Nama : Eko Putra Nugraha

NIM : 1103213212

Analisis Google Collab

Rangkaian simulasi dan visualisasi yang telah kita analisis menggambarkan kemampuan berbagai varian filter Kalman dalam melakukan estimasi posisi objek bergerak dengan memanfaatkan data sensor yang bernoise. Pada awalnya, kita melihat bagaimana Extended Kalman Filter (EKF) berhasil mengestimasi jalur objek yang bergerak dengan kombinasi data GPS yang bernoise dan model IMU. EKF mampu menghaluskan noise GPS dan mengikuti pola umum jalur sebenarnya, meskipun masih ada deviasi dan membutuhkan waktu untuk konvergensi awal. Selanjutnya, Unscented Kalman Filter (UKF) menunjukkan peningkatan performa, dengan estimasi yang lebih halus dan akurat dibandingkan EKF, terutama dalam menangani pergerakan nonlinear objek. Baik EKF maupun UKF menunjukkan keunggulan filter Kalman dalam mengatasi ketidakpastian sensor dan memberikan estimasi yang lebih baik daripada hanya menggunakan data sensor mentah. Kemudian, kita melihat implementasi Kalman Filter sederhana untuk tracking objek dengan gerakan sinusoidal, dimana filter berhasil menghaluskan noise sensor dan mengikuti pola gerakan objek dengan baik, meskipun ada deviasi pada perubahan arah yang tajam. Terakhir, kita menganalisis Kalman Filter untuk tracking drone dengan gerakan parabola, yang menunjukkan hasil yang sangat akurat, dengan garis estimasi filter hampir menempel pada jalur sebenarnya, memvalidasi efektivitas Kalman Filter dalam memodelkan gerakan yang kompleks dan mengurangi dampak noise sensor. Secara keseluruhan, analisis ini mengilustrasikan bagaimana filter Kalman, dalam berbagai implementasinya, adalah alat yang ampuh untuk estimasi posisi dan tracking objek bergerak dalam lingkungan yang penuh dengan noise, dengan kemampuan mengkombinasikan model gerak objek dan data sensor untuk menghasilkan estimasi yang lebih akurat dan andal. Setiap filter memiliki karakteristik unik dan kelebihan dalam menangani jenis pergerakan objek dan noise yang berbeda, tetapi prinsip dasarnya tetap sama, yaitu memanfaatkan informasi dari model gerak dan data sensor untuk menghasilkan estimasi terbaik.

Analisis Weebots

Analisis kf_variance.py

Kode kf_variance.py mengimplementasikan sebuah pengendali untuk robot Webots, yang berfokus pada navigasi *waypoint* dengan mempertimbangkan estimasi posisi dan variansnya. Kode ini melakukan inisialisasi berbagai perangkat keras robot, termasuk motor roda (belakang dan depan), motor kemudi, rem, sensor posisi roda (odometer), akselerometer, IMU (Inertial Measurement Unit), dan GPS. Implementasi navigasi waypoint menggunakan serangkaian *state* untuk mengatur perilaku robot dalam mencapai target: pengereman, kemudi, dan pergerakan lurus. Perhitungan mean dan varians posisi berdasarkan input sensor odometer (diintegrasikan dengan konsep *dead reckoning*) dan GPS yang diimplementasikan dengan pendekatan rekursif. Pergerakan robot diatur oleh fungsi moveToWaypoint, yang menentukan set kecepatan motor dan posisi kemudi. Kode ini juga menggunakan variabel seperti tp (waktu fase dalam siklus navigasi), phi (orientasi robot), alpha (sudut ke target), dan theta (selisih sudut). Secara keseluruhan, kode ini adalah implementasi dasar dari sistem navigasi robot yang menggabungkan estimasi posisi dengan *waypoint following*.

Analisis pos-prediction.py

Kode pos-prediction.py adalah *entry point* dari sistem robot yang lebih kompleks. Ia menggunakan dua *class*, RobotManager dan ArgsManager. ArgsManager kemungkinan menangani penguraian argumen baris perintah (command line arguments) atau konfigurasi, dan RobotManager tampaknya mengelola siklus eksekusi robot, termasuk inisialisasi robot dan perangkat kerasnya, dan mengendalikan simulasinya. Kode ini juga menggunakan numpy untuk perhitungan numerik. Tidak seperti kf_variance.py, kode ini tidak secara langsung mengimplementasikan logika kontrol atau navigasi robot. Sebaliknya, ia bertindak sebagai *framework* untuk menjalankan tugas robot. Penggunaan np.random.seed() menandakan bahwa kode ini mungkin melibatkan simulasi atau percobaan yang memerlukan hasil yang deterministik dan dapat diulang. Secara keseluruhan, pos-prediction.py merupakan kode tingkat tinggi untuk menjalankan simulasi dan mengelola robot, sementara detail kontrol robot dienkapsulasi dalam *class* RobotManager dan kemungkinan *module* lainnya.