TUGAS ROBOTIC WEEK 13

Nama : Dimas Ahmad

Nim : 1103210218

Kelas : TK45GAB09

Analisis google colab :

Kode dalam file tersebut merupakan implementasi **Extended Kalman Filter (EKF)** yang dirancang untuk melacak posisi objek bergerak berdasarkan model gerak dan pengamatan. Pada bagian awal, kode melakukan inisialisasi parameter, seperti waktu langkah (dt), keadaan awal (state), matriks kovariansi awal (covariance), noise proses (process_noise), dan noise pengamatan (measurement_noise). Semua parameter ini digunakan untuk mengelola estimasi dan ketidakpastian dalam EKF. Fungsi motion_model digunakan untuk memodelkan bagaimana posisi (x, y) dan orientasi (theta) objek diperbarui berdasarkan kontrol input berupa kecepatan linier (v) dan kecepatan sudut (omega). Selain itu, fungsi jacobian_motion digunakan untuk menghitung Jacobian dari model gerak, yang penting untuk memperkirakan pengaruh perubahan keadaan terhadap kesalahan prediksi.

Untuk mendapatkan informasi dari pengamatan sensor, kode ini menggunakan fungsi measurement_model, yang memodelkan pengamatan posisi (x, y) dari keadaan saat ini. Fungsi ini dilengkapi dengan jacobian_measurement, yang menghitung Jacobian dari model pengamatan, digunakan untuk mengukur sensitivitas pengamatan terhadap perubahan keadaan. Dalam proses EKF, langkah prediksi (Prediction Step) memanfaatkan fungsi motion_model untuk memperkirakan keadaan berikutnya berdasarkan kontrol input, sementara Jacobian digunakan untuk mempropagasi kovariansi kesalahan menuju langkah berikutnya. Pada langkah pembaruan (Update Step), pengamatan sensor, seperti GPS, digunakan untuk memperbarui estimasi posisi. Proses ini menggunakan Jacobian dari model pengamatan untuk menghitung Kalman gain, sehingga estimasi diperbarui dengan menggabungkan prediksi model dan data sensor.

Kode ini sangat berguna untuk aplikasi pelacakan objek bergerak, seperti robot atau kendaraan, terutama ketika sistem memiliki dinamika non-linear yang mendekati linear dalam skala kecil. EKF memberikan solusi yang efektif dengan mengurangi ketidakpastian estimasi menggunakan pengamatan sensor yang memiliki noise. Jika diperlukan, saya dapat membantu menjelaskan bagian tertentu atau menjalankan analisis lebih lanjut pada kode tersebut.

Analisis Webots :

Kode ini bertujuan untuk mengontrol pergerakan sebuah robot agar dapat melewati serangkaian titik tujuan atau waypoints yang telah ditentukan. Robot dilengkapi dengan perangkat keras seperti motor, rem, GPS, akselerometer, IMU (Inertial Measurement Unit), dan sensor posisi. Fungsi utama dari program ini adalah mengatur navigasi robot dengan memanfaatkan algoritma kinematika sederhana, state machine, serta berbagai sensor untuk menentukan posisi dan orientasi robot dalam lingkungan.

Struktur utama kode terdiri dari beberapa bagian. Pertama, modul-modul penting seperti controller, math, dan numpy diimpor, meskipun numpy tidak digunakan dalam kode ini. Selanjutnya, perangkat keras robot diinisialisasi menggunakan metode getDevice() untuk mengakses motor, rem, sensor posisi, akselerometer, IMU, dan GPS. Selain itu, berbagai parameter robot seperti kecepatan maksimum, radius roda, jarak antar roda, serta daftar waypoint juga didefinisikan untuk mendukung navigasi.

Navigasi robot dilakukan dengan mekanisme state machine yang memiliki sepuluh state (dari state 0 hingga state 9), di mana setiap state mewakili tindakan tertentu seperti pengereman, berbelok ke kiri atau kanan, mengarahkan roda, dan bergerak lurus. Fungsi moveToWaypoint() digunakan untuk menentukan pergerakan robot berdasarkan perbedaan sudut antara posisi robot saat ini dan arah menuju waypoint berikutnya. Robot akan terus memperbarui posisinya menggunakan data dari sensor posisi dan GPS, serta menghitung kecepatan menggunakan akselerometer.

Program ini juga menghitung rata-rata (mean) dan variansi (variance) dari kesalahan posisi robot terhadap koordinat GPS untuk mengevaluasi performa navigasi. Data ini dihitung dalam fungsi getNewMeanOD() dan getNewVarianceOD(). Pada setiap siklus waktu, program akan membaca data dari sensor, memproses data tersebut untuk memperbarui posisi dan orientasi, lalu menjalankan tindakan yang sesuai berdasarkan state yang sedang aktif. Jika waypoint saat ini tercapai, robot akan beralih ke waypoint berikutnya hingga semua waypoint selesai dilalui.

Secara keseluruhan, kode ini merupakan implementasi yang kompleks untuk mengatur pergerakan robot secara otonom dengan memanfaatkan berbagai sensor dan aktuator. Namun, ada beberapa aspek yang perlu diperhatikan, seperti ketidakkonsistenan dalam perhitungan theta dan penggunaan modul numpy yang tidak relevan. Selain itu, optimisasi logika state machine dan dokumentasi yang lebih rinci dapat meningkatkan keterbacaan dan efisiensi kode.