Laporan Hasil Tutorial
Robot Operating System (ROS)

1. Pendahuluan

Robot Operating System (ROS) adalah framework perangkat lunak open-source yang menyediakan berbagai alat dan pustaka untuk mengembangkan perangkat lunak robotik. Meskipun namanya mengandung kata *operating system*, ROS lebih tepat dianggap sebagai *middleware* yang memfasilitasi komunikasi antar perangkat keras dan perangkat lunak dalam pengembangan aplikasi robotik. ROS dirancang untuk membantu pengembang dalam menciptakan aplikasi robot yang lebih modular, terstruktur, dan dapat diperluas.

Tujuan dari pembuatan tutorial ini adalah untuk memberikan panduan praktis bagi pemula dalam menginstal dan menjalankan ROS di sistem operasi berbasis Linux, serta memahami konsep-konsep dasar seperti pembuatan dan komunikasi antar node, penggunaan sensor, serta eksekusi perintah dasar dalam ROS.

2. Persiapan

Sebelum memulai tutorial ini, ada beberapa alat, perangkat lunak, dan sumber daya yang perlu dipersiapkan:

Alat dan Perangkat Keras:

1. Komputer atau Laptop yang terhubung ke internet.
2. Sistem operasi Linux (disarankan menggunakan Ubuntu 20.04 atau lebih baru). ROS 2 dan ROS 1 didukung di berbagai versi Linux, tetapi Ubuntu adalah sistem operasi yang paling kompatibel.
3. Perangkat keras robot (opsional) jika tutorial ini berfokus pada pengembangan perangkat robot nyata. Namun, untuk tutorial ini, kita akan menggunakan simulasi robot.

Perangkat Lunak yang Dibutuhkan:

1. Ubuntu – Sistem operasi berbasis Linux.
2. ROS 1 (dalam tutorial ini, kita akan menggunakan ROS 1) – versi ROS yang lebih stabil, yang lebih kuat, aman, dan mendukung fungsionalitas real-time.
3. Git – Untuk mengunduh kode dari repositori ROS.
4. Visual Studio Code atau Editor Teks Lain – Untuk menulis dan mengedit kode.
5. Gazebo (opsional) – Untuk simulasi robot. Gazebo digunakan untuk menjalankan simulasi robot dan lingkungan 3D.

Resources:

1. Dokumentasi ROS – https://wiki.ros.org/noetic
2. Tutorial ROS – https://wiki.ros.org/ROS/Tutorials
3. Forum ROS – Untuk diskusi dan solusi masalah yang dihadapi selama pengembangan.

3. Langkah Implementasi

Pada bagian ini, kita akan menjelaskan langkah-langkah yang diambil selama pembuatan tutorial, mulai dari instalasi hingga eksekusi.

3.1 Instalasi ROS

Langkah pertama adalah menginstal ROS pada sistem yang menjalankan Ubuntu. Berikut adalah tahapan instalasi ROS (versi Noetic) di Ubuntu 20.04:

1. **Update Paket Sistem**
   Pastikan sistem operasi Ubuntu diperbarui dengan menjalankan perintah berikut:
   ```
   sudo apt update
   sudo apt upgrade
   ```

2. **Tambahkan Repositori ROS**
   Tambahkan repositori yang dibutuhkan dengan perintah berikut:
   ```
   sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
   ```

3. **Menambahkan Kunci GPG**
   Tambahkan kunci GPG untuk memverifikasi paket ROS dengan perintah berikut:
   ```
   sudo apt install curl # jika curl belum terinstall
   curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
   ```

4. **Instalasi ROS**
   Update lagi daftar paket dan instal ROS 2 dengan perintah berikut:
   ```
   sudo apt update
   sudo apt install ros-noetic-desktop-full
   ```

5. **Setup Lingkungan ROS**
   Setelah instalasi selesai, kita perlu mengatur lingkungan ROS dengan menambahkan baris berikut ke file `.bashrc`:
   ```
   source /opt/ros/noetic/setup.bash
   ```

   Kemudian, terapkan perubahan dengan menjalankan perintah berikut:
   ```
   source ~/.bashrc
   ```

6. **Instalasi Dependensi**
   ROS membutuhkan beberapa dependensi tambahan, jadi pastikan untuk menginstal `rosdep` yang akan membantu mengelola dependensi dengan lebih mudah:
   ```
   sudo apt install python3-rosdep
   sudo rosdep init
   rosdep update
   ```

3.2 Membuat Workspace dan Menulis Node ROS

Langkah berikutnya adalah membuat workspace dan menulis node pada ROS:

1. **Membuat Workspace ROS**
   Workspace adalah folder tempat mengelola paket-paket ROS. Buatlah workspace ROS menggunakan perintah berikut:
   ```
   mkdir -p ~/ros_ws/src
   cd ~/ros_ws/
   colcon build
   ```

2. **Membuat Paket ROS**
   Setelah workspace berhasil dibuat, buat paket baru menggunakan perintah berikut:
   ```
   cd ~/ros_ws/src
   ros pkg create --build-type ament_cmake my_first_package
   ```

3. **Membuat Node ROS**
   Di dalam folder paket, buat file Python untuk node pertama. Misalnya, buat file `talker.py` yang berfungsi untuk mengirimkan pesan secara berkala dengan perintah berikut:
   ```
   import rclpy
   from rclpy.node import Node
   from std_msgs.msg import String

   class MinimalPublisher(Node):
      def __init__(self):
         super().__init__('minimal_publisher')
         self.publisher_ = self.create_publisher(String, 'topic', 10)
         timer_period = 2
         self.timer = self.create_timer(timer_period, self.timer_callback)

      def timer_callback(self):
         msg = String()
         msg.data = 'Hello, ROS 2!'
         self.publisher_.publish(msg)
         self.get_logger().info('Publishing: "%s"' % msg.data)

   def main(args=None):
      rclpy.init(args=args)
      minimal_publisher = MinimalPublisher()
      rclpy.spin(minimal_publisher)
      minimal_publisher.destroy_node()
      rclpy.shutdown()

   if __name__ == '__main__':
      main()
   ```

4. **Membangun dan Menjalankan Node**
   Setelah kode selesai, bangun dan jalankan node menggunakan perintah berikut:
   ```
   cd ~/ros_ws
   colcon build
   source install/setup.bash
   ros run my_first_package talker
   ```

3.3 Menggunakan Gazebo untuk Simulasi
Jika kita ingin menambahkan simulasi robot, instal dan konfigurasi Gazebo :

1. **Instal Gazebo**
   Lakukan instalasi Gazebo dengan perintah berikut:
   ```
   sudo apt install gazebo11
   ```

2. **Jalankan simulasi robot**
   Kita bisa menggunakan contoh paket robot yang sudah ada di ROS untuk melakukan simulasi, misalnya dengan perintah:
   ```
   ros launch gazebo_ros empty_world.launch.py
   ```

4. Hasil

Setelah mengikuti tutorial ini, kita akan memiliki lingkungan ROS yang terinstal dengan benar di komputer. Kita juga akan dapat membuat dan menjalankan *node* pertama yang mengirimkan pesan ke topik tertentu. Selain itu, dengan menggunakan Gazebo, kita dapat melihat dan berinteraksi dengan robot dalam lingkungan simulasi 3D.

Hasil akhir tutorial ini sesuai dengan yang diharapkan: ROS terinstal dan berfungsi dengan baik, node yang mengirimkan pesan berhasil dijalankan, dan simulasi robot dapat dioperasikan menggunakan Gazebo.

5. Kesimpulan

Dalam tutorial ini, kita telah belajar bagaimana menginstal dan mengkonfigurasi Robot Operating System (ROS) pada sistem operasi Ubuntu, serta membuat dan menjalankan aplikasi robot sederhana. Beberapa konsep dasar yang dipelajari termasuk pembuatan *node*, komunikasi antar *node* menggunakan topik, serta penggunaan alat simulasi seperti Gazebo juga ikut dibahas.

Manfaat dari tutorial ini adalah pemahaman dasar tentang bagaimana mengembangkan aplikasi robotik menggunakan ROS, serta keterampilan praktis dalam menginstal dan menjalankan ROS pada sistem Linux. Tutorial ini merupakan langkah pertama yang baik bagi siapa saja yang tertarik untuk mendalami dunia robotika dan pengembangan perangkat lunak robot.