URO TEST SOAL 1

1. Apa itu ROS (Robot Operating System), dan bagaimana peran utamanya dalam pengembangan robotik modern? (Jelaskan mengapa ROS penting untuk integrasi berbagai komponen robot seperti sensor, aktuator, dan kamera dalam satu sistem yang bekerja harmonis.)

   Jawaban:

   Robot Operating System merupakan sebuah kumpulan *software open source* yang menyediakan library dan alat untuk membuat aplikasi robot. ROS adalah sebuah platform untuk membuat *software*, yang dalam webnya disebut dengan *middleware* dan berperan sebagai *software developer kit*.

   ROS menjadi basis dalam sebagian besar riset di bidang *robotics*, dari *project* kecil sampai *project* kolaborasi *multi-institution* dan kompetisi skala besar. Selain itu, program ini juga digunakan dalam robot hasil produksi (komersil) yang digunakan di dunia saat ini.

   Komunitas ROS yang sudah berdiri lebih dari 10 tahun membuat sebuah ekosistem *software* untuk *robotics* dengan mewadahi jutaan developer dan user. ROS dikembangkan oleh dan untuk komunitas ini.

   ROS penting karena platform ini memungkinkan developer untuk mengintegrasikan komponen seperti sensor, akuator, kamera, dll. dalam sebuah *framework*. Hal ini memudahkan developer untuk menyatukan interaksi antar komponen, sehingga mengurangi waktu pengembangan. Misalnya, sensor mengambil data suhu, aktuator menggerakkan robot, dan kamera dengan model *computer vision* melakukan *object detection*. Ketika ketiga hal ini ingin diintegrasikan dalam sebuah program dan dijalankan secara *real-time*, diperlukan sebuah *framework* yang dapat melakukan koordinasi antarkomponen yang baik sehingga program berjalan dengan benar.

   Sumber: https://www.ros.org/

2. Apa perbedaan utama antara ROS dan ROS2, dan mengapa pengembang cenderung memilih ROS2 untuk proyek baru? (Jelaskan keunggulan ROS2 dibandingkan ROS dalam hal performa, keamanan, dan pemeliharaan jangka panjang.)

   Jawaban:

   Hal pertama yang perlu diketahui adalah ROS 1 versi paling baru hanya di support secara resmi hingga tahun 2025. Selanjutnya, tipe ini hanya akan ada support dari komunitas.

   Perbedaan signifikan dari ROS 1 dan ROS 2 terbagi menjadi tiga bagian, yaitu arsitektur, fitur, dan tooling/ekosistem.

   a. Arsitektur

      ROS 1 menggunakan *ROS Master-Slave Architecture* dan *XML-RPC middleware* yang sederhana tetapi kurang efisien untuk skala besar, sedangkan ROS 2 menggunakan *Data Distribution Service* yang mendukung efisiensi tinggi, *low-latency*, realibilitas, dan skalabilitas. ROS 1 memiliki dua *library* untuk C++ (roscpp) dan Python (rospy) dan keduanya memiliki fitur berbeda, sedangkan ROS 2 memiliki

library berbasis C (rcl) yang mendukung Python, C++, Java, dan C#. Kemudian, format data pada ROS 2 lebih fleksibel karena menggunakan format rosbag, yang menyebabkan pengelolaan data lebih efisien.

b. Fitur
   ROS 2 memungkinkan node berjalan paralel, memanfaatkan processor multi-core dengan lebih efisien. Selain itu, ROS 2 memungkinkan kita mengatur konfigurasi aliran data, contohnya realiabilitas pesan, deadline, dan prioritas, yang membuat pesan penting tiba tepat waktu. ROS 2 juga mendukung pemrosesan secara real-time dengan *middleware* DDS, khususnya ketika *low-latency* dibutuhkan.
c. Tooling/Ekosistem
   ROS 2 menggunakan Ament untuk menggantikan Catkin sebagai *build system*. Selain itu, ROS 2 menyediakan workspace kedua, sehingga kita bisa melakukan eksperimen tanpa mengubah hal pada workspace utama. Namun, ROS 2 tidak kompatibel dengan ROS 1, yang menyebabkan packages pada ROS 1 tidak bisa digunakan pada ROS 2 secara langsung. Terakhir, ROS 1 hanya kompatibel dengan Ubuntu, tetapi ROS 2 dapat dijalankan di MacOS, Windows, Ubuntu, dan *operating system* lainnya.

Sumber: https://www.model-prime.com/blog/ros-1-vs-ros-2-what-are-the-biggest-differences

3. Mengapa simulasi robotik penting dalam pengembangan robot, dan apa keuntungan menggunakan simulasi sebelum membangun robot fisik? (Berikan contoh kasus di mana simulasi dapat menghemat waktu dan biaya pengembangan robot.)

Jawaban:

Simulasi robotik penting dalam pengembangan robot karena tahap ini dapat mengetahui kelebihan dan kekurangan dari robot tersebut sebelum tahap produksi. Tahap pengujian awal ini menguji desain, algoritma, atau aspek lainnya dari robot tersebut. Hal ini dapat mengurangi waktu dan biaya yang terbuang jika dibandingan dengan pengujian langsung pada komponen fisik. Simulasi lebih mudah dan cepat dilakukan, sehingga pengembangan lebih cepat dilakukan.

Contoh kasusnya adalah robot *surgery*, simulasi dilakukan agar dapat melihat apakah algoritma atau program yang digunakan berjalan dengan baik dan melakukan operasi dengan baik. Dengan tidak melakukan simulasi langsung dengan robotnya, kita dapat melakukan simulasi secara cepat dan tidak membakar uang dengan merakit robot dengan berbagai desain untuk menemukan yang tepat.

4. Apa itu Gazebo, dan bagaimana Gazebo digunakan untuk mensimulasikan lingkungan fisik bagi robot? (Jelaskan langkah-langkah dasar mengintegrasikan ROS dengan Gazebo untuk mengontrol robot dalam simulasi.)

Jawaban:

Gazebo adalah environment open-source untuk simulasi robot. Project ini dijalankan oleh Open Robotics, yaitu grup yang juga menjalankan ROS. Namun, project ini dijalankan secara terpisah dan bukan bagian dari ROS.

Dalam Gazebo, user dapat membuat virtual world, and melakukan load robot versi simulasi ke dalamnya. Sensor dalam simulasi ini juga dapat menangkap data, misalnya suhu, dan melakukan publish data seperti halnya sensor asli. Selain itu, gaya seperti gaya gesek juga bisa disimulasikan ke dalam Gazebo, sehingga membuat simulasi semakin mirip dengan dunia asli.
Langkah mengintegrasikan ROS 1 dengan Gazebo:

- Install ROS dan Gazebo.
- Cari/buat model robot yang ingin di “spawn”.
- Gunakan plugin Gazebo.
- Buat file launch ROS untuk melakukan simulasi di Gazebo.
- Kalau robot sudah muncul di dalam Gazebo, robotnya bisa di kontrol.
- Jalankan file launch untuk memulai simulasi.

Sumber: https://articulatedrobotics.xyz/tutorials/ready-for-ros/gazebo, https://gazebosim.org/

5. Bagaimana cara kerja navigasi robot di dunia simulasi? (Apa saja konsep dasar seperti mapping dan lokalisasi yang perlu dipahami, dan bagaimana fitur ini dapat diimplementasikan pada robot Anda?)

Jawaban:

a. Mapping / Pemetaan
   Robot membuat representasi lingkungan (bisa berupa 2D atau 3D), yang membantu robot menentukan jalur yang akan dilewatinya kemudian. Misalnya: gmapping (Grid-based FastSLAM), yaitu package yang menghasilkan grid 2D untuk merepresentasikan lingkungan yang menunjukkan kosong atau tidaknya setiap sel dari grid tersebut.
b. Lokalisasi
   Robot menentukan posisinya dalam mapping yang telah dibuat sebelumnya, sehingga robot selalu tahu dimana dirinya berada secara real-time. Hal ini berguna agar robot dapat membuat keputusan berdasarkan lokasi akurat. Misalnya: odometry-based localization, yaitu menggunakan data dari roda; sensor-based localization, yaitu melakukan integrasi data dari berbagai sensor; dll.

Berdasarkan mapping dan lokalisasi yang dilakukan, robot akan melakukan perencanaan jalur atau rute dari posisi robot saat ini ke lokasi tujuan. Kemudian, untuk mengimplementasi hal-hal diatas, kita dapat mengintegrasikan sensor seperti LIDAR untuk memberikan data input yang digunakan dalam melakukan mapping dan lokalisasi. Selanjutnya, perencanaan rute akan menggunakan algoritma tertentu.

Sumber: https://medium.com/@mansooralam129047/localization-in-robotics-for-mobile-robots-ec3ad31f99d4, https://medium.com/@mansooralam129047/what-is-mapping-in-robotics-how-to-create-map-in-ros-8c002d409c07

6. Apa itu TF (Transform) dalam konteks ROS, dan bagaimana TF membantu robot memahami posisi dan orientasinya dalam ruang tiga dimensi? (Berikan contoh

bagaimana TF digunakan untuk memastikan robot bergerak dengan benar dalam simulasi.)

Jawaban:

TF adalah sebuah package yang membantu user untuk melacak frame koordinat 3D secara real-time. Ini digunakan dalam ROS untuk melacak posisi dan orientasi bagian robot, sehingga robot mengetahui bagaimana beberapa komponen saling berkaitan.

Beberapa pertanyaan yang bisa dijawab menggunakan ini, yaitu:

- Where was the head frame relative to the world frame, 5 seconds ago?
- What is the pose of the object in my gripper relative to my base?
- What is the current pose of the base frame in the map frame?

Contoh penggunaan TF: Misalnya, robot dengan lengan yang dilengkapi kamera dapat menggunakan TF untuk melacak posisi kamera relatif dengan basis robot. Saat robot bergerak, TF akan terus memperbarui posisi relatif kamera sehingga robot dapat mengarahkan kamera secara akurat ke objek.

Sumber: http://wiki.ros.org/tf, https://foxglove.dev/blog/understanding-ros-transforms