Rafli Limandijaya/1103210243

Hasil Analisis Simulasi MoveIt Installation dan Panda Robot Simulation

1. Installasi MoveIt

Proses instalasi MoveIt berjalan lancar setelah menambahkan repository dan mengikuti langkah-langkah instalasi untuk ROS 2 distro Humble. Setelah instalasi selesai, semua dependency berhasil diinstal dan MoveIt siap digunakan untuk robot Panda.

2. Konfigurasi Panda Robot di RViz

Pada tahap ini, robot Panda berhasil dikonfigurasi di RViz. RViz menyediakan visualisasi yang sangat membantu dalam memantau pergerakan robot secara real-time. Saya mengkonfigurasi model robot Panda dan mengintegrasikannya dengan MoveIt untuk mengontrol sendi robot.

3. MoveIt Setup Assistant

Setup Assistant digunakan untuk mengkonfigurasi robot untuk perencanaan kinematika dan dinamika. Melalui Setup Assistant, saya dapat menentukan grup kontrol robot dan mengatur lingkungan untuk perencanaan Gerakan.

4. Perencanaan Gerakan dan Eksekusi

Setelah instalasi dan konfigurasi selesai, saya dapat merencanakan gerakan untuk Panda Arm. Gerakan dilakukan dengan cara menentukan pose target dan menggunakan perencanaan jalur dari MoveIt untuk menghitung dan menjalankan pergerakan. Fungsi ini berhasil dijalankan di RViz dengan hasil visualisasi gerakan yang sesuai dengan perencanaan.

5. Simulasi Panda Arm

Simulasi robot Panda dilakukan dengan sukses melalui RViz, di mana robot berhasil bergerak sesuai target pose yang ditentukan.

6. Simulasi Pergerakan Looping dengan C++

Pada tahap ini, setelah berhasil menggerakkan robot Panda melalui RViz menggunakan API MoveIt!, saya melanjutkan dengan membuat kode C++ untuk melakukan simulasi pergerakan

robot dalam sebuah loop. Program ini bertujuan untuk menggerakkan robot antara dua titik pose target secara terus-menerus.

Implementasi Kode

1. Pendefinisian Grup Kontrol: Saya menggunakan MoveGroupInterface untuk mengontrol lengan robot (panda_arm). Ini adalah API MoveIt! yang memungkinkan kita merencanakan dan mengeksekusi gerakan untuk grup sendi tertentu.
2. Pose Target: Dua pose target didefinisikan menggunakan tipe data geometry_msgs::msg::Pose. Pose pertama dan kedua memiliki posisi X, Y, dan Z yang sedikit berbeda untuk mendemonstrasikan pergerakan robot ke arah yang berbeda.
3. Perencanaan dan Eksekusi Gerakan: Pergerakan dilakukan melalui fungsi perencanaan dari MoveIt! Jika perencanaan berhasil, pergerakan dieksekusi. Jika gagal, pesan error akan muncul di log.
4. Loop: Program ini dirancang untuk terus berulang selama node ROS 2 masih aktif. Setelah robot mencapai satu pose, ada jeda sebelum melanjutkan ke pose berikutnya.

Robot berhasil bergerak secara berulang-ulang antara dua pose target. Pada simulasi ini, robot mengulangi pergerakan tanpa hambatan, dan visualisasi di RViz menunjukkan jalur gerakan yang sesuai dengan perencanaan yang dibuat.

Simulasi C++ ini sangat efektif untuk menguji algoritma perencanaan jalur secara real-time. Dengan adanya loop, saya bisa mengevaluasi keandalan sistem perencanaan MoveIt! secara berkelanjutan. Hasilnya, Panda arm mampu kembali ke pose awal dan mengulangi pergerakan dengan akurasi yang tinggi.

6. Analisis Akhir

Secara keseluruhan, tutorial MoveIt yang mencakup instalasi hingga simulasi berjalan dengan baik. Meskipun terdapat beberapa kendala seperti kesalahan kinematika awal pada konfigurasi kinematics.yaml, masalah tersebut berhasil diatasi dengan memperbaiki plugin yang tepat. Dengan MoveIt, perencanaan dan eksekusi gerakan robot menjadi lebih mudah dan dapat digunakan sebagai basis untuk pengembangan lebih lanjut.