1. Simulasi Gerakan Maju dengan Open-Loop Control

Pada simulasi gerakan maju, robot e-puck menggunakan kontrol loop terbuka untuk bergerak lurus ke depan tanpa umpan balik dari sensor. Pada kode Python yang digunakan, berikut langkah-langkah utama yang dilakukan:

- **Inisialisasi robot dan komponen**: Kode Python menggunakan Robot() dari pustaka Webots untuk menginisialisasi robot dan komponen seperti roda kiri dan kanan.
- **Atur posisi motor menjadi infinity**: Dengan memanggil setPosition(float('inf')), motor roda tidak akan berhenti pada posisi tertentu, sehingga memungkinkan gerakan berkelanjutan.
- **Set kecepatan roda**: setVelocity(speed) digunakan untuk mengatur kecepatan roda. Pada gerakan maju, kecepatan kedua roda diatur sama, sehingga robot bergerak lurus ke depan.
- **Loop simulasi**: Bagian while robot.step(timeStep) != -1 memastikan simulasi terus berjalan setiap kali langkah waktu simulasi diproses.

Kode ini menunjukkan bahwa tanpa umpan balik sensor, robot tidak mampu menyesuaikan diri jika terjadi gangguan di lintasan, seperti objek di depan. Simulasi ini efektif untuk gerakan konstan di lingkungan yang sederhana, namun dalam skenario yang lebih kompleks, pendekatan ini memiliki keterbatasan signifikan.

2. Simulasi Gerakan Melingkar

Pada simulasi gerakan melingkar, perbedaan kecepatan antara roda kiri dan kanan menghasilkan lintasan melingkar. Berikut adalah langkah-langkah utama dalam kodenya:

- **Pengaturan posisi motor dan kecepatan roda**: Sama seperti simulasi gerakan maju, posisi roda diatur tanpa batas, tetapi pada simulasi ini, kecepatan roda kiri dan kanan berbeda. Roda kiri diatur dengan kecepatan lebih lambat dibandingkan roda kanan, menghasilkan gerakan berputar melingkar.
- **Loop simulasi**: Simulasi akan terus berjalan selama kondisi robot.step(timeStep) dipenuhi.

Kode ini memanfaatkan perbedaan kecepatan antar roda (differential drive) untuk mengatur lintasan melingkar. Meski demikian, tanpa umpan balik sensor, jika ada gangguan eksternal, robot tidak dapat mengoreksi lintasannya. Dengan kontrol closed-loop yang lebih canggih, radius lingkaran bisa lebih presisi dan menyesuaikan perubahan lingkungan.

3. Simulasi Penghentian Robot dengan Sensor Proximity

Pada simulasi ini, robot e-puck menggunakan sensor proximity untuk mendeteksi objek di depannya dan menghentikan gerakan saat objek terdeteksi. Berikut adalah langkah-langkah kodenya:

- **Inisialisasi sensor proximity**: Dengan robot.getDevice('ps0') hingga robot.getDevice('ps7'), delapan sensor proximity diaktifkan untuk mendeteksi objek di berbagai sudut di sekitar robot.
- **Aktivasi sensor proximity**: Sensor diaktifkan dengan proxSensors[i].enable(timeStep) sehingga mereka bisa memberikan umpan balik dalam setiap langkah simulasi.
- **Pembacaan sensor**: Nilai sensor proximity yang mengarah ke depan (sensor 0 dan 7) dijumlahkan untuk mendeteksi apakah ada objek di depan robot. Jika nilai gabungan melebihi ambang batas, robot menghentikan pergerakannya.
- **Kontrol kecepatan dinamis**: Jika tidak ada objek yang terdeteksi, roda terus bergerak maju dengan kecepatan konstan. Namun, ketika objek terdeteksi, kecepatan diatur ke 0 untuk menghentikan robot.

Kode ini mengimplementasikan kontrol closed-loop dengan sensor proximity sebagai umpan balik, memungkinkan robot merespons lingkungan secara dinamis. Hal ini menunjukkan keunggulan kontrol tertutup dalam situasi di mana interaksi dengan lingkungan diperlukan untuk menghindari tabrakan dan menjaga keselamatan.

Kesimpulan dan Analisis Kode

Ketiga simulasi menunjukkan perbedaan utama antara kontrol loop terbuka (open-loop) dan kontrol tertutup (closed-loop). Pada gerakan maju dan gerakan melingkar, kontrol open-loop yang diimplementasikan dalam kode cukup efektif untuk lingkungan yang statis tanpa gangguan. Namun, tanpa adanya umpan balik sensor, robot tidak dapat beradaptasi terhadap perubahan atau kendala di jalur pergerakan.

Sebaliknya, simulasi penghentian robot dengan sensor proximity menunjukkan betapa pentingnya penggunaan sensor dalam mengimplementasikan kontrol closed-loop. Sensor proximity digunakan untuk mendeteksi objek dan memberikan umpan balik yang memengaruhi pergerakan robot, menghentikannya saat dibutuhkan. Kode ini menunjukkan bagaimana kontrol closed-loop memungkinkan robot untuk beradaptasi dengan lingkungan dan mencegah tabrakan, membuatnya lebih aman dan fleksibel dalam situasi dunia nyata.

Penggunaan closed-loop control yang melibatkan sensor, seperti sensor proximity, memungkinkan robot untuk berinteraksi lebih baik dengan lingkungannya. Implementasi kontrol tertutup ini lebih cocok untuk aplikasi dunia nyata yang melibatkan lingkungan dinamis dan tidak terstruktur.