Analisis Tugas Robotika Week 5

Nama: Muhammad Nugraha Sadewa
Nim : 1103210061

Pendahuluan

Pada tugas ini, kami mengimplementasikan tiga algoritma perencanaan jalur, yaitu Dijkstra, A*, dan Cell Decomposition. Selanjutnya, dilakukan simulasi perencanaan jalur pada ROS yang meliputi Path Searching dan Trajectory Optimization. Untuk memperkaya analisis, kami membandingkan hasil dari setiap planner yang ada, termasuk GBFS, Dijkstra, dan A*, serta mengevaluasi hasil animasi dari setiap algoritma tersebut.

1. Implementasi Algoritma Perencanaan Jalur

a. Dijkstra Algorithm

Dijkstra adalah algoritma yang menjamin pencarian jalur terpendek antara titik awal dan tujuan dalam graf dengan biaya non-negatif. Dari hasil implementasi, algoritma ini menunjukkan performa yang baik dalam menemukan jalur optimal tanpa bantuan heuristic. Namun, Dijkstra cenderung lambat di graf besar, terutama dalam lingkungan yang kompleks, karena memeriksa semua kemungkinan node. Penggunaan Dijkstra ideal untuk situasi yang mengutamakan jalur terpendek dengan akurasi tinggi, meskipun mengorbankan waktu komputasi.

***b.* A* Algorithm**

A* meningkatkan kecepatan pencarian jalur dengan menggunakan heuristic yang memperkirakan jarak ke tujuan. Pada implementasi ini, heuristic berupa jarak Manhattan digunakan untuk mempercepat pencarian. Hasil implementasi menunjukkan A* sebagai algoritma yang lebih efisien dibandingkan Dijkstra, terutama pada graf atau grid yang lebih besar. Namun, A* masih dapat menghasilkan jalur yang tidak optimal jika heuristic kurang akurat, sehingga penting untuk memilih heuristic yang sesuai dengan struktur ruang yang diberikan.

c. Cell Decomposition

Algoritma Cell Decomposition membagi ruang navigasi menjadi cell-cell kecil, yang berguna untuk perencanaan jalur dalam ruang besar atau tidak beraturan. Dari hasil simulasi, Cell Decomposition bekerja optimal di lingkungan terbuka dengan rintangan minimal. Namun, pada lingkungan dengan banyak rintangan kecil, pendekatan ini menjadi kurang efisien karena membutuhkan lebih banyak cell dan evaluasi. Dalam hal kompleksitas ruang dan rintangan, Cell Decomposition kurang efisien dibandingkan algoritma yang mengandalkan graf atau grid.

2. Simulasi Motion Planning pada ROS

a. Path Searching

Path Searching dengan ROS memungkinkan pencarian jalur optimal di lingkungan yang lebih kompleks. Kami menggunakan beberapa algoritma pencarian jalur seperti GBFS, Dijkstra, dan A*, masing-masing menunjukkan kelebihan dan kekurangan dalam simulasi ROS:

- **Dijkstra** menemukan jalur optimal dengan metode evaluasi biaya, namun lambat dalam eksekusi pada graf besar.
- **A*** memberikan hasil yang hampir sama optimalnya dengan Dijkstra, tetapi lebih cepat karena menggunakan heuristic.
- **GBFS (Greedy Best-First Search)** bekerja cepat tetapi cenderung menghasilkan jalur yang kurang optimal karena berfokus hanya pada tujuan tanpa mempertimbangkan biaya jalur yang sudah dilalui.

Dari hasil simulasi, A* memberikan keseimbangan terbaik antara kecepatan dan akurasi jalur, sedangkan Dijkstra tetap lebih handal untuk jalur terpendek meskipun lebih lambat.

b. Trajectory Optimization

Optimasi lintasan memberikan aspek kelancaran dan kecepatan pada jalur yang dihasilkan. Pada simulasi, teknik optimasi seperti gradient descent atau sampling-based optimization diterapkan untuk memperhalus jalur agar lebih realistis dan aman untuk robot. Dari simulasi ROS, terlihat bahwa optimasi lintasan membantu mengurangi perubahan arah dan kecepatan yang mendadak, terutama untuk robot bergerak dalam ruang nyata. Tantangan utama optimasi lintasan adalah menjaga keseimbangan antara kelancaran dan waktu tempuh, di mana jalur yang terlalu halus cenderung memperpanjang waktu tempuh.

3. Analisis Hasil Animasi Setiap Planner

Animasi pada setiap algoritma memberikan visualisasi yang membantu memahami perbedaan antar planner:

- **GBFS (Greedy Best-First Search)** Pada animasi, GBFS menunjukkan pencarian jalur yang cepat tetapi cenderung tidak optimal karena hanya berfokus pada tujuan akhir tanpa mempertimbangkan biaya total jalur. Jalur yang dihasilkan sering kali berkelok-kelok, kurang efisien, dan bisa menimbulkan masalah pada robot jika perubahan arah terlalu mendadak.
- **Dijkstra** Dijkstra memberikan animasi yang lebih sistematis dengan pencarian menyeluruh pada semua node hingga mencapai tujuan. Jalur yang dihasilkan akurat dan optimal, tetapi proses pencarian terlihat lambat pada graf yang besar. Kelebihan Dijkstra terlihat dalam kepastian menemukan jalur terpendek tanpa tergantung pada posisi relatif node dengan tujuan.
- **A*** Pada animasi, A* memperlihatkan pencarian yang lebih cepat dari Dijkstra karena mengandalkan heuristic untuk memfokuskan pencarian pada jalur yang kemungkinan

besar mengarah ke tujuan. A* memberikan jalur yang hampir selalu optimal dan menunjukkan pemilihan node yang lebih relevan dengan arah tujuan.

Ringkasan Perbandingan dan Observasi

Dari hasil animasi, Dijkstra menjamin jalur terpendek tetapi lambat, A* menghemat waktu pencarian dengan kualitas jalur yang hampir selalu optimal, dan GBFS unggul dalam kecepatan tetapi sering kali memberikan jalur yang kurang efisien. Dalam konteks optimasi lintasan, Dijkstra dan A* menunjukkan hasil terbaik ketika jalur dioptimalkan untuk kehalusan dan keamanan, sedangkan GBFS sulit dioptimalkan karena perubahan arah yang tidak terstruktur.

Kesimpulan

Setiap algoritma memiliki keunggulan dan kelemahan unik sesuai aplikasinya:

- **Dijkstra** cocok untuk aplikasi yang membutuhkan jalur terpendek dengan jaminan akurasi.
- **A*** unggul dalam pencarian cepat dengan hasil optimal dalam banyak kasus berkat penggunaan heuristic.
- **GBFS** cocok untuk lingkungan sederhana atau terbuka tetapi kurang optimal untuk pencarian jalur yang benar-benar efisien.
- **Cell Decomposition** bermanfaat untuk navigasi di ruang terbuka, tetapi kurang efisien di area dengan rintangan kompleks.

Trajectory Optimization menambahkan aspek kelancaran dan keamanan pada jalur, yang penting untuk aplikasi di dunia nyata. A* dan Dijkstra mendapatkan manfaat paling besar dari optimasi ini, menghasilkan jalur yang lebih halus dan aman.