Nama : Muhammad Husain Kaasyiful Ghitha

NIM : 1103220047

Kelas TK-45-G09

Robotika dan Sistem Cerdas

Analisis Tugas Robotika Week 5

Pada tugas ini, terdapat dua bagian implementasi perencanaan jalur yang akan dianalisis. Berikut adalah perinciannya:

1. Implementasi Algoritma Perencanaan Jalur dalam Python

   Terdapat tiga algoritma yang akan diterapkan ke sebuah program dengan menggunakan bahasa Python, yaitu Djikstra, A*, dan Cell Decomposition.

   a. Algoritma Djikstra

      Algoritma Djikstra adalah algoritma perencanaan jalur optimal yang sepenuhnya bergantung kepada jarak sebenarnya dari titik awal ke titik tujuan, tanpa menggunakan pengukuran lain, seperti nilai heuristik. Hal ini dilakukan dengan cara mengeksplorasi ruang secara menyeluruh dan bertahap. Penjelajahan ini membuat robot bisa saja menjauh dari tujuan untuk menemukan jalur, sehingga tidak efisien apabila ruang yang dipetakan sangat besar dan rumit. Meskipun tidak efisien, algoritma ini menjamin optimalitas jalur dan bekerja sangat baik untuk graf berbobot.

   b. Algoritma A*

      Algoritma A* adalah algoritma perencanaan jalur yang menggunakan gabungan dari nilai jarak sebenarnya dengan nilai perkiraan atau heuristik. Nilai heuristik biasanya berupa jarak garis lurus (Euclidean) atau jarak garis tegak (Manhattan) dari titik saat ini ke titik tujuan. Penambahan nilai heuristik pada perhitungan jalur optimal membuat algoritma ini bisa membuat robot lebih terfokus pada tujuan dan efisien. Algoritma ini sangat cocok digunakan untuk pemetaan berbasis grid dan graf.

c. Algoritma Cell Decomposition

Algoritma Cell Decomposition adalah algoritma perencanaan jalur yang bekerja dengan cara memecah ruang bebas pemetaan menjadi bagian-bagian (sel) yang bertujuan untuk menyederhanakan proses perencanaan. Algoritma ini bekerja secara global (pemetaan penuh) sekaligus secara lokal (memecah peta menjadi sel). Walaupun implementasinya yang sederhana, kecenderungannya untuk terjebak pada minimum lokal memerlukan teknik tambahan untuk memastikan kelengkapan dari pencarian jalur yang menggunakan algoritma ini.

2. ROS Motion Planning

ROS Motion Planning adalah repository perencanaan jalur berbasis ROS (Noetic) yang mencakup pencarian jalur (menemukan jalur yang tepat dan bebas dari rintangan) dan optimisasi trajektori (memperbaiki jalur yang telah diberikan berdasarkan kinematika dan dinamika). Pada dasarnya, repository ini memuat satu ruang pemetaan yang divisualisasikan menggunakan RViz untuk konfigurasi dan interaksi, dan disimulasikan menggunakan Gazebo untuk melihat keadaan realistis dari pergerakan.

Ketika robot diberi suatu titik tujuan baru, maka robot akan bergerak dan berputar untuk mendekatkan diri ke tujuan. Berdasarkan sensor, robot mengindera rintangan dan menyiasati jalur untuk menghindari tabrakan. Selain itu, ketika titik tujuan robot diubah secara drastis, maka robot harus menyesuaikan arah dengan pertimbangan dinamika agar tidak limbung.

ROS Motion Planning juga memperagakan beberapa algoritma perencanaan jalur yang umum digunakan beserta basis ilmiahnya. Berikut penjelasannya:

a. Greedy Best First Search (GBFS): robot sangat terpaku pada arah tujuan sebelum terjebak pada minimum lokal
b. Djikstra, jalur ditentukan berdasarkan penjelajahan menyeluruh ruang sebelum mencapai tujuan
c. A*: metode paling sederhana dalam menyeimbangkan heuristik dengan optimalisasi jalur
d. Jump Point Search (JPS): jalur ditarik lurus setiap sebelum rintangan
e. Dynamic A* (D*): perencanaan diperbarui berdasarkan bagian yang telah diketahui.
f. Lifelong Planning A* (LPA*): perencanaan diperbarui secara berkala

g. D* Lite: seperti D* namun dengan overhead yang lebih kecil
h. Voronoi: algoritma Voronoi menentukan jalur berdasarkan kedekatannya pada rintangan, yang membuat robot tercegah dari tabrakan.
i. Theta*: perancangan jalur menggunakan algoritma ini memungkinkan perencanaan yang memiliki pergerakan yang bebas
j. Lazy Theta*: sama seperti Theta*, namun rintangan hanya diperiksa ketika diperlukan
k. Hybrid A*: gabungan antara algoritma A* dengan penentuan dinamika robot yang lokal
l. Rapidly-exploring Random Tree (RRT): algoritma yang menggunakan "sambaran" sampel untuk mencari jalur dengan cepat
m. RRT*: sama dengan RRT, namun dengan optimalisasi yang lebih baik.
n. Informed RRT: juga sama dengan RRT, namun dengan pertimbangan heuristik area.
o. RRT-Connect: versi dari RRT yang menerapkan koneksi dua pohon
p. Ant Colony Optimization (ACO): algoritma ini mencari titik jalur yang optimal layaknya serbuan semut.
q. Genetic Algorithm (GA): algoritma ini menentukan jalur dengan cara membari titik mutatif yang diseleksi secara optimal.
r. Particle Swarm Optimisation (PSO): mirip dengan ACO, cocok untuk lingkungan yang kompleks.
s. Proportional-Integral-Derivative (PID): bekerja secara lokal dengan mengatur dinamika robot itu sendiri.
t. LQR (Linear Quadratic Regulator): Algoritma perencanaan jalur berbasis model fungsi yang mengoptimalkan jalur.
u. Dynamic Window Approach (DWA): algoritma yang mementingkan ketepatan kecepatan, cocok untuk aplikasi kecepatan tinggi.
v. Artificial Potential Field (APF): algoritma ini bekerja dengan menolak rintangan dan mendekatkan diri ke tujuan. Implementasi yang sederhana.
w. RPP (Randomized Path Planning): Algoritma yang menggunakan sampel acak untuk menemukan jalur.
x. MPC (Model Predictive Control): Algoritma yang menentukan jalur optimal berdasarkan bekal pemodelan sistem.