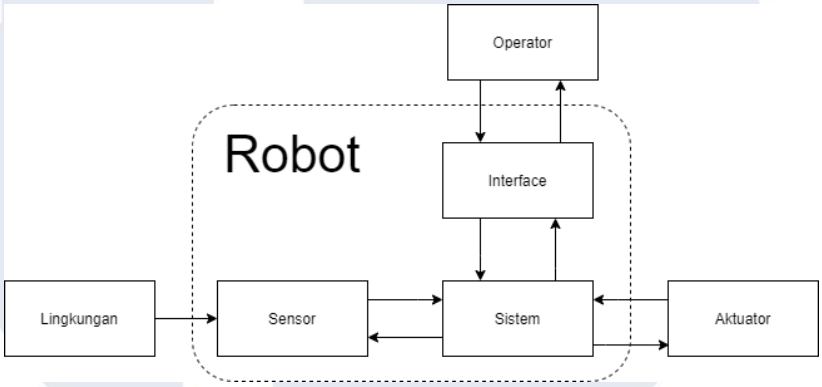
BAB III

PERANCANGAN DAN IMPLEMENTASI SISTEM

3.1 Tinjauan Desain Sistem

3.1.1 Desain Sistem Keseluruhan

Pada simulasi mobile robot desain sistem secara keseluruhan ini akan menjelaskan hubungan dari sistem robot dengan dunia yang terdapat di gazebo. Kemudian konfigurasi sistem dan pembacaan data sensor dapat dilihat pengguna sebagai *interface* menggunakan *software* RViz 2 karena aplikasi ini bersifat interaktif. Gambar 3.1 menunjukkan *Data Flow Diagram* (DFD) *Level* 0 dari sistem robot dengan dunia gazebo.



Gambar 3.1 - *Data Flow Diagram Level* 0 Simulasi Robot *Rover*

Tabel 3.1 - Penjelasan DFD *Level* 0 Simulasi Robot *Rover*

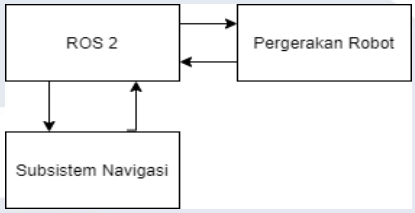
Parameter	Keterangan
Input	● Konfigurasi sistem dari pengguna. ● Pembacaan sensor terhadap lingkungan di sekitar robot.
Output	● Hasil pemetaan menggunakan SLAM *toolbox*. ● Data jarak dan sudut objek di sekitar robot. ● Pergerakan robot ke titik tujuan. ● Tampilan perintah yang dapat diberikan ke robot. ● Tampilan perintah yang sedang dijalankan. ● Data dari sistem /*cmd_vel*.
Fungsi	● Melakukan *localization* dan *mapping*. ● Mengatur pergerakan robot. ● Menampilkan perintah yang sedang dijalankan. ● Menampilkan data yang sedang berjalan di /*cmd_vel*.

Berdasarkan Tabel 3.1, simulasi robot akan bergerak dengan cara robot menerima titik tujuan dari pengguna. Kemudian robot bergerak ke titik tujuan dan menghindari halangan yang berada di sekitar robot. Robot dapat

mendeteksi keberadaan objek di sekitar menggunakan data pembacaan jarak dan sudut yang kemudian diproses menjadi data *localization* dan *mapping*.

3.1.2 Desain Subsistem Navigasi

Gambar 3.2 menunjukkan *Data Flow Diagram* (DFD) *Level* 1 yang menjelaskan hubungan sistem ROS 2 dengan subsistem navigasi dan program pergerakan robot.



Gambar 3.2 - *Data Flow Diagram Level* 1 Simulasi Robot *Rover*

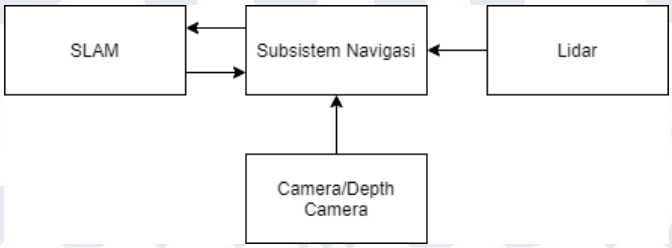
Tabel 3.2 - Penjelasan DFD Level 1 Simulasi Robot *Rover*

Parameter	Keterangan
Input	● *Point Clouds* dari *laserscan*. ● Data warna dan kedalaman dari *depth camera*. ● NAV2.
Output	● *Topic /global/map*. ● *Topic /depth/camera.* ● *Obstacle Avoidance*.
Fungsi	● Menampilkan hasil pemetaan di dalam RViz 2. ● Menampilkan gambar pembacaan kamera di dalam RViz 2. ● Robot bergerak menghindari halangan.

Berdasarkan Tabel 3.2, Robot akan menggunakan titik-titik *point clouds* dari hasil pendeteksian sensor lidar untuk dijadikan *topic* yang ditampilkan di dalam RViz 2. *Topic* tersebut dijadikan *mapping* dan *localization*, serta menunjukkan keberadaan objek di sekitar robot. Dengan menggunakan program NAV2, robot dapat menghindari halangan yang terdeteksi *laserscan* dan robot dapat bergerak ke titik tujuan menggunakan *path planning* A*.

3.1.3 Diagram Sistem Navigasi Robot

Gambar 3.3 menunjukkan *Data Flow Diagram* (DFD) *Level* 2 yang menjelaskan hubungan subsistem navigasi.



Gambar 3.3 - *Data Flow Diagram Level* 2 Subsistem Navigasi Simulasi Robot *Rover*

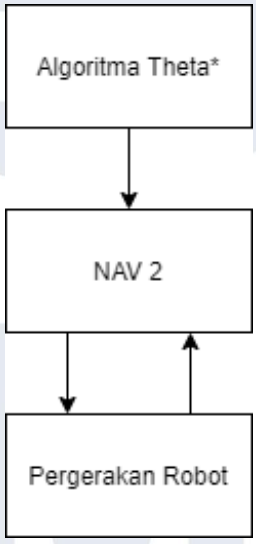
Tabel 3.3 - Penjelasan DFD Level 2 Subsistem Navigasi Simulasi Robot *Rover*

Parameter	Keterangan
Input	● *Topic* /*laserscan* dari sensor lidar. ● *Topic* /*depth/image* dari kamera. ● Data gambar dan kedalaman menggunakan *depth camera*.
Output	● *Topic* /*map* dari hasil *mapping*. ● Gambar lingkungan di depan robot. ● Tanda keberadaan objek di sekitar robot.
Fungsi	● Robot dapat membuat peta menggunakan SLAM *toolbox*. ● Robot menampilkan gambar lingkungan yang ada di depannya.

Berdasarkan Tabel 3.3, sensor lidar mengirim data jarak dan sudut menjadi *topic laserscan* yang digunakan untuk robot membuat SLAM *toolbox* di dalam aplikasi RViz 2. Selain pembacaan sensor lidar, kamera juga dapat mengirim *topic image* yang dapat ditampilkan di dalam aplikasi RViz 2.

3.1.4 Diagram Sistem Pergerakan Robot

Gambar 3.4 menunjukkan *Data Flow Diagram* (DFD) *Level* 2 yang menjelaskan hubungan algoritma A* di dalam NAV untuk pergerakan robot.



Gambar 3.4 - Sistem Pergerakan Robot Berdasarkan Algoritma A*

Tabel 3.4 - Penjelasan Sistem Pergerakan Robot Berdasarkan Algoritma A*

Parameter	Keterangan
Input	● Algoritma A* di dalam sistem robot. ● */cmd_vel*.
Output	● *Obstacle avoidance*.
Fungsi	● Robot bergerak dari titik *start* ke titik tujuan.

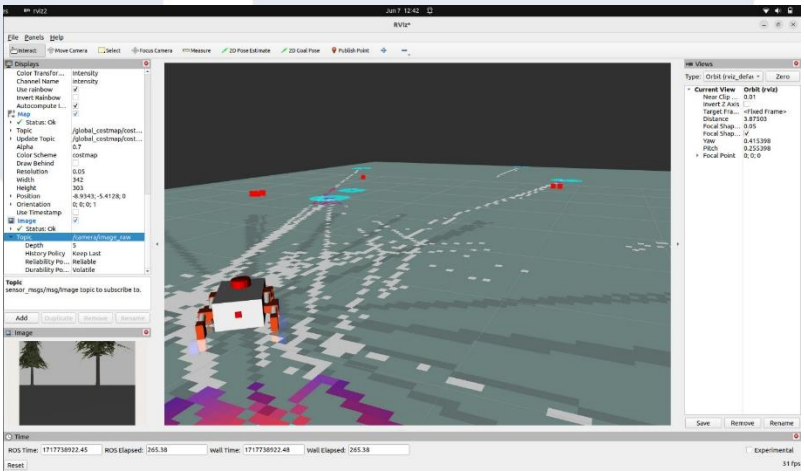
	• Robot bergerak dengan menghindari halangan yang ada di dalam sekitarnya.

Berdasarkan Tabel 3.4, algoritma A* diprogram di dalam sistem NAV2 untuk robot bergerak dari titik *start* ke titik tujuan dengan mengambil jalur tercepat dan paling optimal. Pada sistem ini robot melakukan perencanaan pergerakan serta bergerak menghindari halangan.

3.2 Implementasi Sistem

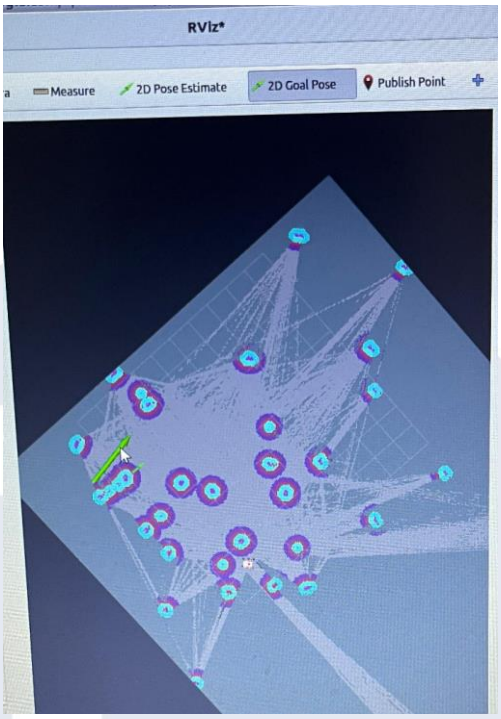
3.2.1 Hasil Implementasi

Percobaan pertama simulasi robot *rover outdoor* menggunakan pohon sebagai halangan dan permukaan yang datar. Pada percobaan ini robot menampilkan gambar yang berisi keadaan di depan robot beserta pembacaan *laserscan*. Berdasarkan Gambar 3.5, hasil *topic camera* dan *laserscan* ditampilkan di dalam aplikasi RViz 2. Kemudian dengan menggunakan SLAM *toolbox*, RViz 2 dapat menampilkan peta yang berisi informasi keberadaan objek di sekitar robot.



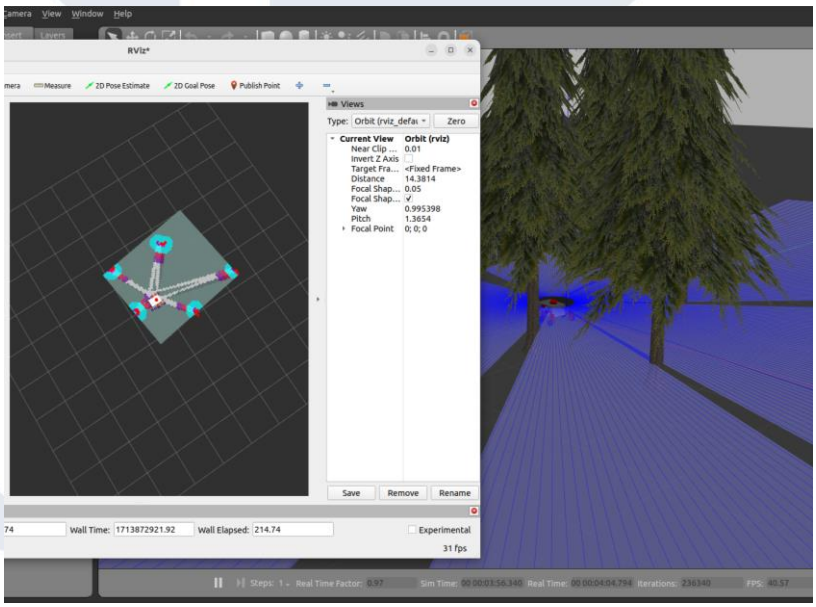
Gambar 3.5 - Tampilan Seluruh *Topic* Simulasi Robot *Rover* di Rviz 2

Selanjutnya setelah berhasil menampilkan *topic* dari robot, pengambilan data dilakukan dengan cara memberikan perintah titik tujuan kepada robot di dalam RViz 2. Ketika titik tujuan diberikan, robot akan bergerak berdasarkan algoritma A* yang telah diprogram di dalam NAV2 seperti yang ditunjukkan pada Gambar 3.6. Percobaan pertama menggunakan *file* rata.*world* untuk mengambil data waktu dari titik *start* ke titik tujuan. Gambar 3.6 menunjukkan implementasi awal robot dari titik *start* menuju titik tujuan. Pada percobaan ini, menggunakan pohon sebagai halangan dan permukaan datar sebagai jalur robot.



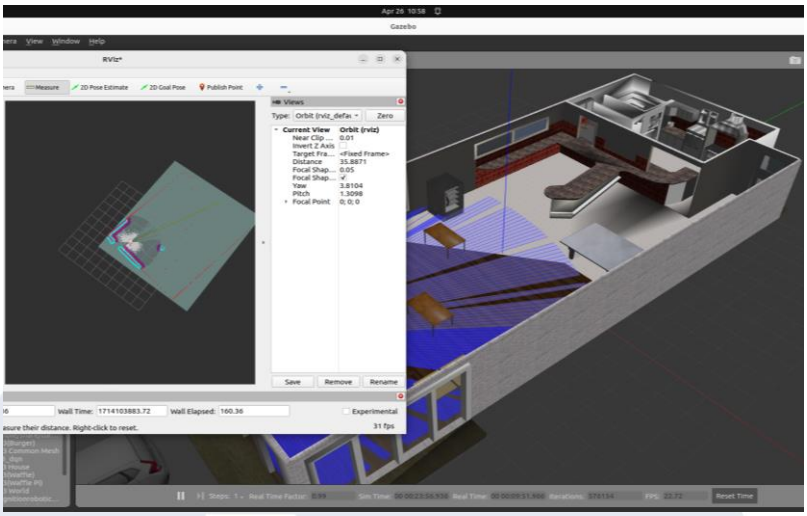
Gambar 3.6 - Perintah Titik Tujuan Simulasi Robot di RViz 2 (rata.*world*)

Percobaan berikutnya merupakan simulasi robot pada rata2.*world*. Percobaan ini dilakukan sebanyak 12 kali untuk mengambil data *linear*, *angular*, jarak, waktu dan kecepatan. Gambar 3.7 menunjukkan hasil pembacaan *laserscan* dan *mapping* SLAM *toolbox* di dalam rata2.*world*.



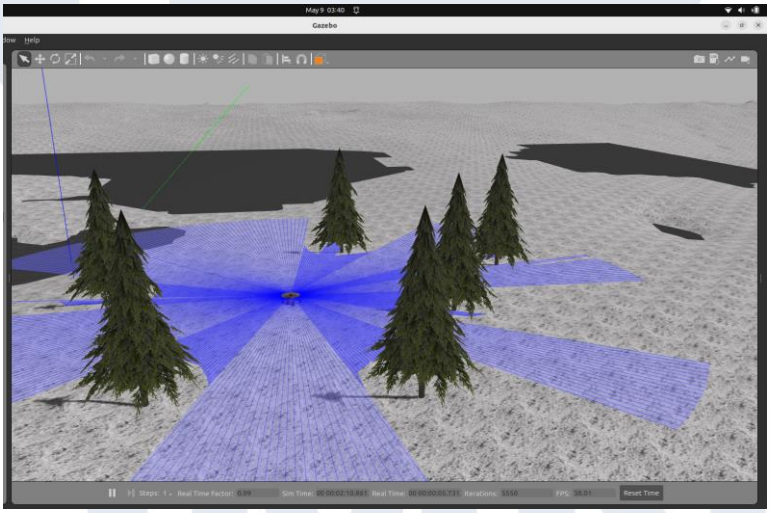
Gambar 3.7 - Simulasi Robot di rata2.*world*

Percobaan selanjutnya simulasi robot *indoor*. Percobaan dilakukan dari titik *start* ke titik tujuan sebanyak 20 kali untuk mendapatkan data jarak, waktu, kecepatan, *linear* dan *angular*. Gambar 3.8 merupakan hasil simulasi robot *indoor*.



Gambar 3.8 – Simulasi Robot *Indoor*

Percobaan selanjutnya adalah simulasi robot *outdoor* dengan permukaan tidak rata. Pada percobaan simulasi ini robot akan bergerak dari titik *start* ke titik tujuan sebanyak 20 kali. Simulasi ini dilakukan untuk mengetahui apakah robot dapat bergerak pada permukaan yang tidak rata dan mengambil data perubahan nilai jarak, waktu dan kecepatan. Gambar 3.9 merupakan percobaan *outdoor* permukaan tidak rata.



Gambar 3.9 – Simulasi Robot *Outdoor* Permukaan Tidak Rata

3.2.2 Hambatan Implementasi

1. *Mapping* yang ditampilkan SLAM *toolbox* menggunakan *topic /costmap* terkadang mengalami gangguan yaitu hasilnya tidak sesuai dengan *world* gazebo.
2. Objek hasil pembacaan *laserscan* terlihat bergerak dan bergeser ketika robot berputar. Hal ini dapat dilihat di dalam hasil *mapping* pada aplikasi RViz 2.
3. Penambahan objek membuat gazebo menjadi *lag* dan *force close*.

3.2.3 Solusi yang Diterapkan

1. Solusinya mengulangi ros2 *run* SLAM *toolbox* pada program robot, jika masih terdapat *error* maka ROS 2 mengalami *bug* yang dapat dilihat di dalam terminal. Untuk mengatasi permasalahan ini adalah dengan *close* terminal kemudian buka terminal baru dan jalankan proses dari awal lagi.
2. Solusi untuk permasalahan *laserscan* ini adalah dengan menjalankan kembali robot dari titik *start* ke titik tujuan.
3. Ketika gazebo mengirim perintah tutup aplikasi atau menunggu. Pilih bagian menunggu sampai objek berhasil terinstal. Notifikasi ini akan muncul berkali-kali, namun dengan menunggu sampai instalasi selesai maka sistem gazebo akan kembali normal.