CUKUROVA UNIVERSITY

FACULTY OF ENGINEERING-COMPUTER ENGINEERING

COURSE: INTRODUCTION TO AUTONOMOUS ROBOT

Autonomous Navigation Robot Simulation Using ROS and Gazebo

PROJECT MEMBERS:

ALİ TEPELİ-2020555060

SEMİH ZENGİNOĞLU-2020555072

ADVISER:

Assistant Professor Dr. Barış Ata

DELIVERY DATE:

27.12.2024

INTRODUCTION:

In this project, we used the ROS and Gazebo simulation environments to enable the TurtleBot-3 robot to move autonomously in an unknown environment. The primary goal of the project is to ensure that the robot can detect obstacles around it and avoid them, while also being able to create a map of the environment. During this process, the data from the robot's sensors was processed to identify objects and obstacles in the surroundings.

The robot used a LIDAR sensor to perceive its environment in real time. The data from these sensors was processed using ROS-based algorithms to make movement decisions. The Gazebo simulation environment provided a platform for testing and developing the robot.

Furthermore, as part of the project, the robot gained the ability to create a map of the environment while avoiding obstacles. As a result, the robot was able to move safely and successfully generate a model of the surrounding area.

YOUTUBE VIDEO LINKS:

Ali TEPELİ 2020555060: https://youtu.be/1s9mx2ytxJs

Semih ZENGİNOĞLU 2020555072: https://youtu.be/pXRXFIPQhII

CONSTRUCTION PHASE:

At the beginning of the project, we first installed Ubuntu 20.04. Then, we downloaded the Noetic version of ROS, which is compatible with Ubuntu 20.04. We are sharing the commands we used to do this:

1.3 Set up your keys

```
sudo apt install curl # if you haven't already installed curl
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

1.4 Installation

First, make sure your Debian package index is up-to-date:

```
sudo apt update
```

Now pick how much of ROS you would like to install.

Desktop-Full Install: (Recommended) : Everything in **Desktop** plus 2D/3D simulators and 2D/3D perception packages

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

or click here

Desktop Install: Everything in **ROS-Base** plus tools like rqt and rviz

```
sudo apt install ros-noetic-desktop
```

or click here

ROS-Base: (Bare Bones) ROS packaging, build, and communication libraries. No GUI tools.

```
sudo apt install ros-noetic-ros-base
```

or click here

There are even more packages available in ROS. You can always install a specific package directly.

```
sudo apt install ros-noetic-PACKAGE
```

e.g.

```
sudo apt install ros-noetic-slam-gmapping
```

To find available packages, see ROS Index or use:

Later, we designed a world in the Gazebo environment where the robot will move. We tried to create an office environment. In this design, there are obstacles that the robot will encounter while autonomously moving to the desired point. To design this world, we downloaded the objects we will use from GitHub using the command shown:

README GPL-3.0 license

Gazebo models and worlds collection

license GPLv3

This repository contains models and worlds files for Gazebo, which are collected from several public projects.

Usage

To use the models, the `models` directory needs to be added to the `GAZEBO_MODEL_PATH` environment variable. To do so, add the following line to the end of `~/.bashrc` :

```
export GAZEBO_MODEL_PATH=$GAZEBO_MODEL_PATH:<path to this repo>/models
```

It is also possible to add model path in the Gazebo Client GUI.

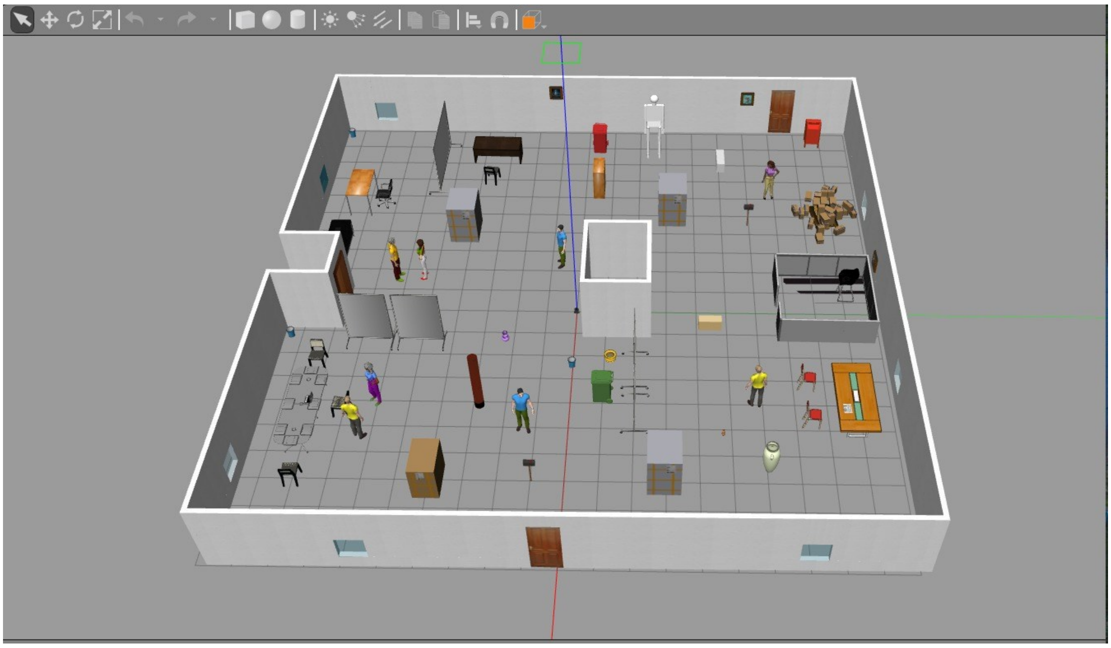
Adding `worlds` directory to the `GAZEBO_RESOURCE_PATH` environment variable allows you to specify the world without absolute path. To do so, add the following line to `~/.bashrc` :

```
export GAZEBO_RESOURCE_PATH=$GAZEBO_RESOURCE_PATH:<path to this repo>/worlds
```

Preview

Gazebo screenshots are provided in `screenshots` to preview the worlds.

The environment we created is as follows:

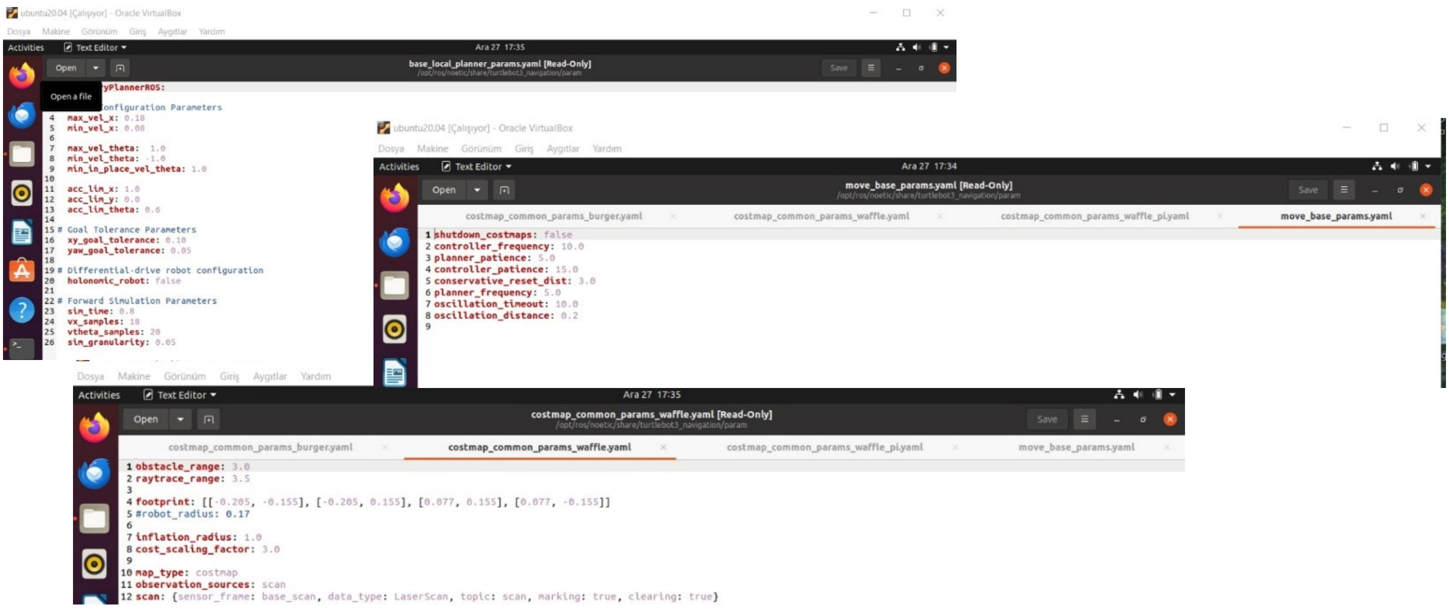


In this environment, we are sharing the robot we will use, along with the packages that enable the robot to move autonomously using navigation, and some of the contents of these packages:

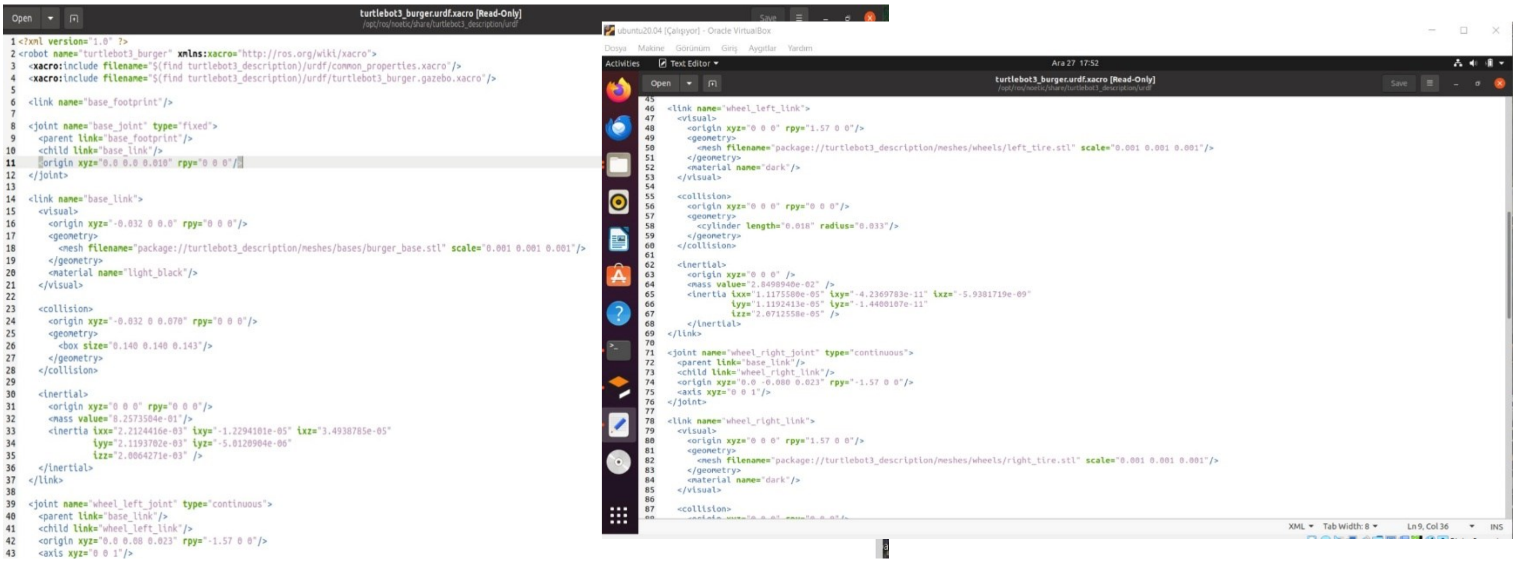
bash Kodu kopyala

```bash
sudo apt install ros-noetic-turtlebot3-simulations ros-noetic-turtlebot3-navigation
```

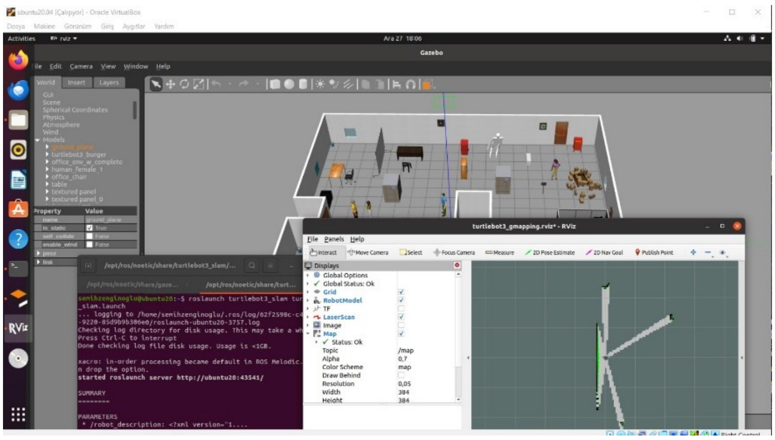
```
export TURTLEBOT3_MODEL=burger
roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch
```



The TurtleBot-3 robot uses the LIDAR sensor as part of its functionality while performing autonomous movement. Some of the code it uses for the LIDAR sensor is as follows:



In the final stage, we used the RViz tool to observe the robot's movement in the world we created. This tool allowed us to visualize the data from the robot's LIDAR sensor in 2D. While doing this, we used specific commands. First, we ensured that the robot only mapped its starting position using SLAM. The commands we used are as follows:



-- roslaunch turtlebot3_navigation turtlebot3_navigation.launch map_file:=/home/semihzenginoglu/ofis1_map.yaml

-- rosrun map_server map_saver -f ~/ofis1_map

Then, by using certain commands, we enabled the robot to load into our world, display the data from the LIDAR sensor, and observe the robot's movement in RViz. The sequence of commands we used is as follows:

```
alitpl01@DESKTOP-RV9KI3J:~$ roslaunch gazebo_ros empty_world.launch world_name:=/home/alitpl01/ofis1.world
```

We opened our world with the above command.

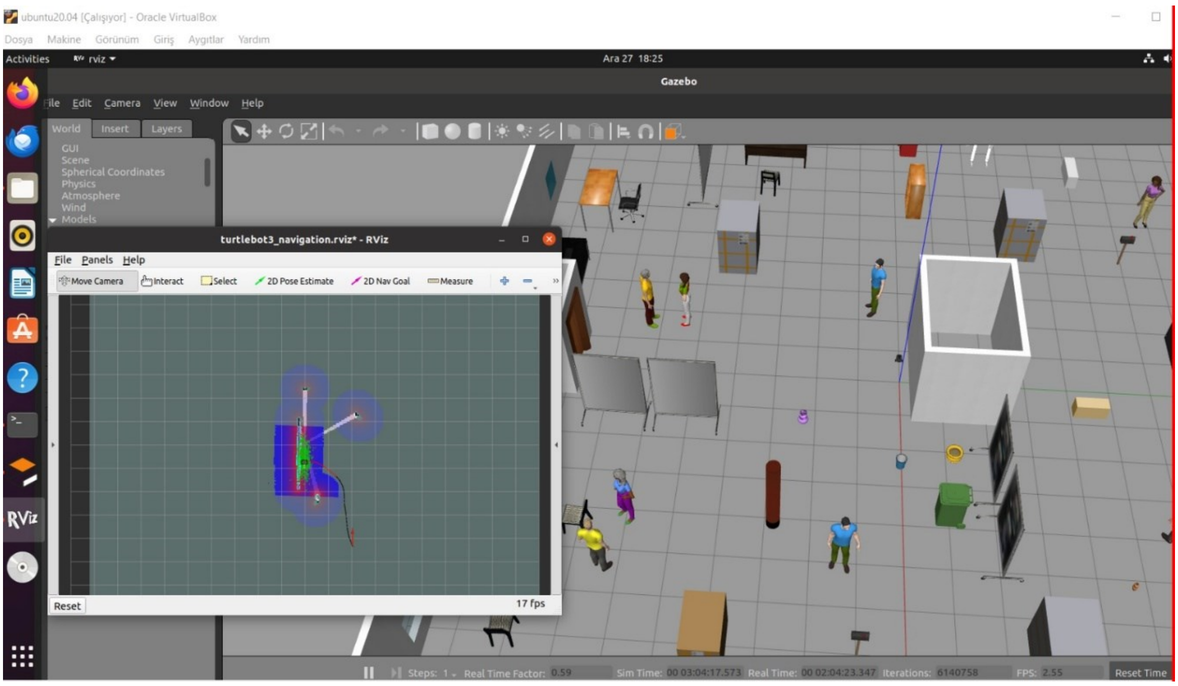
```
alitpl01@DESKTOP-RV9KI3J:~$ roslaunch turtlebot3_navigation turtlebot3_navigation.launch map_file:=/home/alitpl01/ofis1_map.yaml
```

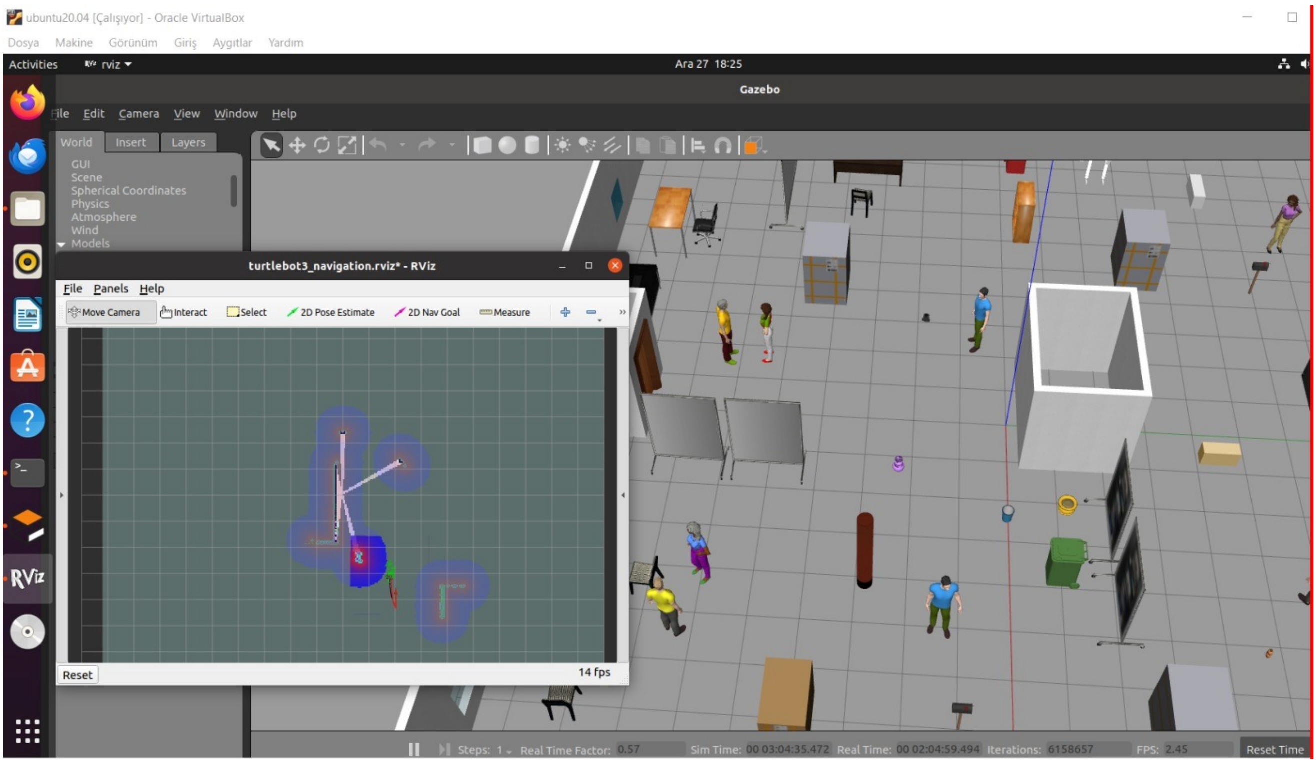
With this command, we opened the robot's starting position in RViz as we mapped it and displayed the robot's location there. To make the robot move to the desired location, we used the 2D Nav Goal tool and observed the robot.

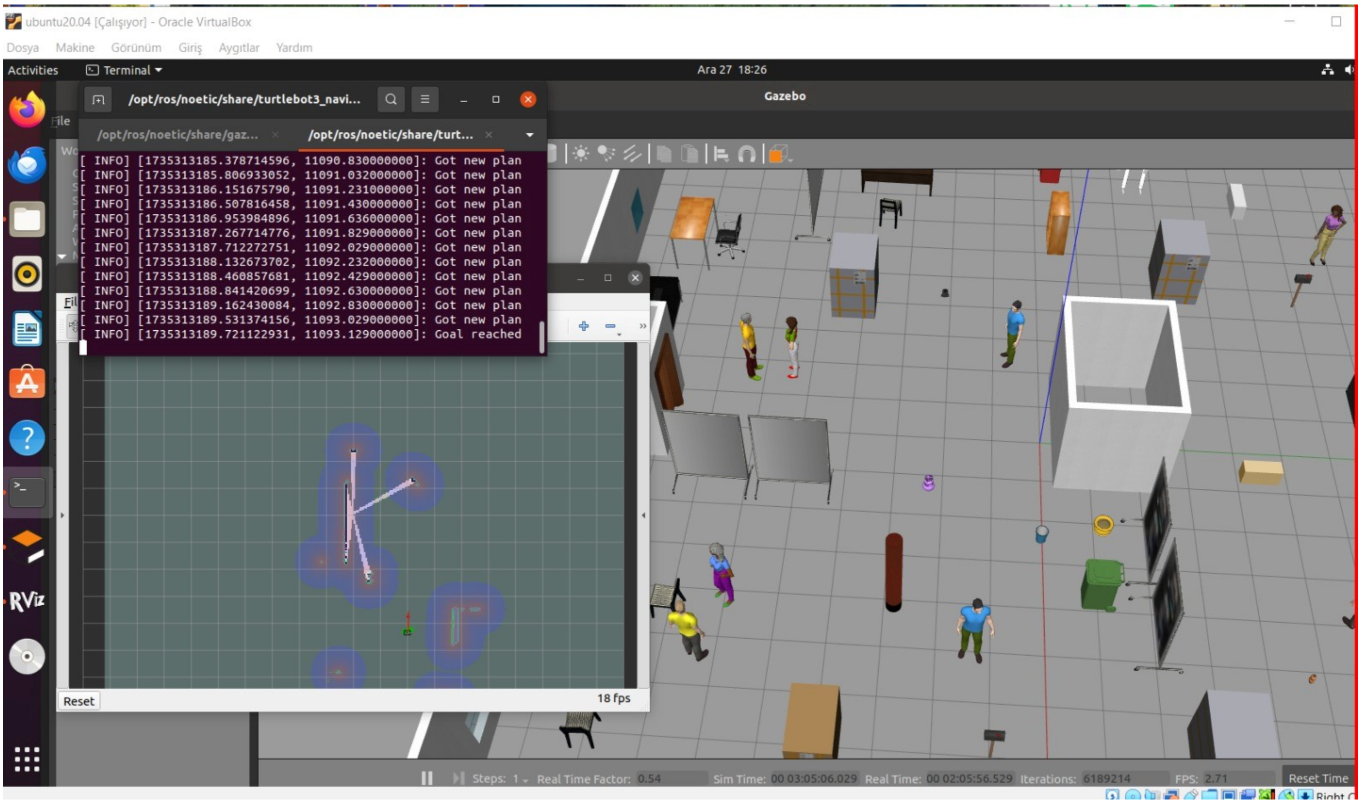
```
alitpl01@DESKTOP-RV9KI3J:~$ rostopic echo /scan
```

The above command allows us to display the data from the robot's LIDAR sensor in the command line.

Finally, I share the stages of the robot's movements while moving to a certain point in RVIZ.







The outline and process of our project was like this.

AS A RESULT:

In this project, we learned how to design a simulation of an autonomous robot. We learned what we need and the tools we must use to achieve autonomous movement. We explored the usage of Ubuntu and learned how to design an environment for the autonomous robot system using Gazebo. We also learned the use and working principles of the LIDAR sensor in autonomous movement, and the navigation algorithms (A*, Dijkstra, etc.) the robot uses to avoid obstacles. Then, we learned how to perform mapping using SLAM technology. Finally, we visualized the robot's movement paths and the data received from the LIDAR sensor using RViz.

REFERENCES:

- ROS.org- https://wiki.ros.org/noetic/Installation/Ubuntu

- ChatGPT

- for Gazebo models;
  https://github.com/leonhartyao/gazebo_models_worlds_collection
  https://github.com/osrf/gazebo_models

- Mecharithm - Robotics and Mechatronics (YouTube)
  https://youtube.com/playlist?list=PLlqdnFs9xNwql5KET7v7zyl393y10qxtw&si=hcC7VA9DWv3nZl1E