Nama : Hamdan Syaifuddin Zuhri
NIM : 1103220220
Kelas : TK-45-G09

#Tugas Week 5 Robotika

Analisis Simulasi

A. Algoritma Djikstra

Algoritma Dijkstra untuk menemukan jalur terpendek dalam graf yang dibuat menggunakan pustaka NetworkX. Fungsi utama dijkstra menerima graf, sumber, dan target sebagai input. Dalam fungsi ini, algoritma memelihara struktur data untuk mencatat jarak terpendek dari sumber ke setiap node, serta node mana yang menjadi orang tua dari setiap node yang dikunjungi. Setelah menyelesaikan pencarian, fungsi `backtrace` digunakan untuk menentukan jalur terpendek dengan melacak kembali dari node tujuan ke sumber berdasarkan hubungan orang tua yang dicatat sebelumnya. Jika jalur ke node tujuan ditemukan, jalur tersebut dikembalikan dan dicetak, jika tidak, fungsi mengembalikan list kosong.

Setelah menjalankan fungsi `dijkstra`, program menggunakan Matplotlib untuk menggambarkan graf, meyorot jalur terpendek dengan warna merah dan garis yang lebih tebal. Dalam hasil visualisasi, semua node digambarkan dengan warna biru muda, dan label bobot pada setiap sisi ditampilkan.

B. Algoritma A*

Program visualizer algoritma A* ini memanfaatkan Pygame untuk menggambarkan bagaimana algoritma perencanaan jalur bekerja dalam konteks grid. Kode ini terdiri dari kelas `Node` yang merepresentasikan elemen dalam grid, dengan kemampuan untuk menggambar dan memperbarui tetangga berdasarkan penghalang yang ditetapkan oleh pengguna. Interaksi pengguna didukung dengan penggunaan mouse untuk menetapkan titik awal dan tujuan serta menambahkan penghalang, sementara algoritma A* dijalankan ketika pengguna menekan spasi. Jalur tercepat yang ditemukan ditampilkan dalam warna ungu, sedangkan jalur yang dieksplorasi berwarna orange.

Hasil yang diperoleh dari menjalankan program ini menunjukkan efisiensi algoritma A* dalam menemukan jalur dari titik awal ke titik tujuan meskipun ada penghalang yang diatur secara manual.

C. Cell Decomposition

Kode pada cell_decomposition.py mengimplementasikan algoritma Dijkstra untuk menemukan jalur terpendek dalam graf yang dibuat menggunakan pustaka NetworkX. Fungsi utama `dijkstra` menerima graf, sumber, dan target sebagai input. Dalam fungsi ini, algoritma memelihara struktur data untuk mencatat jarak terpendek dari sumber ke setiap node, serta node mana yang menjadi orang tua dari setiap node yang dikunjungi. Setelah menyelesaikan pencarian, fungsi `backtrace` digunakan untuk membangun jalur terpendek dengan melacak kembali dari node tujuan ke sumber berdasarkan hubungan orang tua yang dicatat sebelumnya. Jika jalur ke node tujuan ditemukan, jalur tersebut dikembalikan dan dicetak, jika tidak, fungsi mengembalikan list kosong.

Setelah menjalankan fungsi `dijkstra`, program menggunakan Matplotlib untuk menggambarkan graf, menyoroti jalur terpendek dengan warna merah dan garis yang lebih tebal. Dalam hasil visualisasi, semua node digambarkan dengan warna biru muda, dan label bobot pada setiap sisi ditampilkan.

D. Simulasi ROS Motion Planning

Simulasi ROS Motion Planning yang mencakup Path Searching dan Trajectory Optimization menunjukkan perbedaan signifikan antara algoritma yang digunakan, seperti Dijkstra, A*, dan Greedy Best-First Search (GBFS). Dijkstra memberikan solusi terjamin untuk jalur terpendek tetapi dengan waktu pencarian yang lebih lama, terlihat dari animasi yang menunjukkan jalur lebih panjang dengan banyak langkah. Sebaliknya, A* lebih efisien dengan penggunaan heuristik yang mengarahkan pencarian ke jalur lebih menjanjikan, yang tercermin dalam animasi dengan jalur lebih langsung dan waktu pencarian lebih singkat. GBFS, meskipun cepat, sering kali menghasilkan jalur yang tidak optimal, dengan visualisasi yang menunjukkan jalur berkelok-kelok.

Pada sisi optimasi trajektori, simulasi menampilkan gerakan robot yang halus dan terencana, meminimalkan energi dan waktu perjalanan sambil menghindari rintangan. Hasil animasi menunjukkan robot mengikuti jalur yang direncanakan dengan tepat, serta mampu menyesuaikan gerakan terhadap batasan fisik dan lingkungan. Secara keseluruhan, pemilihan algoritma yang tepat bergantung pada kebutuhan aplikasi, apakah mencari jalur terpendek, kecepatan pencarian, atau efisiensi pergerakan robot dalam lingkungan yang kompleks.