Nama : Muhammad Husain Kaasyiful Ghitha

NIM : 1103220047

Kelas TK-45-G09

Robotika dan Sistem Cerdas

Analisis Tugas Robotika Week 12

Pada tugas ini, dilaksanakan beberapa peragaan teori Localization and Filtering menggunakan bahasa Python dengan library numpy dan matplotlib. Praktik-praktik yang dilakukan adalah sebagai berikut:

- Implementasi Filter Kalman untuk Estimasi Posisi Robot.
- Implementasi Filter Partikel untuk Estimasi Posisi Robot.
- Implementasi Lokalisasi dengan Sensor IMU dan Lidar.
- Implementasi Simulasi Extended Kalman Filter untuk Navigasi.
- Implementasi Filter Partikel untuk Navigasi.

Selain memrogram kode Python untuk simulasi tersebut, terdapat juga simulasi yang berfokus pada implementasinya pada robot menggunakan simulator Webots. Simulasi yang dilakukan pada Webots tersebut adalah estimasi robot E-puck menggunakan Kalman Filter.

Berikut adalah pembahasan mengenai kode Python dan simulasi-simulasi yang dilakukan.

1. Membuat kode Python menggunakan numpy dan matplotlib untuk simulasi mengenai Lokalisasi dan Filtering.

   Pada kode simulasi ini, penjelasan fungsi-fungsi yang dimuat di program adalalah sebagai berikut:

   a. Implementasi Filter Kalman untuk Estimasi Posisi Robot.

      Kode ini menerapkan algoritma Kalman Filter untuk memperkirakan posisi dari suatu objek berdasarkan pengukuran yang terkontaminasi oleh noise. Pertama, kode ini mendefinisikan variabel-variabel penting, termasuk matriks sistem A, matriks input kontrol B, dan matriks pengukuran H. Kode ini juga mendefinisikan nilai-

nilai awal, termasuk posisi awal x, kovarians awal P, noise sistem Q, dan noise pengukuran R. Selanjutnya, kode ini membuat garis waktu untuk simulasi dan menggenerasikan pengukuran posisi yang mengandung noise.

Dalam loop, kode ini melakukan dua langkah utama dari Kalman Filter: prediksi dan koreksi. Pada langkah prediksi, posisi dan kovarians diprediksi berdasarkan model sistem. Pada langkah koreksi, pengukuran aktual digunakan untuk memperbarui estimasi posisi dan kovarians dengan menghitung gain Kalman. Hasil estimasi posisi ini kemudian ditampilkan bersama dengan posisi sebenarnya dan posisi yang diukur untuk membandingkan efektivitas dari Kalman Filter dalam mengestimasi posisi yang sebenarnya.

Dapat dilihat bahwa Kalman Filter mampu memperbaiki perkiraan posisi robot dengan membuat estimasi dari hasil pengukuran menjadi lebih stabil, dengan garis yang tampak lebih mulus dibandingkan dengan hasil pengukuran. Terlihat juga bahwa noise pengukuran dapat diredam oleh Filter, dengan garis estimasi yang tidak sering bersilangan dengan garis posisi sebenarnya dibandingkan dengan garis pengukuran yang penuh dengan noise.

b. Implementasi Filter Partikel untuk Estimasi Posisi Robot.

Kode ini menerapkan algoritma Particle Filter untuk mengestimasi posisi objek berdasarkan pengukuran yang terkontaminasi oleh noise. Kode ini dimulai dengan menginisialisasi 1000 partikel dengan posisi acak dalam rentang tertentu. Pengukuran diterima dengan penambahan noise untuk mensimulasikan ketidakpastian pengukuran sebenarnya.

Dalam loop, kode ini memperbarui bobot partikel berdasarkan pengukuran yang diterima, dengan bobot yang lebih besar diberikan pada partikel yang lebih dekat dengan pengukuran. Setelah pembaruan bobot, kode melakukan resampling untuk memilih partikel berdasarkan bobot yang telah diperbarui, dengan partikel yang memiliki bobot lebih besar memiliki peluang lebih tinggi untuk dipilih. Estimasi posisi akhir diambil sebagai rata-rata dari partikel-partikel yang disampel ulang, dan hasilnya dipetakan dalam histogram untuk menggambarkan distribusi partikel dan posisi estimasi akhir.

Output dari kode ini menunjukkan distribusi partikel yang telah diresampling dan posisi akhir yang diestimasi sebagai garis vertikal merah pada histogram. Posisi akhir yang diestimasi mendekati nilai rata-rata dari pengukuran yang diterima (yaitu

2 ditambah atau dikurang dengan noise pengukuran). Secara keseluruhan, kode ini menerapkan algoritma Particle Filter dengan cukup baik untuk memperkirakan posisi berdasarkan pengukuran yang dipenuhi noise.

c. Implementasi Lokalisasi dengan Sensor IMU dan Lidar.

Kode ini mengimplementasikan Sensor Fusion untuk menggabungkan data dari dua sensor berbeda, yaitu IMU dan Lidar, untuk mengestimasi posisi objek. Data IMU disimulasikan dengan kecepatan konstan, namun ditambahkan noise normal dengan standar deviasi 0.1 untuk mensimulasikan ketidakakuratan pengukuran. Data Lidar disimulasikan dengan jarak sebenarnya yang bervariasi secara sinusoidal ditambah noise normal dengan standar deviasi 0.5 untuk mensimulasikan ketidakpastian pengukuran.

Metode Weighted Average digunakan untuk menggabungkan data dari kedua sensor. Bobot $\alpha = 0.8$ diberikan pada data IMU dan bobot $\beta = 0.2$ diberikan pada data Lidar, mencerminkan kepercayaan lebih besar pada data IMU. Hasil akhir dari penggabungan data ini adalah estimasi posisi yang lebih akurat dibandingkan jika hanya menggunakan salah satu sensor saja. Hasil estimasi ini kemudian dipetakan bersama dengan data IMU dan Lidar untuk memberikan visualisasi efektivitas sensor fusion dalam estimasi posisi.

Berdasarkan hasil yang ditampilkan, dengan menggunakan sensor fusion, posisi yang diestimasi menjadi lebih akurat dibandingkan jika hanya menggunakan salah satu sensor saja. Hal ini karena sensor fusion menggabungkan kekuatan dari kedua sensor, mengurangi efek noise, dan meningkatkan keandalan estimasi posisi.

d. Implementasi Simulasi Extended Kalman Filter untuk Navigasi.

Kode ini menerapkan algoritma Extended Kalman Filter (EKF) untuk mengestimasi posisi robot dalam sistem non-linear. Fungsi robot_model mendefinisikan model kinematika robot, di mana input u digunakan untuk memperbarui posisi dan orientasi robot. Fungsi ekf_predict melakukan prediksi posisi dan kovarians berdasarkan model sistem non-linear, sementara fungsi ekf_update melakukan pembaruan posisi dan kovarians berdasarkan pengukuran yang diterima.

Selama simulasi, posisi robot diprediksi dan pengukuran dengan noise ditambahkan. Setiap langkah iterasi melibatkan prediksi posisi baru dan pembaruan berdasarkan pengukuran aktual. Hasilnya adalah tiga jalur: jalur sebenarnya (true_path), jalur yang diukur (measured_path), dan jalur estimasi yang dihasilkan oleh EKF (estimated_path). Jalur-jalur ini kemudian divisualisasikan menggunakan matplotlib.

Pada plot yang dikeluarkan oleh program, terlihat tiga jalur: hijau yang menggambarkan pergerakan robot yang bersifat non-linear, merah untuk hasil pengukuran posisi, dan biru untuk posisi estimasi dari EKF di koordinat posisi robot di ruang 2D. Jalur yang diestimasi (biru), meskipun masih bersifat kasar dan patah-patah, namun lebih mendekati jalur sebenarnya dibandingkan jalur yang diukur (merah), mencerminkan akurasi EKF dalam mengestimasi posisi berdasarkan pengukuran dan model sistem.

e. Implementasi Particle Filter untuk Navigasi.

Kode ini mengimplementasikan Particle Filter untuk memperkirakan posisi robot dalam dua dimensi. Kode dimulai dengan mendefinisikan jumlah partikel (N) dan inisialisasi partikel-partikel secara acak dalam ruang dua dimensi. Posisi robot yang sebenarnya dimodelkan dan diperbarui dengan menggunakan fungsi `move_robot` yang mempertimbangkan kecepatan dan sudut pergerakan. Noise sensor ditambahkan pada pengukuran untuk mensimulasikan ketidakakuratan pengukuran aktual.

Setelah melakukan simulasi gerakan robot, filter partikel diterapkan. Setiap partikel diperbarui bobotnya berdasarkan jarak dari pengukuran yang sebenarnya, dan proses resampling dilakukan untuk memilih partikel baru berdasarkan bobot yang diperbarui. Posisi akhir yang diestimasi diambil sebagai rata-rata dari partikel-partikel yang diresample. Visualisasi hasil menunjukkan distribusi partikel, posisi estimasi, dan posisi sebenarnya, memberikan gambaran bagaimana filter partikel dapat digunakan untuk mengestimasi posisi robot dengan baik dalam kondisi noise pengukuran.

Berdasarkan visualisasi, dapat dilihat bagaimana partikel sampel tersebar di ruang, namun filter mampu menunjukkan posisi yang akurat dari robot. Posisi yang

diestimasi (merah) cukup dekat dengan posisi sebenarnya (biru), menunjukkan bahwa Particle Filter berhasil dalam estimasi posisi robot meskipun ada noise dalam pengukuran.

2. Simulasi Webots Implementasi Kalman Filter untuk Lokalisasi Robot.

Kode controller yang diberikan mengimplementasikan Kalman Filter dalam lingkungan simulasi Webots untuk memperkirakan posisi robot. Pada awalnya, parameter Webots diatur dengan TIME_STEP sebesar 32 milidetik. Robot diinisialisasi dengan sensor roda, termasuk motor roda kiri dan kanan yang disetel ke mode kecepatan tanpa batas (float('inf')), dan diberi kecepatan awal 0.0. Sensor encoder pada kedua roda diaktifkan untuk mengambil data gerakan roda, serta sensor jarak diaktifkan untuk pengukuran posisi.

Fungsi kalman_filter didefinisikan untuk memprediksi dan memperbarui posisi robot. Dalam fungsi ini, langkah prediksi (x_pred) diperoleh dengan menambahkan input pergerakan u ke posisi saat ini x. Kovarians prediksi (P_pred) diperbarui dengan menambahkan noise proses. Pada langkah koreksi, Gain Kalman (K) dihitung berdasarkan kovarians prediksi, dan posisi baru (x_new) diperbarui menggunakan pengukuran z. Kovarians juga diperbarui untuk mencerminkan ketidakpastian baru.

Dalam loop utama, nilai encoder dari kedua roda diambil untuk memperkirakan pergerakan robot (u), yang merupakan rata-rata dari jarak yang ditempuh oleh kedua roda. Pengukuran dari sensor jarak (z) juga diambil untuk memperbarui estimasi posisi. Fungsi kalman_filter kemudian diterapkan untuk memperbarui posisi estimasi (x) dan ketidakpastian (P). Hasil estimasi posisi robot dicetak pada setiap iterasi loop.

Secara keseluruhan, kode ini menunjukkan penggunaan Kalman Filter untuk memperbaiki estimasi posisi robot dalam simulasi Webots, dengan memanfaatkan data dari sensor roda dan sensor jarak. Ini adalah contoh dari teknik filter Kalman yang menggabungkan prediksi model sistem dengan pengukuran sensor untuk mendapatkan estimasi yang lebih akurat.

Output dari program ini akan menampilkan estimasi posisi robot yang diperbarui setiap 32 milidetik, berdasarkan Kalman Filter. Pada setiap iterasi loop, estimasi posisi robot akan dicetak ke konsol. Karena Kalman Filter menggabungkan prediksi berdasarkan model gerakan dan pembaruan berdasarkan pengukuran sensor jarak, hasil estimasi posisi akan semakin akurat seiring berjalannya waktu, meskipun terdapat noise pada pengukuran. Pada awalnya, karena posisi awal dan ketidakpastian

awal, estimasi mungkin tidak akurat. Namun, dengan setiap iterasi dan pengukuran baru, Kalman Filter memperbaiki estimasi tersebut. Hasil akhirnya adalah estimasi posisi yang konvergen mendekati posisi sebenarnya dari robot, mengurangi dampak noise dari pengukuran sensor