Andi Cleopatra Maryam Jamila

1103213071

Naskah Chapter 1 - Introduction to ROS.

- Robot Operating System (ROS) adalah kerangka kerja fleksibel yang menyediakan berbagai alat dan perpustakaan untuk menulis perangkat lunak robotik. Ini menawarkan beberapa fitur canggih untuk membantu pengembang dalam tugas-tugas seperti penyampaian pesan, komputasi terdistribusi, penggunaan kembali kode, dan implementasi algoritme canggih untuk aplikasi robotik. Proyek ROS dimulai pada tahun 2007 oleh Morgan Quigley dan pengembangannya dilanjutkan di Willow Garage, sebuah penelitian robotika Lab untuk mengembangkan perangkat keras dan perangkat lunak open source untuk robot. Tujuan ROS adalah untuk menetapkan cara standar untuk memprogram robot sambil menawarkan perangkat lunak siap pakai komponen yang dapat dengan mudah diintegrasikan dengan aplikasi robot khusus. Ada banyak alasan untuk memilih ROS sebagai framework pemrograman, dan beberapa di antaranya adalah sebagai berikut:
  - Kemampuan kelas atas: ROS hadir dengan fungsionalitas siap pakai. Misalnya Lokalisasi dan Pemetaan Simultan (SLAM) dan Monte Adaptif Paket Carlo Localization (AMCL) di ROS dapat digunakan untuk memiliki otonom navigasi di robot bergerak, sedangkan paket MoveIt dapat digunakan untuk gerakan Perencanaan untuk manipulator robot. Kemampuan ini dapat langsung digunakan di perangkat lunak robot tanpa kerumitan.
  - Dukungan untuk sensor dan aktuator kelas atas: ROS memungkinkan kami menggunakan perangkat yang berbeda driver dan paket antarmuka dari berbagai sensor dan aktuator dalam robotika. Demikian sensor kelas atas termasuk LIDAR 3D, pemindai laser, sensor kedalaman, aktuator, dan lebih.
  - Pengoperasian antar-platform: Middleware penglewatan pesan ROS memungkinkan komunikasi antara program yang berbeda. Di ROS, middleware ini dikenal sebagai simpul. Node ini dapat diprogram dalam bahasa apa pun yang memiliki klien ROS perpustakaan. Kita dapat menulis node high-haveance dalam C++ atau C dan node lain dalam python atau jawa.
  - Modularitas: Salah satu masalah yang dapat terjadi pada sebagian besar robot mandiri aplikasi adalah bahwa jika salah satu utas dari kode utama mogok, seluruh robot aplikasi bisa berhenti. Di ROS, situasinya berbeda menulis yang berbeda node untuk setiap proses, dan jika satu node mogok, sistem masih dapat berfungsi.
  - Penanganan sumber daya bersamaan: Menangani sumber daya perangkat keras melalui lebih dari dua proses selalu memusingkan. Bayangkan bahwa kita ingin memproses gambar dari kamera untuk deteksi wajah dan deteksi gerakan; kita dapat menulis kode sebagai entitas tunggal yang dapat melakukan keduanya, atau kita dapat

menulis potongan kode utas tunggal untuk concurrency. Jika kita ingin menambahkan lebih dari dua fitur ke utas, aplikasi perilaku akan menjadi kompleks dan sulit untuk di-debug. Namun di ROS, kita bisa mengakses perangkat yang menggunakan topik ROS dari driver ROS. Sejumlah node ROS dapat berlangganan pesan gambar dari driver kamera ROS, dan setiap node dapat memiliki fungsi yang berbeda. Ini dapat mengurangi kompleksitas dalam komputasi dan juga meningkatkan kemampuan debugging seluruh sistem.