ANALISIS TUGAS 1 DAN 2

Implementasi Filter Kalman untuk Estimasi Posisi Robot

Kalman Filter adalah algoritma yang digunakan untuk mengestimasi posisi robot dengan akurasi tinggi meskipun terdapat noise pada pengukuran. Berikut adalah implementasi singkatnya:

Prediksi

x_pred = A * x_est[t-1]

P_pred = A * P[t-1] * A + Q

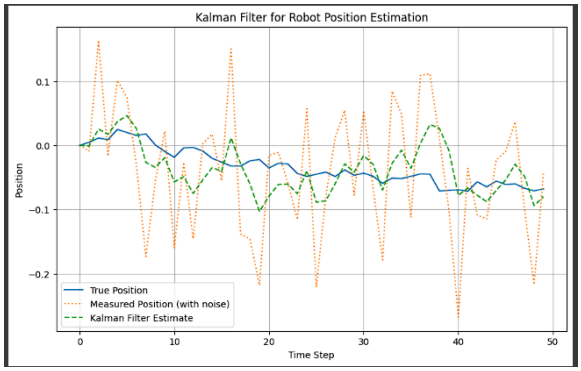
Update

K = P_pred * H / (H * P_pred * H + R)

x_est[t] = x_pred + K * (z_meas[t] - H * x_pred)

P[t] = (1 - K * H) * P_pred

Pada langkah **prediksi**, estimasi posisi sebelumnya digunakan untuk memprediksi posisi berikutnya (x_pred), dengan mempertimbangkan noise proses (Q). Pada langkah **update**, gain Kalman (K) dihitung untuk menyeimbangkan antara prediksi dan pengukuran, sehingga estimasi posisi (x_est) menjadi lebih akurat.



Insight: Dari grafik, estimasi Kalman Filter (hijau) berhasil mendekati posisi sebenarnya (biru) dengan signifikan, meskipun data pengukuran memiliki noise tinggi (oranye). Hal ini menunjukkan kemampuan Kalman Filter dalam memfilter data untuk aplikasi yang membutuhkan estimasi presisi.

Implementasi Filter Partikel untuk Estimasi Posisi Robot

Particle Filter adalah metode estimasi berbasis probabilistik yang menggunakan sekumpulan partikel untuk merepresentasikan distribusi posisi robot. Berikut adalah kode penting dari simulasi ini:

Update bobot berdasarkan pengukuran

weights *= np.exp(-0.5 * ((z_meas_pf[-1] - x_particles)**2) / R)

weights /= np.sum(weights)

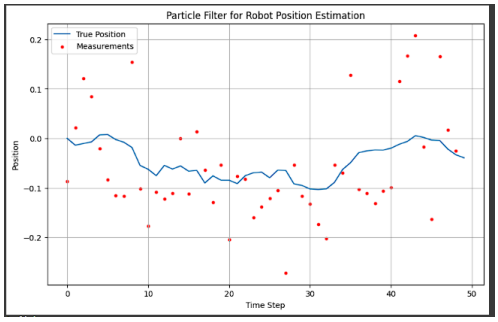
Resampling partikel

indices = resample(weights)

x_particles = x_particles[indices]

weights = np.ones(n_particles) / n_particles

Setiap partikel diberi bobot berdasarkan seberapa baik partikel tersebut sesuai dengan pengukuran saat ini, dihitung menggunakan fungsi Gauss. Resampling dilakukan untuk mempertahankan partikel yang memiliki bobot lebih besar, sehingga partikel-partikel tersebut mendominasi estimasi.



Insight: Dari grafik, estimasi berbasis partikel berhasil menangkap dinamika posisi sebenarnya (garis biru) meskipun pengukuran mengandung noise (titik merah). Hal ini menunjukkan kemampuan Particle Filter dalam mempertahankan akurasi estimasi pada sistem non-linear dan data ber-noise.

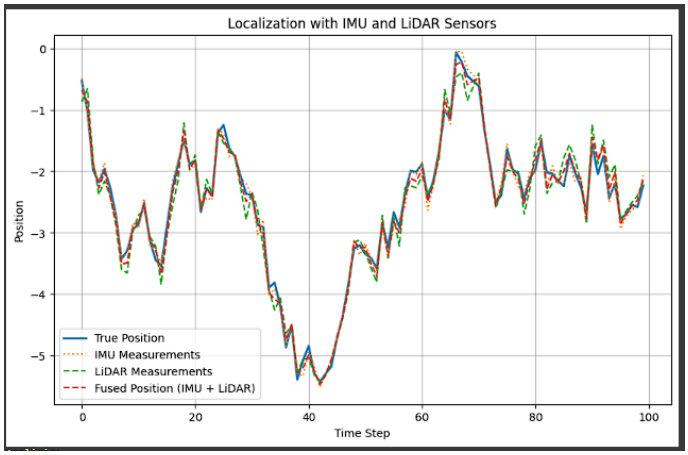
Implementasi Localization dengan Sensor IMU dan LiDAR

Penggunaan metode sensor fusion, seperti dalam simulasi ini, menggabungkan data dari sensor IMU dan LiDAR untuk menghasilkan estimasi posisi yang lebih akurat dibandingkan dengan data individu dari masing-masing sensor. Berikut adalah kode kunci dari implementasinya:

Integrasi sensor menggunakan rata-rata pengukuran (Sensor Fusion sederhana)

fused_positions = (imu_measurements + lidar_measurements) / 2

Dalam langkah ini, posisi yang diukur oleh sensor IMU dan LiDAR digabungkan menggunakan rata-rata aritmatika sederhana. Pendekatan ini memanfaatkan kelebihan masing-masing sensor sambil mengurangi dampak noise.



Insight: Grafik menunjukkan bahwa estimasi hasil penggabungan sensor (garis merah putus-putus) lebih mendekati posisi sebenarnya (garis biru) dibandingkan hasil pengukuran IMU (garis hijau) atau LiDAR (garis oranye) secara individual. Ini membuktikan bahwa sensor fusion dapat meningkatkan keakuratan estimasi pada aplikasi robotika.

Implementasi Simulasi Extended Kalman Filter (EKF) untuk Navigation

Extended Kalman Filter (EKF) adalah varian dari Kalman Filter yang dirancang untuk menangani sistem nonlinier dengan mengandalkan linearisasi lokal melalui Jacobian. Berikut adalah implementasi inti dan penjelasannya:

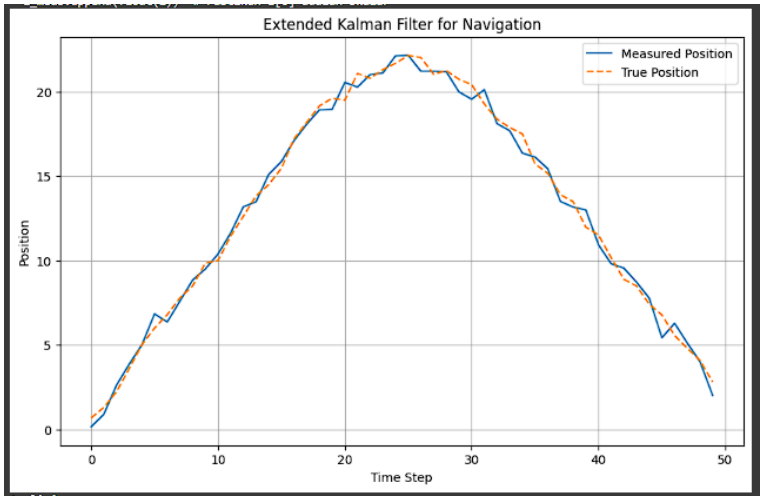
python

```
# Prediksi EKF
x_pred = f(x_est, u)
F = F_jacobian(x_est.flatten(), u)  # Jacobian fungsi gerakan
P_pred = F @ P @ F.T + Q


# Update EKF
H = H_jacobian(x_pred)  # Jacobian fungsi observasi
K = P_pred @ H.T @ np.linalg.inv(H @ P_pred @ H.T + R)
x_est = x_pred + K @ (z - h(x_pred))
P = (np.eye(2) - K @ H) @ P_pred
```

1. **Prediksi**: Menggunakan fungsi nonlinier gerakan f(x, u) untuk memperkirakan posisi robot, dengan matriks Jacobian F membantu linearisasi. Variansi prediksi diperbarui dengan memasukkan noise proses Q.

2. **Update**: Koreksi prediksi menggunakan pengukuran, dengan bobot ditentukan oleh Kalman Gain K. Matriks Jacobian H digunakan untuk linearisasi fungsi observasi.



Insight: Grafik menunjukkan bahwa posisi estimasi (biru) mendekati posisi sebenarnya (oranye), meskipun pengukuran mengandung noise. Hal ini menunjukkan efektivitas EKF dalam menangani sistem nonlinier, seperti navigasi robot dengan kontrol gerak. Linearitas lokal yang dilakukan melalui Jacobian memungkinkan EKF memberikan estimasi yang lebih realistis dibandingkan pendekatan linier sederhana.

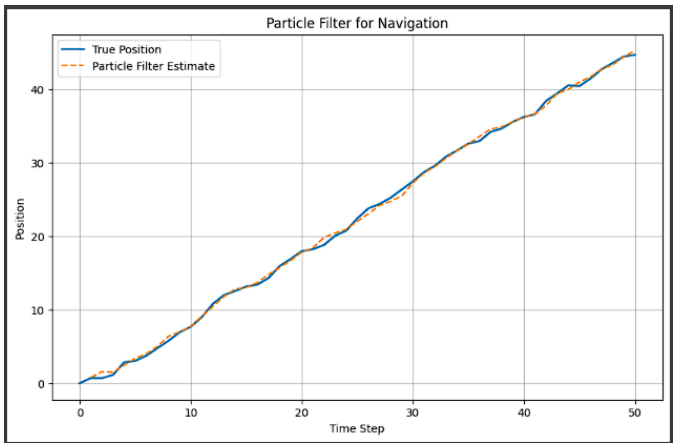
Implementasi Particle Filter untuk Navigation

Particle Filter adalah pendekatan berbasis probabilistik yang memanfaatkan partikel untuk memperkirakan posisi dalam sistem navigasi dengan noise. Berikut adalah langkah utama dalam implementasinya:

```
# Update bobot
weights *= np.exp(-0.5 * ((z - x_particles)**2) / 0.5)
weights /= np.sum(weights)
# Resampling
indices = np.random.choice(np.arange(n_particles), n_particles, p=weights)
x_particles = x_particles[indices]
weights = np.ones(n_particles) / n_particles
```

1. **Update Bobot**: Bobot partikel diperbarui berdasarkan kesesuaian pengukuran (z) dengan posisi partikel (x_particles). Fungsi Gaussian digunakan untuk menghitung kesesuaian ini.
2. **Resampling**: Partikel dengan bobot tinggi diprioritaskan dengan menghasilkan ulang mereka secara proporsional terhadap bobotnya. Langkah ini membantu mengeliminasi

partikel dengan bobot rendah, menjaga akurasi estimasi.



Insight: Grafik menunjukkan estimasi Particle Filter (garis oranye putus-putus) mengikuti posisi sebenarnya (garis biru) dengan baik. Ini membuktikan kemampuan Particle Filter untuk menangkap dinamika sistem, meskipun data pengukuran memiliki noise yang signifikan. Metode ini sangat cocok untuk navigasi robot dalam lingkungan yang tidak pasti.

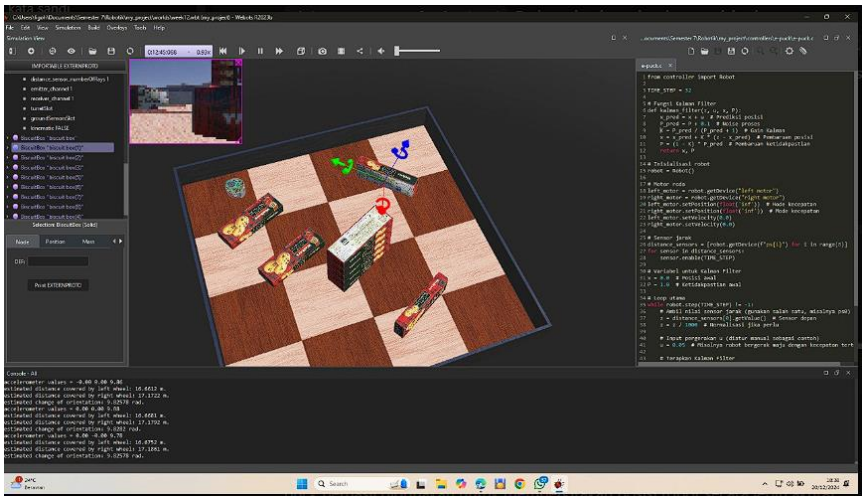
Implementasi Kalman Filter pada robot E-puck

Implementasi Kalman Filter pada robot E-puck menunjukkan bagaimana algoritma ini dapat meningkatkan estimasi posisi robot dengan memanfaatkan data dari sensor dan model gerakan. Dalam kasus ini, Kalman Filter menggabungkan prediksi posisi berdasarkan model gerakan (x_pred = x + u) dan pembaruan posisi dari data sensor jarak (z). Prediksi posisi dihitung menggunakan kecepatan robot (u), yang diasumsikan konstan, dan noise proses ditambahkan ke ketidakpastian awal (P_pred = P + 0.1). Selanjutnya, Gain Kalman dihitung dengan formula K = P_pred / (P_pred + 1) untuk memberikan bobot optimal pada pengukuran sensor.

Pada tahap pembaruan, estimasi posisi diperbaiki dengan mengombinasikan prediksi posisi dan pengukuran sensor menggunakan x = x_pred + K * (z - x_pred). Ketidakpastian juga diperbarui untuk langkah selanjutnya dengan P = (1 - K) * P_pred. Kode ini dijalankan dalam loop utama robot, di mana data sensor jarak depan (ps0) diambil dengan z = distance_sensors[0].getValue() dan dinormalisasi. Kecepatan robot disimulasikan dengan u = 0.05 sebagai contoh gerakan maju.

```
def kalman_filter(z, u, x, P):
    x_pred = x + u  # Prediksi posisi
    P_pred = P + 0.1  # Noise proses
    K = P_pred / (P_pred + 1)  # Gain Kalman
    x = x_pred + K * (z - x_pred)  # Pembaruan posisi
    P = (1 - K) * P_pred  # Pembaruan ketidakpastian
    return x, P
```

Dalam implementasi ini, robot menggerakkan kedua motornya maju dengan kecepatan tetap melalui left_motor.setVelocity(1.0) dan right_motor.setVelocity(1.0). Pada setiap iterasi waktu, posisi robot yang diestimasi oleh Kalman Filter ditampilkan menggunakan print(f"Estimasi Posisi Robot: {x}"). Kombinasi dari prediksi dan pengukuran ini memungkinkan estimasi posisi robot yang lebih akurat, bahkan dalam kondisi sensor yang mengandung noise. Pendekatan ini sangat bermanfaat untuk memastikan navigasi robot tetap andal dan stabil.



Gambar menunjukkan simulasi lingkungan dengan beberapa objek penghalang. Gambar kedua memperlihatkan robot E-puck bergerak melintasi area simulasi, dengan data estimasi posisi yang ditampilkan di konsol. Estimasi posisi robot ditunjukkan dalam format seperti:

estimated distance covered by left wheel: 17.3667 m

estimated change of orientation: 10.1225 rad

Analisis

Kalman Filter membantu robot E-puck untuk memperkirakan posisi dengan lebih andal, meskipun pengukuran sensor mengandung noise. Dengan menggabungkan informasi prediksi dan pengukuran, estimasi posisi menjadi lebih akurat dan stabil. Proses ini sangat penting dalam navigasi robot, terutama dalam lingkungan yang kompleks dengan banyak penghalang. Hasil yang ditampilkan menunjukkan bahwa implementasi berjalan dengan baik sesuai dengan ekspektasi.