Nama : Satya Athaya Daniswara

NIM : 1103213152

Proyek "NVIDIA JetBot Collision Avoidance" bertujuan untuk mengembangkan kontroler pada robot JetBot agar dapat mendeteksi dan menghindari tabrakan dalam waktu nyata menggunakan model pembelajaran mesin. JetBot menggunakan kamera untuk memperoleh data visual, yang kemudian diproses oleh model deep learning untuk menentukan apakah ada hambatan di jalurnya. Jika hambatan terdeteksi, JetBot akan berbelok untuk menghindarinya.

Webots digunakan sebagai simulator untuk menguji JetBot dalam lingkungan virtual sebelum implementasi di dunia nyata, yang memungkinkan pengembangan dan pengujian lebih aman dan hemat biaya.

- Struktur dan Analisis Kode Penghindaran Tabrakan

Kode utama penghindaran tabrakan pada JetBot terdiri dari beberapa bagian penting, yaitu pengaturan model dan kamera, fungsi utama untuk memproses data gambar, serta loop utama yang menjalankan kontroler.

1.Inisialisasi dan Normalisasi Gambar

```
mean = torch.Tensor([0.485, 0.456, 0.406])
std = torch.Tensor([0.229, 0.224, 0.225])
normalize = torchvision.transforms.Normalize(mean, std)
```

Nilai `mean` dan `std` digunakan untuk menormalisasi gambar yang diambil dari kamera JetBot. Normalisasi ini penting agar data masukan memiliki rentang nilai yang konsisten, yang membantu meningkatkan akurasi dan stabilitas prediksi dari model deep learning.

2. Fungsi preprocessCameraImage

```
def preprocessCameraImage(camera):
    global device, normalize
    data = camera.getImage()
    image = PIL.Image.frombytes('RGBA', (camera.getWidth(), camera.getHeight()), data, 'raw', 'BGRA').convert('RGB')
    image = transforms.functional.to_tensor(image).to(device)
    image.sub_(mean[:, None, None]).div_(std[:, None, None])
```

```
    return image[None, ...]
```

Fungsi ini menangani proses pengambilan gambar dari kamera JetBot. Gambar diambil dalam format `BGRA`, lalu dikonversi menjadi `RGB` dan diubah menjadi tensor agar kompatibel dengan model. Selanjutnya, normalisasi diterapkan pada gambar sebelum dimasukkan ke model untuk inferensi.

3. Memuat Model yang Telah Dilatih

```
model = torchvision.models.resnet18(weights=None)
model.fc = torch.nn.Linear(512, 2)
model.load_state_dict(torch.load(model_path))
```

Model ResNet18 digunakan sebagai arsitektur dasar, tetapi layer akhir (`fc`) dimodifikasi menjadi torch.nn.Linear(512, 2)`, yang memungkinkan dua kelas output, yaitu "terhalang" atau "tidak terhalang". Model dilatih untuk mendeteksi kemungkinan adanya hambatan, dan hasil pelatihan ini disimpan dalam file model yang kemudian dimuat pada saat inisialisasi.

4. Loop Utama (Main Loop)

```
while robot.step(timestep) != -1:
    camera_value = preprocessCameraImage(robot.camera)
    y = model(camera_value)
    y = F.softmax(y, dim=1)
    prob_blocked = float(y.flatten()[0])
    if prob_blocked < 0.5:
      if direction != 'forward':
        robot.forward(0.5)
        direction = 'forward'
    elif direction != 'left':
      direction = 'left'
      robot.left(0.4)
    pass
```

Pada loop utama ini:

- Gambar diambil dari kamera, diproses, dan dimasukkan ke dalam model.

- Model menghasilkan dua probabilitas (untuk "terhalang" dan "tidak terhalang") yang dinormalisasi menggunakan `softmax` untuk mendapatkan distribusi probabilitas.

- Berdasarkan probabilitas `prob_blocked` (kemungkinan jalur terhalang), JetBot memutuskan tindakannya:

  - Jika `prob_blocked` kurang dari 0.5, JetBot bergerak maju.

  - Jika `prob_blocked` lebih dari atau sama dengan 0.5, JetBot akan berbelok ke kiri untuk menghindari tabrakan.

Analisis Penerapan Proyek dalam Lingkungan Webots

Webots adalah simulator robotik yang memungkinkan pengujian robot dalam berbagai lingkungan. Proyek ini menggunakan Webots untuk menguji perilaku JetBot dalam kondisi yang beragam dan realistis, yang memungkinkan pengembangan tanpa harus mengoperasikan robot secara fisik. Ini sangat bermanfaat karena:

- Keamanan Pengujian dilakukan dalam lingkungan virtual, menghindari risiko kerusakan pada perangkat keras.

- Efisiensi Waktu dan Biaya: Pengujian yang ekstensif dapat dilakukan di simulator sebelum implementasi di dunia nyata.

- Replikasi Kondisi Uji: Webots memungkinkan pengaturan lingkungan uji yang konsisten untuk mengevaluasi kinerja model secara akurat.

Kesimpulan

Proyek "NVIDIA JetBot Collision Avoidance" pada Webots ini adalah aplikasi robotika yang kuat dan efisien, memanfaatkan kombinasi antara pembelajaran mesin, pemrosesan gambar, dan simulasi robotik. Dengan menggunakan model ResNet18 untuk mendeteksi hambatan, proyek ini menunjukkan bagaimana robot dapat dilatih untuk beroperasi secara otonom dalam lingkungan yang dinamis. Webots memungkinkan pengujian proyek dalam lingkungan simulasi, yang menjadikannya alat yang berguna bagi pengembang robotik untuk menguji dan mengoptimalkan sistem tanpa risiko langsung pada perangkat keras.