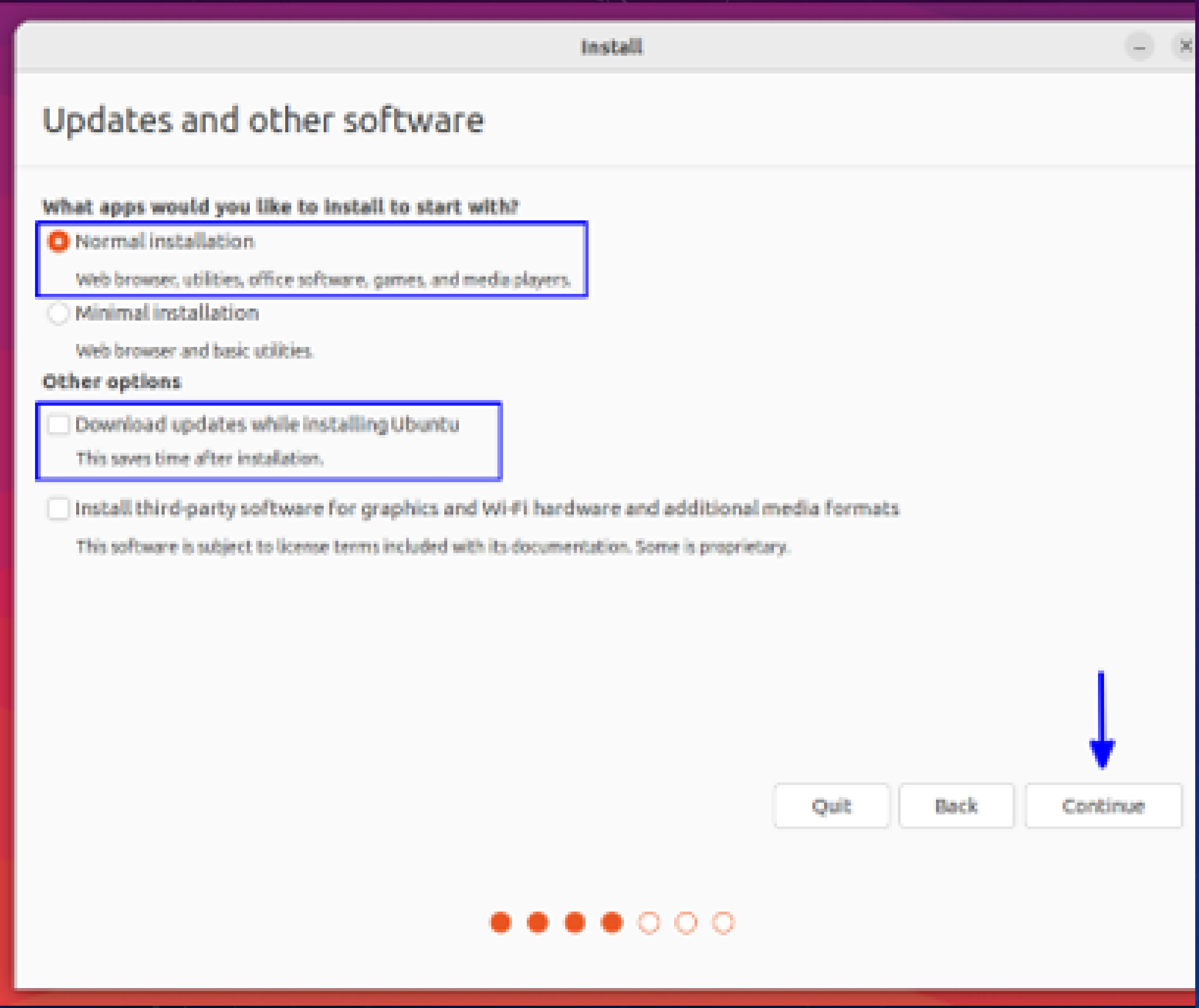
SELECT LANG.

- English then continue



SELECT LANGUAGE

- English US



NORMAL INSTALLATION

- Download updates while installing Ubuntu

SET UP YOUR SOURCES LIST

First, set up your computer to accept software from the ROS repository. Open a terminal and run the following commands:

sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

SET UP YOUR KEYS

- Next, add the ROS key to your system to authenticate the packages. Run these commands:
  - sudo apt update
  - sudo apt install curl
  - curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -

INSTALL ROS NOETIC

- sudo apt update
- sudo apt install ros-noetic-desktop-full

```
h2s@ubuntu:~$ sudo apt install ros-noetic-desktop-full
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libpipewire-0.2-1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  autopoint comerr-dev curl cython3 debhelper dh-autoreconf dh-strip-nondeterminism dwz fltk1.3-doc
  fluid fonts-lato freeglut3-dev gazebo11 gazebo11-common gazebo11-plugin-base gettext
  gir1.2-gtk-2.0 gir1.2-harfbuzz-0.0 hdf5-helpers ignition-tools intltool-debian krb5-multidev
  libaec-dev libarchive-cpio-perl libarchive-zip-perl libarmadillo-dev libarpack2-dev libass9
  libatk1.0-dev libavdevice-dev libavdevice58 libavfilter-dev libavfilter7 libblas-dev libblkid-dev
  libbs2b0 libbullet-dev libbullet2.88 libcairo-script-interpreter2 libcairo2-dev libccd-dev
  libccd2 libcfitsio-dev libcfitsio-doc libcharls-dev libcroco3 libcurl3-gnutls libcurl4
  libcurl4-openssl-dev libdap-dev libdapserver7v5 libdart-collision-bullet-dev
  libdart-collision-ode-dev libdart-dev libdart-external-ikfast-dev
  libdart-external-odelcpsolver-dev libdart-utils-dev libdart-utils-urdf-dev libdart6
  libdart6-collision-bullet libdart6-collision-ode libdart6-external-odelcpsolver libdart6-utils
  libdart6-utils-urdf libdatrie-dev libdebhelper-perl libepsilon-dev libfcl-dev libfcl0.5
  libffi-dev libfftw3-double3 libfile-stripnondeterminism-perl libflann-dev libflann1.9 libflite1
  libfltk-cairo1.3 libfltk-forms1.3 libfltk-gl1.3 libfltk-images1.3 libfltk1.3 libfltk1.3-dev
```

INITIALIZE ROS DEPENDENCIES

You'll need to initialize rosdep, which is a tool for managing package dependencies.

sudo rosdep init

rosdep update

ENVIRONMENT SETUP

```
echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
```

```
source ~/.bashrc
```

ROBOTICS CORNER

THANK YOU