الجامعة المصرية اليابانية للعلوم والتكنولوجيا
E-JUST
Egypt - Japan University of Science and Technology
エジプト日本科学技術大学

MTE - 431

Mobile Robots

Group Course Project Report

Submitted by:

Mohamed Alaa Abd EL-Salam	120200195
Mohab EL-Deen Yasser	120200019
Mohamed Maher Gamal	120200042
Abdelrahman Ahmed	120200125
Youssef Ahmed	120200025

To:

Dr. Haitham El-Hussieny

I. Introduction:

This project delves into the development of a mobile robot, offering the choice between a four-wheeled Mecanum configuration or a three-wheeled Swedish 90-degree setup. The focus is on understanding the kinematics, constructing a Unified Robot Description Format (URDF) file, and implementing control systems through ROS2 and Python. Participants navigated through phases involving mechanical assembly, hardware integration, and software development, gaining insights into the field of mobile robotics.

II. Objectives:

The objectives of this project are centered around achieving a comprehensive understanding of various aspects crucial to the development of an omnidirectional mobile robot. Each objective plays a distinct role in shaping the final outcome:

1. **Kinematic Analysis:**
    - Making the required analysis of the chosen wheel configuration's kinematics.
    - Exploring the motion dynamics, including the translation and rotation of the robot.
    - Finally, getting out the required transformation matrix that transforms the required rotational and linear speed into every wheel’s rotational speed.

2. **URDF Creation:**
    - Constructing a URDF file for the robot to define it geometrically in the simulation part, which Facilitate simulation and control within the Robot Operating System (ROS2).

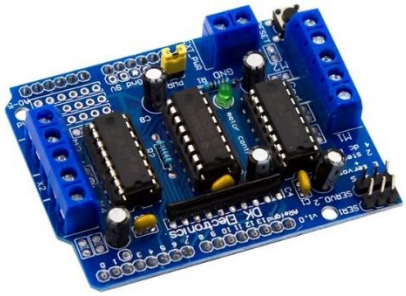
3. **Control System Development:**
    - Developing a robust control system that establishes communication with the robot.
    - Interfacing with an Arduino Uno, utilizing a USB connection to send motor commands.

III. Hardware Requirements:

1. **Mecanum Wheels (Four):**
    - **Role:** Enable omnidirectional movement by allowing the robot to move forward, backward, and rotate seamlessly.

2. **Geared DC Motors (Four):**
    - **Role:** Drive the Mecanum wheels, translating electrical signals into mechanical motion.

3. **L293D Motor Driver:**
    - **Role:** Acts as an interface between the Arduino Uno and the four DC motors, facilitating motor control and direction for the four wheels at the same time.



4. **Arduino Uno:**
    - **Role:** Serves as the brain of the robot, processing inputs and generating motor control signals.

5. **Batteries:**
    - **Role:** Power source for the entire system, including motors and the Arduino Uno.

IV. Project Phases:

- ## Phase 1-Design and Kinematic Analysis:

We chose to use the four mecanum wheels configuration and we used the dimensions and sketches to estimate the parameters needed to obtain the kinematic model for our robot, after that we used a python script that took those parameters as input and transformed them into the kinematic model, below is a snippet from that python code and the code is included in the submitted files.

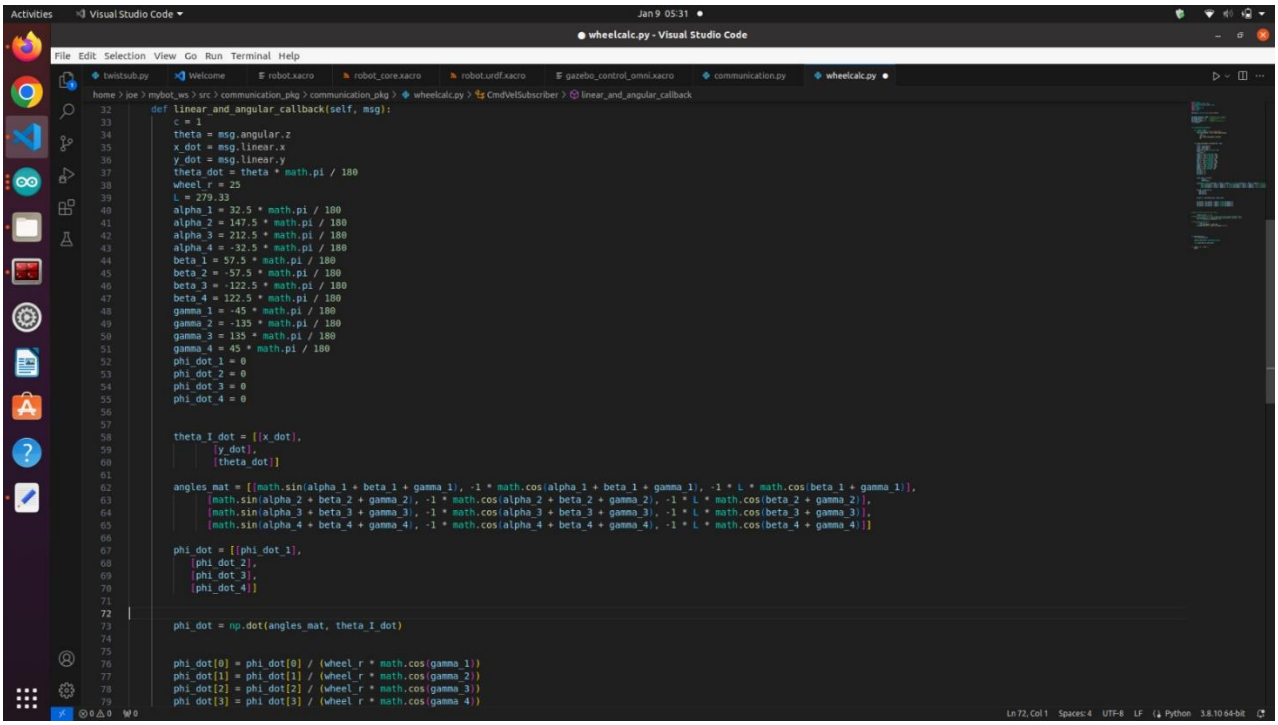


Figure 1 (Snippet from the kinematic analysis python script)

- **<u>Phase 2 – Building The Robot:</u>**

We used all the mentioned parts above to assemble and build the robot physically, we mounted the mecanum wheels on motors and connected them to the motor driver which is connected to Arduino Uno, in addition to the batteries which act as the power source.

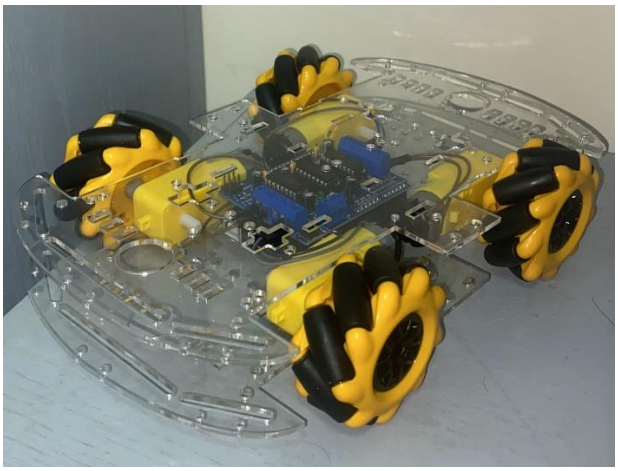


Figure 2(The kit after assembly)

- **Phase 3 – Software Development:**

During this phase, we are focusing on the creation of the URDF file and the implementation of control logic using ROS2 and Python. A dedicated folder was assembled, including the following xacro files, each intricately defining specific aspects of the robot's structure and characteristics:

1. **Base.xacro:**
   - **Role:** Establishes the foundational structure of the robot.
   - **Features:**
     - Dimensions and geometry of the robot's base.
     - Inertial properties such as mass, center of mass, and moments of inertia.
     - Visual and collision properties for accurate representation and interaction in simulations.
2. **Lidar.xacro:**
   - **Role:** Defines the properties and position of the LiDAR sensor on the robot.
   - **Features:**
     - LiDAR sensor specifications, including field of view and range.
     - Placement coordinates and orientation on the robot's structure.
3. **Motor.xacro:**
   - **Role:** Describes the geared DC motors that drive the Mecanum wheels.
   - **Features:**
     - Motor specifications: torque, maximum speed, and power rating.
     - Inertial properties to accurately model motor behavior in simulations.
4. **Pillar.xacro:**
   - **Role:** Specifies the structural elements or pillars of the robot.
   - **Features:**
     - Dimensions and geometry of the pillars.
     - Inertial properties for accurate representation in simulations.
5. **Robot.urdf.xacro:**
   - **Role:** Main assembly file linking all components together to create the complete robot model.
   - **Features:**
     - Inclusion of **Base.xacro**, **Lidar.xacro**, **Motor.xacro**, **Pillar.xacro**, and **Wheel.xacro**.
     - Joint definitions to the moving parts.
     - Inertial properties for the entire robot assembly.

6. **Wheel.xacro:**
    - **Role:** Describes individual Mecanum wheels in terms of Wheel specifications: radius, width, and number of rollers, inertial properties for accurate representation in simulations.
7. **Gazebo_control_omni.xacro:**
    - **Role:** Specifies control parameters for the robot within the Gazebo simulation environment.

These files are intricately interconnected through the **Robot.urdf.xacro**, which acts as the central unit for the assembly.

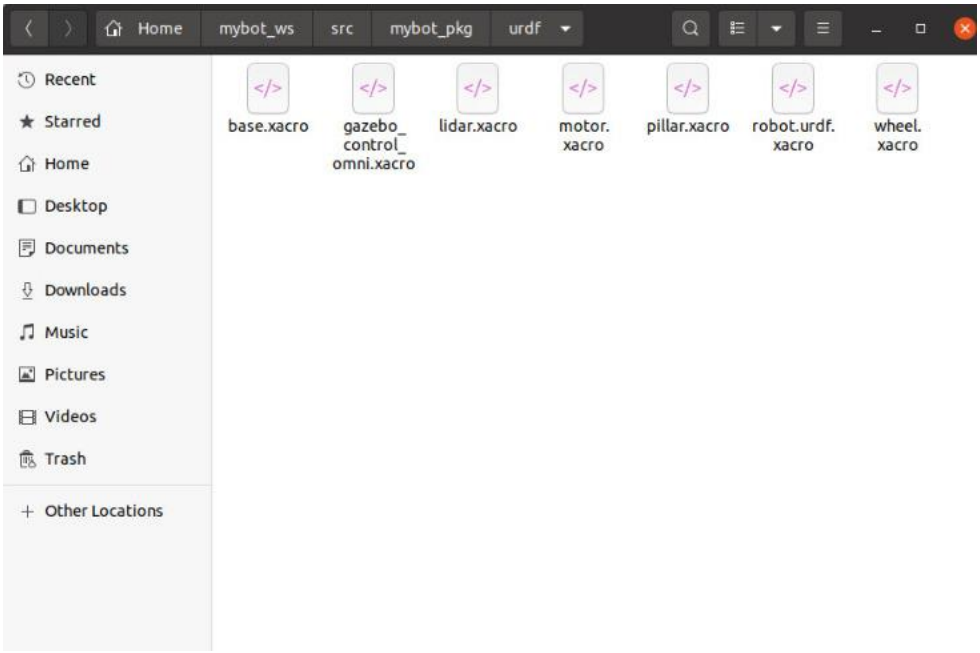


Figure 3 (The folder containing all Xacro files)

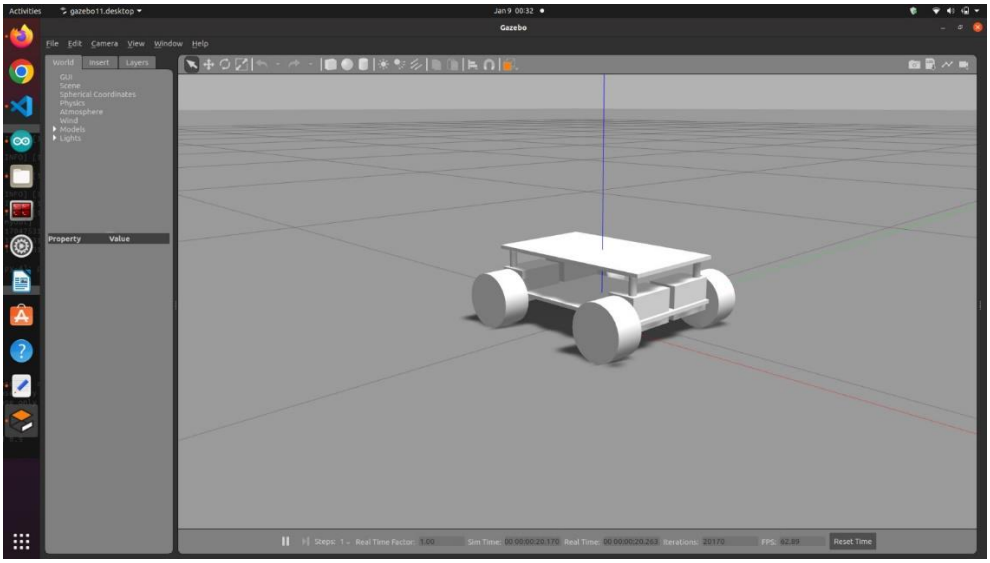


Figure 4 (Simulation of the robot in Gazebo)

In the software development phase, the Python code plays the important role in transformingvelocity commands received on the **cmd_vel** topic into motor control signals:

- **Serial Communication Setup:**
  The code initiates serial communication with the Arduino Uno, providing a connection for transmitting motor speed information.

```
SerialObj = serial.Serial('/dev/ttyACM0')
```

- **Control Logic Node:**
  - A ROS2 node named **CmdVelSubscriber** is created to subscribe to the **cmd_vel** topic, where velocity commands are published.

```
class CmdVelSubscriber(Node):

    def __init__(self):
        super().__init__('cmd_vel_subscriber')
        self.subscription = self.create_subscription(
            Twist,
            'cmd_vel',
            self.linear_and_angular_callback,
            10)
```

- **Kinematic Calculations:**
  - The received linear and angular velocity commands (**x_dot**, **y_dot**, **theta_dot**) are transformed into wheel speeds (**phi_dot**) using a kinematic model based on the Mecanum wheel configuration.

```
phi_dot[0] = phi_dot[0] / (wheel_r * math.cos(gamma_1))
phi_dot[1] = phi_dot[1] / (wheel_r * math.cos(gamma_2))
phi_dot[2] = phi_dot[2] / (wheel_r * math.cos(gamma_3))
phi_dot[3] = phi_dot[3] / (wheel_r * math.cos(gamma_4))
```

- **Serial Transmission to Arduino:**
  - The normalized wheel speeds are then converted to a string format and transmitted serially to the Arduino Uno.

```
# Define a format string for the floats

        format_string = "{:.2f}"
# Convert the column vector into a single line string with formatted floats
        phi_dot_string = '\n'.join(map(format_string.format, phi_dot[:, 0]))
        li = list(phi_dot_string.split(" "))

# Display the column vector
        for item in li:
            self.get_logger().info(item)
            SerialObj.write(item.encode())
```

- **Arduino Code:**
  - In this code, the received wheel speeds that are then transformed and corrected into functional values that are then sent to the motor driver to move the wheels with the right direction and speed to obtain the required robot movement.

V. Conclusion:

This Mobile Robots course project sums up so many aspects in the design and implementation of an imnidirectional mobile robot using mecanum wheels, Arduino and ROS2, it focused on the following aspects specifically:

1. **Kinematic Understanding:**
    - A detailed analysis of the Mecanum wheel configuration's kinematics which led for precise control and maneuverability.
2. **URDF Development:**
    - Creation of the URDF file to make simulation and control in ROS2, providing a detailed representation of the robot.
3. **Software communication:**
    - The Python code translated high-level velocity commands into specific wheel speeds, demonstrating effective communication with the Arduino Uno.
4. **Hardware Integration:**
    - The integration of Mecanum wheels, geared DC motors, a motor driver, and the Arduino Uno resulted in a fully functional mobile robot.
5. **Testing and Calibration:**
    - Rigorous testing and calibration phases enhanced the robot's robustness, ensuring smooth and accurate performance.

In conclusion, this project delivers a functional mobile robot while imparting practical knowledge and skills crucial for future projects in mobile robotics.