SCRIPT ALUR

1. Introduction to ROS

Alur Pengerjaan:

1. Pahami konsep dasar ROS dari video atau dokumentasi.
2. Identifikasi keunggulan ROS untuk pengembangan robotika.
3. Catat poin-poin utama seperti struktur file sistem, modularitas, dan komunitas ROS.

Perintah ROS:

1. Membuat Paket ROS
    - catkin_create_pkg package_name [dependency1] [dependency2]
    - Contoh: catkin_create_pkg mastering_ros_demo_pkg roscpp std_msgs actionlib actionlib_msgs
2. Membangun Workspace
    - catkin_make
    - Perintah ini digunakan untuk membangun paket-paket dalam workspace catkin.
3. Menjalankan ROS Master
    - roscore
    - Perintah ini memulai ROS master, yang diperlukan untuk menjalankan node ROS.
4. Mendaftar Topik Aktif
    - rostopic list
    - Perintah ini mencantumkan semua topik aktif dalam sistem ROS.
5. Menerbitkan ke Topik
    - rostopic pub /topic_name message_type args
    - Contoh: rostopic pub /numbers std_msgs/Int32 10
6. Menyubscribe ke Topik
    - rostopic echo /topic_name
    - Perintah ini menampilkan pesan yang diterbitkan ke topik yang ditentukan.
7. Memeriksa Pesan
    - rosmsg show message_type
    - Contoh: rosmsg show std_msgs/Int32
8. Bekerja dengan Layanan
    - Memanggil Layanan: rosservice call /service_name args
    - Mendaftar Layanan: rosservice list
    - Mendapatkan Info Layanan: rosservice info /service_name
9. Menggunakan Parameter Server
    - Mengatur Parameter: rosparam set parameter_name value
    - Mendapatkan Parameter: rosparam get parameter_name
    - Mendaftar Parameter: rosparam list

---

2. Getting Started with ROS

Alur Pengerjaan:

1. **Jalankan ROS Core:**

   ```
   roscore  # Menyalakan layanan inti ROS.
   ```

2. **Clone Repositori:**

   ```
   git clone https://github.com/PacktPublishing/Mastering-ROS-for-
   Robotics-Programming-Third-edition
   ```

3. **Masuk Direktori dan Build:**

   ```
   cd Chapter2  # Masuk ke folder.
   catkin_make  # Membangun file yang ada.
   ```

4. **Jalankan Publisher:**

   ```
   rosrun <nama_folder> <nama_file>  # Menjalankan file Publisher.
   ```

5. **Jalankan Subscriber:** Buka terminal lain:

   ```
   rosrun <nama_folder> <nama_file_subscriber>  # Menjalankan file
   Subscriber.
   ```

6. Verifikasi apakah Publisher dan Subscriber berjalan dengan baik.

---

3. 3D Modeling with ROS

Alur Pengerjaan:

1. **Build File:**

   ```
   cd Chapter3  # Masuk ke folder.
   catkin_make  # Membangun file seperti langkah sebelumnya.
   ```

2. **Jalankan File Launch untuk RViz:**

   ```
   roslaunch <nama_folder> <nama_file.launch>  # Meluncurkan file RViz.
   ```

3. **Buka Window RViz:**
   - Gunakan RViz untuk memodifikasi model robot.
   - Uji model dengan menjalankan file lainnya jika diperlukan.

---

4. Simulation Robot with ROS and Gazebo

Alur Pengerjaan:

1. **Masuk Direktori dan Build:**

   ```
   cd Chapter4  # Masuk ke folder.
   catkin_make  # Bangun file di direktori Chapter4.
   ```

2. **Jalankan File Gazebo:**

   ```
   roslaunch <nama_folder> <nama_file.launch>  # Meluncurkan simulasi
   Gazebo.
   ```

3. **Verifikasi Simulasi:**
   - Pastikan Gazebo terbuka.
   - Jalankan skenario simulasi sesuai kebutuhan.

---

5. Simulating Robots Using ROS, CoppeliaSim, and Webots

Alur Pengerjaan:

1. **Ekstrak File CoppeliaSim:**

   ```
   tar vxf CoppeliaSim_Edu_V4_8_0_rev0_Ubuntu20_04.tar
   ```

2. **Setel Variabel Lingkungan:**

   ```
   echo "export COPPELIASIM_ROOT=$HOME/Downloads/CoppeliaSim" >> ~/.bashrc

   source ~/.bashrc
   ```

3. **Install Dependensi:**

   ```
   sudo apt-get update
   sudo apt-get install liblua5.3-dev qt5-qmake qtbase5-dev
   ```

4. **Jalankan ROS Core:**

   ```
   roscore  # Jalankan ROS Core.
   ```

5. **Jalankan CoppeliaSim:**

   ```
   cd $COPPELIASIM_ROOT
   ```

```
./coppeliaSim.sh
```

6. **Verifikasi Integrasi ROS:**

```
rosnode list  # Pastikan sim_ros_interface terlihat.
```

7. **Buka File Simulasi:**
    - Pilih "Open Scene" pada menu dan jalankan file simulasi yang diinginkan.
