1. a. ROS (Robot Operating System) adalah sebuah kerangka kerja perangkat lunak sumber terbuka yang dapat dipakai untuk mempermudah mengembangkan aplikasi robotika. ROS sangat membantu dan penting untuk integrasi komponen-komponen elektronika dikarenakan ROS dapat membuat komponen elektronika dikerjakan satu persatu dan dapat langsung dihubungkan dengan bantuan sistem ROS. ROS juga dapat menggabungkan komponen elektronika dengan bahasa pemrograman yang berbeda, sehingga kerjasama antar tim robotika dapat lebih efisien sesuai keahlian bahasa pemrograman di bidangnya masing-masing.
   https://en.wikipedia.org/wiki/Robot_Operating_System
   b. ROS 1 dirancang untuk pembuatan aplikasi akademis dan tidak dapat digunakan untuk proyek-proyek besar, sedangkan ROS 2 memang dirancang untuk proyek komersial. Selain itu, ROS 2 juga menghadirkan fitur-fitur baru yang lebih support untuk sistem-sistem baru dimana ROS 1 belum terlalu mensupport untuk sistem-sistem sekarang ini, akan tetapi seluruh fitur yang ada di ROS 1 belum seluruhnya dihadirkan di ROS 2. ROS 2 lebih unggul dalam beberapa fitur, misalnya penggunaan DDS (Data Distribution Service) sebagai middleware komunikasi sehingga lebih dekat dengan keamanan yang dapat disertifikasi dan membuat komunikasi antar node lebih efisien. Selain itu ROS 2 mendukung adanya real time support yang membuat manajemen memori lebih baik. Meskipun begitu, masih banyak pengguna yang lebih memilih memakai ROS karena pertimbangan lainnya.
   https://www.swri.org/industry/industrial-robotics-automation/blog/the-ros-1-vs-ros-2-transition
   c. Simulasi robot penting untuk melihat pengaplikasian dari teori yang telah dihitung sedemikian rupa, dan agar dapat mengetahui hal apa saja yang dapat diperbaiki lebih lagi dari robot secara lebih akurat (perhitungan matematis tidak dapat menentukan semua ini karena adanya faktor kesalahan dan kalkulus tidak dapat menghitung seakurat itu), selain itu fungsionalitas robot dan waktu kerja dapat dilihat langsung. Contoh kasusnya adalah pembuatan robot ekspedisi NASA. Robot seperti ini pastinya sangat mahal, dan apabila ternyata saat sudah dikirim ke antariksa gagal, pasti akan sangat merugikan secara biaya dan waktu, sehingga NASA pastinya melakukan berbagai simulasi terlebih dahulu dengan memperhitungkan lingkungan yang ada disana serta gangguan eksternal lainnya.
   https://facteon.global/news/key-benefits-of-robot-simulation/
   d. Gazebo adalah suatu program simulasi robotika 2D/3D sumber terbuka. Program simulasi 3D ini akan sangat membantu simulasi robot dalam lingkungan fisik asli dengan perhitungan yang cukup akurat nantinya, sehingga dapat hemat biaya dan waktu tentunya. Untuk langkah dasar pengintegrasian Gazebo dengan ROS:
   -Pastikan sudah mendownload Gazebo dan ROS
   -Persiapkan model robot yang ingin dibuat
   -Load model ke Gazebo dan plugin Gazebo-ROS
   -Robot sudah dapat dikontrol
   -Kasih fitur lainnya seperti sensor untuk fungsionalitas lebih tinggi
   -Gunakan Rviz untuk mempermudah gerak robot dan pembacaan data sensor
   https://classic.gazebosim.org/tutorials?tut=ros_overview
   e. Simulasi Navigasi pada Gazebo dapat dilakukan untuk mengecek bagaimana kerja robot secara mandiri tanpa kontrol manusia. Hal ini dapat dilakukan dengan beberapa langkah berikut:
   1. Mapping (Membuat peta lingkungan dimana robot akan disimulasikan)

2. Localization (Setelah peta selesai, kita dapat melakukan lokalisasi untuk memastikan robot hanya bertempat sesuai map)
3. Path Planning (Pembuatan algoritma untuk mengatur tujuan robot yang paling optimal dan efisien)
4. Pantau robot di Gazebo dengan bantuan Rviz agar semakin mudah dilihat pergerakan dan data yang dihasilkan

Contoh pengimplementasian pada robot adalah:
Pembuatan robot autocleaner untuk membersihkan lantai dari debu.

1. Mapping area lantai
2. Lokalisasi daerah jangkauan robot yaitu lantai-lantai dan hanya dapat bergerak 360 derajat sejajar dengan bidang datar
3. Membuat algoritma paling efisien untuk mengecek semua debu sekali saja tanpa mengulangi jalan yang sama
4. Pantau robot di Gazebo dengan bantuan Rviz

https://wiki.ros.org/3d_navigation

f. TF memiliki peran penting dalam beberapa aspek robotika, terutama mapping, lokalisasi, navigasi, dan manipulasi. TF dapat mengelola hubungan antar frame secara 3D dengan mengelola setiap perubahan posisi dan orientasi antar frame secara dinamis dan otomatis. Selain itu waktu yang digunakan di ROS akan disinkronisasi dengan waktu asli, sehingga dapat melacak posisi frame pada momen waktu tertentu, sehingga dapat mengetahui posisi 3D secara akurat. Hal ini juga penting agar dapat mengetahui gerakan robot dalam setiap eksekusinya, terutama ketika ada penundaan atau percepatan perpindahan posisi seiring waktu.
https://foxglove.dev/blog/understanding-ros-transforms