Pasinformatik, Volume 3, Issue 2, July 2024
E-ISSN : 2986-5360

Penerapan Sistem *Automatic Emergency Braking* Pada Agen *Autonomous Driving* Berbasis *Proximal Policy Optimization*

Fauzan Nursalma Mawardi [1] , Handoko Supeno [2]

Jurusan Teknik Informatika, Fakultas Teknik, Universitas Pasundan
Jln. Dr. Setiabudhi no. 193 Bandung, Jawa Barat
[1] contact.fauzannursalma@gmail.com , [2] handoko@unpas.ac.id

Abstrak : Dalam era industri saat ini, transportasi memegang peran yang sangat penting dalam kehidupan manusia dan mengalami perkembangan pesat seiring dengan kemajuan teknologi. Namun, masih ada sejumlah tantangan yang terkait dengan transportasi, seperti tingkat kemacetan dan kecelakaan yang tinggi yang seringkali terjadi disebabkan oleh faktor kelalaian manusia (human error). Penelitian ini bertujuan untuk menerapkan sistem Avoidance Collision, khususnya Automatic Emergency Braking (AEB) pada agen autonomous driving untuk membantu mengurangi dampak dari kelalaian manusia (human error) dengan menggunakan Deep Reinforcement Learning berbasis algoritma Proximal Policy Optimization (PPO), sehingga agen dapat mengambil tindakan secara optimal untuk mengurangi risiko kecelakaan atau menghindari tabrakan yang disebabkan oleh kelalaian manusia dalam lalu lintas. Model ini diujicobakan dan dijalankan pada dunia simulator CARLA.

Kata Kunci : *Avoidance Collision*, *Automatic Emergency Braking* (AEB), *Machine Learning*, *Deep Reinforcement Learning*, *Proximal Policy Optimization* (PPO)

I. PENDAHULUAN

Transportasi manusia telah mengalami perkembangan yang pesat seiring dengan kemajuan zaman. Namun, masih terdapat beberapa tantangan yang terkait dengan transportasi, salah satunya adalah kecelakaan yang disebabkan oleh faktor kelalaian manusia (human error) [1]. Kendaraan otonom dapat menjadi salah satu teknologi yang berpotensi untuk mengatasi permasalahan tersebut. Hal ini dapat terlihat dengan perkembangan yang pesat di dunia otomotif, dimana banyak perusahaan otomotif yang kini fokus pada pengembangan kendaraan otonom. Kendaraan otonom memanfaatkan pembelajaran mesin untuk memungkinkan kendaraan beroperasi secara mandiri tanpa intervensi dari pengemudi manusia. Berbagai penelitian telah dilakukan oleh para akademisi sebelumnya, dengan memanfaatkan berbagai teknik pembelajaran mesin dalam penerapan sistem kemudi otomatis pada kendaraan otonom. Deep Reinforcement Learning (DRL) merupakan salah satu jenis teknik pembelajaran mesin yang sangat cocok untuk kendaraan otonom atau Autonomous Driving. DRL memungkinkan kendaraan atau agen untuk belajar mengambil sebuah keputusan dalam lingkungan yang interaktif dan dinamis dengan metode trial and error menggunakan reward atau umpan balik dari tindakan dan pengalaman yang diperolehnya sendiri [2] [3]. Pada riset ini, fokus diberikan pada pengembangan sistem untuk mengurangi tingkat kecelakaan dalam berkendara yang dipicu oleh faktor human error dengan menerapkan Automatic Emergency Braking (AEB) pada agen Autonomous Driving. AEB merupakan sistem keamanan aktif pada kendaraan yang menggunakan sensor dan teknologi pengenalan untuk mendeteksi ancaman tabrakan dan secara otomatis mengambil tindakan pengereman darurat untuk mengurangi kecepatan atau mencegah tabrakan [4]. Dalam penerapan sistem pengereman otomatis pada agen Autonomous Driving, pemilihan algoritma yang tepat menjadi kunci penting. Salah satu algoritma yang dapat digunakan adalah Proximal Policy Optimization (PPO) [5] [6]. PPO merupakan algoritma reinforcement learning yang digunakan untuk mengoptimalkan kebijakan (policy) dalam pembelajaran mesin. PPO sangat cocok untuk autonomous driving karena dapat digunakan untuk melatih agen untuk membuat keputusan yang tepat dalam situasi yang kompleks dan bervariasi di jalan raya, sehingga meningkatkan keamanan dan efisiensi berkendara. Secara keseluruhan, hasil penelitian ini menunjukkan bahwa sistem pengereman otomatis yang dikendalikan oleh algoritma PPO dapat menjadi solusi yang efektif untuk mengurangi tingkat kecelakaan dalam berkendara yang dipicu oleh faktor human error. Sistem ini masih dalam tahap pengembangan, namun memiliki potensi untuk menjadi teknologi yang dapat meningkatkan keselamatan berkendara di masa depan [7].

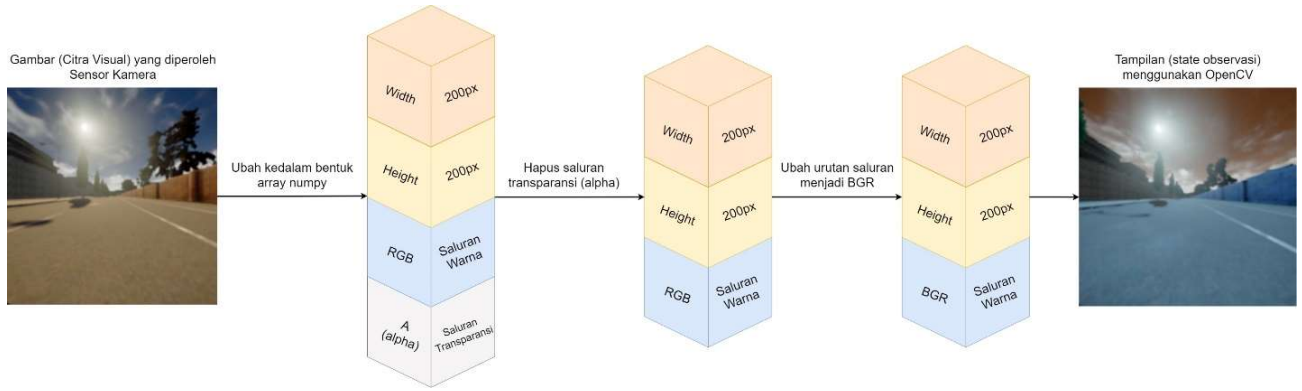
II. METODE PENELITIAN

Metode yang digunakan dalam penelitian ini adalah Deep Reinforcement Learning. Metode penelitiannya adalah sebagai berikut: (1) Perancangan state, (2) Perancangan aksi, (3) Perancangan reward function, (4) Perancangan model pembelajaran, (5) Perancangan algoritma, (6) Perancangan simulasi, (7) Pelatihan agent, (8) Evaluasi agent.

Hasil penelitian ini akan memberikan kontribusi berupa algoritma dan reward function yang dibutuhkan untuk membangun agen yang dapat melakukan *Automatic Emergency Braking* (AEB) hanya dengan input visual.

A. Perancangan State

Dalam penelitian ini, dirancang sebuah state yang terbentuk dari beberapa masukan (input) yang digunakan. Rancangan state ini berisi informasi mengenai kondisi agen dalam menentukan tindakan selanjutnya. State dalam penelitian ini berupa citra visual yang didapat menggunakan kamera sebagai masukan (input), kemudian di representasikan kedalam bentuk array numpy dengan atribut yang berupa frame yaitu width (lebar) dan height(tinggi) yang masing-masing memiliki nilai 200 pixel serta dimensi memiliki nilai 4 (RGBA). Kamera nantinya akan digunakan untuk mendeteksi halangan (obstacle) berupa mobil kendaraan NPC yang berhenti di tengah jalan. Hal ini bertujuan dalam melatih sistem pengereman darurat otomatis. Data visual yang diperoleh dari kamera akan digunakan oleh agen untuk memahami lingkungan sekitarnya dan mengambil keputusan yang tepat. Rancangan state menggunakan citra dengan dimensi 200x200 piksel yang dipilih dengan presisi untuk efisiensi komputasi, namun memiliki batasan dalam cakupan area pandangan sensor kamera. Untuk mengatasi keterbatasan ini, FOV (field of view) sensor kamera diatur menjadi 110 derajat dalam CARLA Simulator. Pengaturan ini meningkatkan kemampuan kendaraan otonom dalam mendeteksi objek dan situasi di sekitarnya, mengatasi kendala area pandang yang sempit. Tahap pre-processing dalam pengelolaan gambar melibatkan transformasi citra visual dari sensor kamera agen menjadi state yang digunakan dalam pelatihan. Citra visual awalnya diubah menjadi array numpy yang mencakup informasi tentang ukuran frame (200x200 piksel) dan 4 dimensi (R, G, B, A). Tujuan utamanya adalah untuk mempermudah pengolahan data. Tahap pre-processing ini dilakukan sesuai dengan tahapan yang terlihat dalam Gambar 1.



Gambar 1 Rancangan State Preprocessing

Tahap kedua pre-processing melibatkan penghapusan saluran transparansi (A) dari citra sensor kamera, sehingga hanya saluran warna (R, G, B) yang dipertahankan. Hal ini dilakukan karena saluran transparansi tidak relevan dan mengurangi dimensi citra, menghemat sumber daya komputasi, dan mempercepat proses preprocessing. Selanjutnya, urutan saluran warna diubah dari RGB menjadi BGR untuk konsistensi dengan konvensi yang digunakan oleh pustaka OpenCV, yang dapat meningkatkan kinerja dalam pemrosesan gambar dalam beberapa kasus. Sehingga untuk tahap rancangan state secara keseluruhan dapat dilihat pada gambar 1, dimana gambar (state observation) yang awalnya didapatkan oleh sensor kamera dengan pengaturan default dari CARLA Simulator akan di proses terlebih dahulu dengan mengatur pengaturan FOV (Field of View) dan tahap Pre-Processing sehingga menghasilkan gambar akhir (state observation) yang akan dijadikan sebagai input untuk agen dalam mengambil suatu tindakan dalam lingkungan.

B. Perancangan Aksi

Dalam penelitian ini, terdapat dua jenis aksi yang dirancang, yaitu pengereman (brake) dan percepatan (throttle). Nilai aksi pengereman diatur sebagai variabel kontinu dengan rentang antara 0.0 hingga 1.0, sedangkan aksi percepatan nilainya tetap pada 0.5. Nilai aksi brake dengan 0.0 menunjukkan bahwa kendaraan tidak melakukan pengereman, sementara nilai lebih dari 0.0 hingga 1.0 mengindikasikan intensitas pengereman yang meningkat dengan nilai yang lebih tinggi. Nilai aksi throttle tetap pada 0.5, menunjukkan percepatan sedang. Dalam pengambilan keputusan, agen akan memilih nilai aksi brake dengan throttle default berdasarkan situasi dan tujuan utama, yaitu menghindari tabrakan dengan halangan melalui pengereman darurat otomatis. Dengan menggunakan nilai aksi kontinu, agen dapat merespons situasi yang kompleks dan dinamis dengan baik, meningkatkan kemampuan kendaraan dalam menghindari tabrakan secara efektif.

C. Perancangan reward function

Reward function dalam penelitian ini dirancang untuk memberikan insentif kepada agen agar mencapai tujuan utama, yaitu menerapkan sistem pengereman darurat otomatis (AEB). Terdapat empat komponen utama dalam reward function:

a. Reward berdasarkan kecepatan: Memberikan reward positif (+1) jika agen mencapai kecepatan yang diinginkan (misalnya > 20 km/jam) untuk mendorong kendaraan agar bergerak.
b. Reward berdasarkan intensitas pengereman: Memberikan reward negatif berdasarkan intensitas pengereman yang diambil oleh agen, jika tidak terdeteksi halangan (obstacle). Tujuannya adalah memberikan sanksi berupa reward negatif untuk mencegah agen melakukan pengereman yang tidak perlu saat tidak ada halangan yang terdeteksi.
c. Reward berdasarkan deteksi halangan (obstacle): Memberikan reward negatif (-1) jika sensor obstacle mendeteksi halangan, tetapi kendaraan agen masih bergerak. Selain itu, memberikan reward positif berdasarkan intensitas pengereman yang dilakukan oleh agen. Tujuannya adalah mendorong agen untuk melakukan pengereman agar kendaraan dapat melambat dan berhenti untuk menghindari tabrakan dengan halangan.
d. Reward jika terjadi tabrakan (collision): Memberikan reward negatif yang sangat besar (-100) jika terjadi tabrakan. Hal ini bertujuan untuk mendorong agen untuk mencegah terjadinya tabrakan dan belajar menghindarinya.

Secara matematis, berikut merupakan persamaan reward function yang digunakan pada penelitian ini:

$$R = (1 * [kmh > 20]) - brake - (1 * [obstacle \ \& \ kmh > 0]) - (100 * collision)$$

Dimana:

R: Reward yang diberikan kepada agen berdasarkan action (tindakan) yang diambil.

1 * [kmh > 20]: Reward positif (+1) jika kecepatan kendaraan melebihi 20 km/jam.

-brake: Reward negatif berdasarkan brake intensity (0.0 sampai 1.0) jika tidak terdeteksi obstacle

-1 * [obstacle & kmh > 0]: Reward negatif (-1) jika terdeteksi adanya obstacle (rintangan) dan kecepatan kendaraan lebih besar dari 0.

-100 * [collision]: Reward negatif besar (-100) jika terjadi tabrakan (collision).

Keterangan variabel:

[kmh > 20]: Variabel bernilai 1 jika kecepatan kendaraan melebihi 20 km/jam.

[collision]: Variabel bernilai 1 jika terjadi tabrakan (collision).

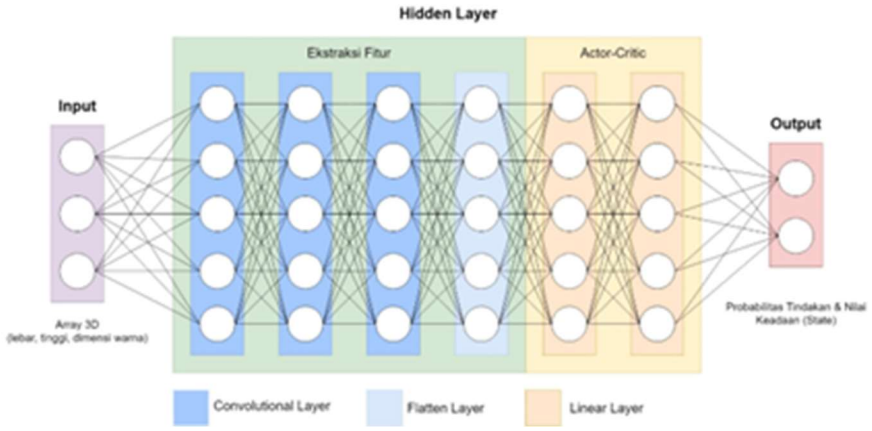
[obstacle]: Variabel bernilai 1 jika terdeteksi halangan (obstacle)

[obstacle & kmh > 0]: Variabel bernilai 1 jika terdeteksi adanya halangan (obstacle) dan kecepatan kendaraan lebih besar dari 0 kmh.

brake: Variabel dengan nilai berdasarkan intensitas pengereman (brake intensity) antara 0.0 sampai 1.0

D. Perancangan model pembelajaran

Dalam penelitian ini, pendekatan yang digunakan adalah pendekatan Actor-Critic yang telah di sediakan oleh CNN (Convolutional Neural Network) Policy pada PPO (Proximal Policy Optimization) Stable Baseline sebagai model arsitektur, sebagaimana yang terlihat pada Gambar 2 dibawah ini.



Gambar 2 Model Pembelajaran

Secara keseluruhan, rancangan arsitektur untuk CNN Policy Stable Baseline yang terdiri atas sebagai berikut:

a. Input, Keadaan lingkungan yang direpresentasikan sebagai gambar.
b. Layer, Terdiri dari 2 bagian utama yaitu:
c. Bagian ekstraksi fitur: Bagian ini terdiri dari convolutional layers (3 layer) dan flatten layer. Bagian ini digunakan untuk mengekstrak fitur dari keadaan lingkungan yang direpresentasikan sebagai gambar.
d. Bagian kebijakan (actor) dan nilai (critic): Bagian ini terdiri dari linear layer (2 layer), action net, dan value net. Bagian ini digunakan untuk menghasilkan probabilitas tindakan dan menilai seberapa baik kebijakan aktor.
e. Output, Probabilitas tindakan dan nilai keadaan.

Pendekatan Actor-Critic pada CNN Policy Stable Baselines memberikan keuntungan dari kedua pendekatan tersebut. Pendekatan Aktor memungkinkan agen untuk menghasilkan aksi berdasarkan kebijakan yang diperoleh dari pengalaman, sementara pendekatan Kritik memungkinkan agen untuk memodelkan dan memperbarui nilai kritik yang membantu dalam mengoptimalkan aksi yang dihasilkan [8].

E. Perancangan Algoritma

Berdasarkan hasil identifikasi permasalahan yang telah diuraikan pada bab-bab sebelumnya, penelitian ini bertujuan untuk mengimplementasikan sistem pengereman darurat secara otomatis atau Autonomic Emergency Braking (AEB) pada agen autonomous driving dengan menggunakan metode Deep Reinforcement Learning yang menggunakan algoritma Proximal Policy Optimization (PPO) [7]. Proses perancangan sistem ini melibatkan beberapa langkah, termasuk menyiapkan lingkungan simulasi dengan parameter seperti peta, model kendaraan, hambatan, sensor, serta definisi kondisi dan tindakan yang digunakan. Reward function digunakan sebagai penilaian kinerja sistem dan memberikan umpan balik kepada agen berdasarkan tindakannya. Pemilihan jaringan saraf dengan metode Actor-Critic dan penggunaan algoritma PPO dari Stable-Baselines menjadi pendekatan utama. Pengaturan hyperparameter seperti learning rate, gamma, betas, dan parameter lainnya juga penting dalam pelatihan agen. Dengan perancangan ini, penelitian ini bertujuan menciptakan sistem pengereman darurat otomatis yang efektif, meningkatkan keamanan dalam konteks autonomous driving. Pada tahap perancangan pembelajaran, langkah pertama adalah memilih algoritma yang digunakan untuk melatih agen agar dapat mengambil keputusan dan tindakan secara optimal. Dalam penelitian ini, algoritma yang digunakan adalah Proximal Policy Optimization yang telah dirancang oleh Stable-Baseline [8].

Algorithm 1 PPO, Actor-Critic Style

for iteration = 1, 2, . . . **do**

  for actor = 1, 2, . . ., N **do**

    Run policy $\pi_{\theta_{old}}$ in environment for T timesteps

    Compute advantage estimates $\hat{A}_1, . . ., \hat{A}_T$

  end for

  Optimize surrogate L wrt θ, with K epochs and minibatch size $M \leq NT$

  $\theta_{old} \leftarrow \theta$

end for

Dalam algoritma PPO yang diterapkan dalam pendekatan Actor-Critic yang dirancang oleh PPO Stable-baseline, kebijakan agen diperbarui menggunakan surrogate advantage objective dan value function diperbarui dengan fungsi kerugian value function, sehingga menggabungkan keuntungan dari kedua pendekatan tersebut. Harapannya, hal ini akan meningkatkan keselamatan dan mengurangi risiko kecelakaan dalam sistem pengereman darurat otomatis. PPO juga memanfaatkan beberapa hyperparameter, seperti learning rate, gamma, betas, dan parameter lainnya yang memengaruhi pembelajaran, sehingga pemilihan dan penyesuaian hyperparameter yang tepat menjadi kunci dalam mencapai performa optimal. Berikut ini adalah daftar hyperparameter yang akan digunakan dan disajikan dalam tabel 1.

Tabel 1 Hyperparameter PPO

No	Hyperparameter	Nilai
1	N_STEPS	2048
2	BATCH_SIZE / MINIBACH	64
3	N_EPOCHS	10
4	LEARNING_RATE (α)	0.001
5	GAMMA (γ)	0.99
6	CLIP_RANGE / EPSILON (ε)	0.2
7	GAE_LAMBDA	0.95
8	ENTROPY_COEF	0.01
9	VALUE_COEF	0.5
10	BETAS (β)	(0.9, 0.999)
11	MAX_GRAD_NORM	0.5
12	VERBOSE	1

F. Perancangan Simulasi

Rancangan lingkungan pada penelitian ini akan menggunakan jalan lurus dimana terdapat 2 objek kendaraan yaitu kendaraan agen (ego vehicle) dan kendaraan obstacle (NPC / Non-Player Character). Jarak antara agent dengan halangan (obstacle) yaitu 70 meter, dimana untuk pelatihannya agen di harapkan dapat melaju sejauh 65 meter dan mampu melakukan pengereman serta berhenti di area safe zone sebelum menabrak kendaraan NPC (obstacle). Penelitian ini menggunakan CARLA Simulator sebagai lingkungan pelatihan agen kendaraan otonom. CARLA Simulator dipilih karena kemampuannya mensimulasikan lingkungan nyata dengan efektif. Keputusan ini dipengaruhi oleh ketersediaan sumber daya dan fleksibilitas CARLA yang memungkinkan peneliti untuk menyesuaikan parameter simulasi, menciptakan berbagai skenario, dan mengamati perilaku agen di berbagai kondisi lalu lintas. CARLA Simulator menjadi pilihan yang cocok untuk pengujian dan pelatihan agen dalam lingkungan simulasi yang realistis dan aman[9]. Dalam perancangan lingkungan penelitian ini, Town03 dalam CARLA Simulator digunakan sebagai peta simulasi. Model kendaraan Tesla Model 3 menjadi kendaraan utama (Ego) yang akan digunakan oleh agen, sedangkan Jeep Rubicon ditempatkan diam di tengah jalan sebagai kendaraan NPC (Non-Player Character) yang berfungsi sebagai rintangan. Sensor kamera digunakan sebagai sensor utama pada kendaraan agen untuk mengenali lingkungan dan mengatasi rintangan. Penempatan sensor kamera melibatkan penentuan ukuran frame citra visual, koordinat sumbu x untuk posisi horizontal, dan sumbu z untuk posisi vertikal. Dengan penempatan yang tepat, agen dapat memahami lingkungan sekitarnya. Selain sensor kamera, sensor tambahan seperti sensor collision dan sensor obstacle digunakan untuk mendeteksi

hambatan di depan kendaraan agen dan mengidentifikasi tabrakan dengan objek lain dalam lingkungan simulasi. Hal ini mendukung pelaksanaan pengereman darurat otomatis dalam penelitian ini.

G. Pelatihan Agen

Setelah melakukan berbagai perancangan pada penelitian ini, tahap selanjutnya adalah proses pelatihan agen. Proses ini melibatkan interaksi agen dengan lingkungan dalam iterasi yang berulang. Dengan menggunakan algoritma Proximal Policy Optimization (PPO), agen dapat memperbarui kebijakan untuk memilih tindakan terbaik dan menerima reward yang optimal. Dalam pelatihan agen, langkah-langkah utama termasuk mendefinisikan hyperparameter, mempersiapkan lingkungan CARLA, inisialisasi agen PPO, dan menentukan jumlah iterasi dan total timestep pada setiap pelatihannya. Selama pelatihan, agen melakukan ekplorasi, mengambil tindakan berdasarkan kondisi, dan menerima imbalan yang disimpan pada rollout buffer. Setelah sejumlah langkah (n_step) tercapai, proses eksplorasi dihentikan untuk memproses pembelajaran berdasarkan data pengamatan yang telah disimpan. Pada proses pembelajaran agen, dimulai dari mengakumulasi nilai value, probabilitas dan entropy dari rollout buffer untuk menghitung keuntungan (advantages) & Rasio Kebijakan. Hal tersebut nantinya digunakan untuk:

- Menghitung loss dari kebijakan (clipped surrogate clip), mengukur sejauh mana agen perlu memperbarui atau mengubah kebijakannya. Tujuannya untuk memaksimalkan keuntungan yang diharapkan dari tindakan yang diambil oleh agen, sembari menghindari perubahan yang terlalu drastis. Berikut merupakan perhitungannya:

$$loss_1^{policy} = advantages * ratio$$

$$loss_2^{policy} = advantages * clamp\ (ratio, 1 - clip\ range, 1 + clip\ range)$$

$$loss^{policy} = \min\left(loss_1^{policy}, loss_2^{policy}\right).mean()$$

Penjelasan: Rumus tersebut adalah inti dari algoritma Proximal Policy Optimization (PPO). Terdiri dari dua komponen utama kerugian kebijakan. Komponen pertama, policy_loss_1, mengevaluasi sejauh mana model kebijakan baru dapat mengungguli model lama berdasarkan keuntungan yang diharapkan. Komponen kedua, policy_loss_2, membatasi perubahan kebijakan dengan memotong perbandingan tindakan. Kerugian kebijakan akhir dihitung sebagai rata-rata nilai minimum dari kedua komponen ini, dengan tujuan untuk meningkatkan kinerja model kebijakan dalam pembelajaran penguatan.

- Menghitung loss dari value (value loss), untuk mengukur sejauh mana perkiraan nilai (value) yang diperkirakan oleh agen mendekati nilai yang sebenarnya. Ini membantu dalam mengoreksi perkiraan nilai agen agar lebih akurat. Berikut merupakan perhitungannya:

$$loss^{value} = MSE(value_{pred}, return)$$

Penjelasan: Rumus tersebut menggunakan perhitungan Mean Squared Error (MSE) untuk membandingkan nilai prediksi dengan nilai sebenarnya (return aktual). Tujuannya adalah mengurangi perbedaan antara perkiraan agen dan nilai sebenarnya. Dengan mengurangi kerugian nilai ini, agen dapat lebih baik dalam memprediksi nilai keuntungan suatu keadaan dalam lingkungan, yang membantu dalam pembelajaran tindakan yang lebih baik.

- Menghitung loss dari entropy (entropy loss), untuk mendorong eksplorasi lebih lanjut oleh agen. Tujuannya untuk memaksimalkan entropi kebijakan, yang berarti menghargai kebijakan yang lebih eksploratif. Berikut merupakan perhitungannya:

$$entropy = -\min\ (prob * \log\ \text{prob})$$

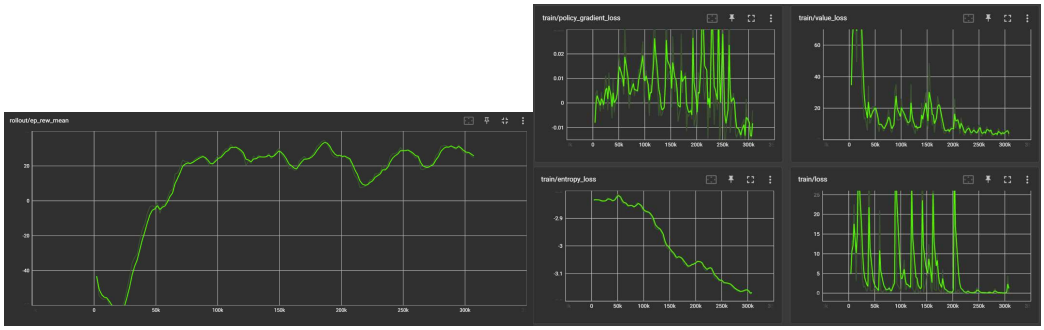
$$loss^{entropy} = -mean(entropy)$$

Penjelasan: Entropy dihitung dari distribusi probabilitas tindakan yang dihasilkan oleh agen kebijakan. Entropy diukur dengan rumus -min(prob * log_prob), di mana prob adalah probabilitas untuk setiap tindakan dan log_prob adalah logaritma alami dari probabilitas tersebut. Entropy loss dihitung sebagai rata-rata negatif dari entropy untuk mendorong agen kebijakan agar lebih cenderung menghasilkan tindakan dengan tingkat kepastian yang lebih rendah. Hal ini membuka peluang bagi agen untuk lebih banyak menjelajahi tindakan yang berbeda.

III. HASIL DAN PEMBAHASAN

Bagian ini akan memberikan gambaran mengenai hasil dari pelatihan agen yang dilakukan menggunakan Proximal Policy Optimization (PPO). Pelatihan agen telah berlangsung dengan menggunakan parameter-parameter yang telah ditentukan sebelumnya. Selama proses pelatihan, agen telah belajar dan mengoptimalkan kebijakan tindakannya untuk mencapai tujuan yang telah ditetapkan dalam lingkungan

tertentu. Selanjutnya, hasil pelatihan ini akan diuraikan dan dianalisis untuk mengevaluasi sejauh mana agen berhasil dalam mempelajari tugasnya dan tingkat kinerjanya dalam mengambil tindakan yang tepat.



Gambar 3 (Kiri) Grafik Reward Pelatihan Agen. (Kanan) Grafik Loss Pelatihan Agen

Dari hasil pelatihan yang telah dilakukan dengan melatih agen dalam 30 iterasi pelatihan, di mana setiap iterasi pelatihan terdiri dari 10.000 timestep (sehingga total mencapai 300.000 timestep), dapat dilihat pada Gambar 3 bahwa grafik tersebut yang menggambarkan perubahan reward yang di peroleh agen selama proses pelatihan. Reward tertinggi tercapai antara timestep ke-100.000 hingga ke-350.000, yang menandakan bahwa kebijakan agen telah mencapai konvergensi. Hal ini ditandai dengan adanya reward yang stabil pada tingkat tertinggi yang telah dicapai, sehingga mengindikasikan bahwa agen mampu secara efektif melakukan eksplorasi dan mengambil tindakan yang optimal dalam lingkungan yang diberikan. Grafik policy gradient, value loss, entropy loss dan loss mengindikasikan bahwa agen mengalami perbaikan kinerja secara bertahap ditandai dengan penurunan pada grafik, meskipun tren grafik menunjukkan adanya fluktuasi sementara selama proses pembelajaran pada setiap timestep. Penurunan pada policy gradient loss menunjukkan bahwa agen sedang meningkatkan kebijakan tindakan, sedangkan penurunan pada value loss menandakan kemajuan dalam estimasi nilai keadaan. Sementara penurunan pada entropy loss mengindikasikan bahwa agen sedang mengurangi tingkat ketidakpastian dalam tindakan atau prediksi dalam proses pembelajaran. Penurunan keseluruhan dalam loss menjadi indikasi positif bahwa agen secara keseluruhan sedang memperbaiki kinerjanya selama proses pelatihan dalam lingkungan CARLA Simulator.
Pengujian skenario pertama dilakukan dalam lingkungan dan kondisi yang sama seperti saat proses pelatihan. Tujuan pengujian skenario pertama adalah untuk memastikan bahwa agen dapat berhasil menerapkan sistem Automatic Emergency Braking (AEB). Keberhasilan diukur dengan kemampuan agen untuk berhenti di zona aman dan mencegah terjadinya tabrakan.

Tabel 2 Pengujian Skenario 1

Pengujian Skenario 1					
Nama Model	Berhenti		Menabrak (collision).	Kecepatan	Rata-rata Reward
	Didalam Safe zone	Diluar safe zone			
Model A (190.000)	10 episode	0 episode	0 episode	18 kmh	51.06
Model B (290.000)	10 episode	0 episode	0 episode	18 kmh	49.49

Hasil pengujian skenario pertama yang ditunjukan pada tabel 2 menunjukkan bahwa kedua model yang diuji, terutama model A (190.000) dan model B (290.000), mampu melakukan pengereman darurat dan berhasil menghindari tabrakan selama 10 episode pengujian, serta mampu berhenti di zona aman. Meskipun model B memiliki reward rata-rata yang sedikit lebih rendah daripada model A, ini mungkin menunjukkan bahwa agen melakukan pengereman yang lebih sedikit selama pengujian karena ketiadaan halangan yang terdeteksi, sesuai dengan reward function yang telah dijelaskan sebelumnya. Dalam pengujian ini, agen telah merespons secara tepat dalam situasi darurat dan melakukan tindakan yang diperlukan, seperti pengereman, untuk menghindari tabrakan. Hasil ini sejalan dengan pengujian sistem Automatic Emergency Braking (AEB) yang dilakukan oleh Euro NCAP pada tahun 2014 untuk jenis pengujian Low Speed City System dengan kecepatan kendaraan antara 10 hingga 50 km/jam[10]. Pengujian skenario kedua dilakukan di lingkungan yang sama dengan skenario pertama, tetapi dengan menghapus kendaraan NPC (Non-Player Character) yang berperan sebagai halangan (obstacle). Tujuan pengujian ini adalah untuk memverifikasi apakah agen dapat menghindari melakukan pengereman ketika tidak ada halangan. Fokus pengujian ini adalah pada persentase penggunaan brake (pengereman) oleh agen. Keberhasilan pengujian akan terlihat jika agen tidak berhenti di lokasi di mana halangan ditempatkan selama pelatihan, dan tidak melakukan pengereman yang tidak perlu karena ketiadaan halangan.

Tabel 3 Pengujian Skenario 2

Pengujian Skenario 2					
Nama Model	Berhasil melewati area *obstacle*	Berhenti di area *obstacle*	Kecepatan	Persentase *Brake*	Rata-rata Reward
Model A (190.000)	10 episode	0 episode	18 kmh	9.14%	-7.62
Model B (290.000)	10 episode	0 episode	18 kmh	10.41%	-8.43

Dari hasil pelatihan dan pengujian yang telah dilakukan menunjukan bahwasannya model masih mampu untuk memenuhi tugas utamanya yaitu melakukan pengereman darurat otomatis sebelum menabrak halangan (obstacle) untuk jenis pengujian Low Speed City System (kecepatan 10-50 kmh) yang dilakukan oleh Euro NCAP pada tahun 2014. [10] Untuk perbandingan berdasarkan nilai reward yang di peroleh kedua model yang digunakan pada proses pengujian, model A (190.000) sedikit lebih baik di bandingkan dengan model B (290.000).

IV. KESIMPULAN

Berdasarkan hasil analisis penelitian dan pengamatan yang di peroleh sejauh ini, maka dapat di simpulkan sebagai berikut:

(1) Dengan menggunakan Proximal Policy Optimization (PPO) sangat memungkinkan untuk agen autonomous driving dapat menerapkan sistem Autonomic Emergency Braking (AEB).
(2) Pada proses pelatihan dan pembelajaran, agen dapat mengobservasi dan eksplorasi lingkungan dengan baik sehingga dapat mengambil tindakan yang optimal untuk memperoleh reward yang maksimal pada setiap episodenya.
(3) Pada proses pengujian, agen masih layak untuk menerapkan sistem automatic emergency braking dengan melakukan pengereman darurat sebelum menabrak halangan (obstacle) pada dunia simulasi CARLA Simulator.

Ucapan Terima Kasih
Peneliti mengucapkan banyak terimakasih kepada Fakultas Teknik dan Program Studi Teknik Informatika Universitas Pasundan, Ketua Program Studi, para dosen dan pihak lain yang telah mendukung berjalannya kegiatan penelitian ini.

Referensi

[1] I. Barabás, A. Todoruţ, N. Cordoş, and A. Molea, “Current challenges in autonomous driving,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 252, p. 012096, Oct. 2017, doi: 10.1088/1757-899X/252/1/012096.
[2] B. R. Kiran *et al.*, “Deep Reinforcement Learning for Autonomous Driving: A Survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4909–4926, Jun. 2022, doi: 10.1109/TITS.2021.3054625.
[3] M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner, Eds., *Autonomous Driving*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016. doi: 10.1007/978-3-662-48847-8.
[4] On-Road Automated Driving (ORAD) committee, *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. doi: 10.4271/J3016_202104.
[5] Y. Guan, Y. Ren, S. E. Li, Q. Sun, L. Luo, and K. Li, “Centralized Cooperation for Connected and Automated Vehicles at Intersections by Proximal Policy Optimization,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 12597–12608, Nov. 2020, doi: 10.1109/TVT.2020.3026111.
[6] D. Quang Tran and S.-H. Bae, “Proximal Policy Optimization Through a Deep Reinforcement Learning Framework for Multiple Autonomous Vehicles at a Non-Signalized Intersection,” *Appl. Sci.*, vol. 10, no. 16, p. 5722, Aug. 2020, doi: 10.3390/app10165722.
[7] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” Aug. 28, 2017, *arXiv*: arXiv:1707.06347. Accessed: Jul. 16, 2024. [Online]. Available: http://arxiv.org/abs/1707.06347
[8] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: reliable reinforcement learning implementations,” *J Mach Learn Res*, vol. 22, no. 1, Jan. 2021.
[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An Open Urban Driving Simulator,” Nov. 10, 2017, *arXiv*: arXiv:1711.03938. Accessed: Jul. 17, 2024. [Online]. Available: http://arxiv.org/abs/1711.03938
[10] B. Fildes *et al.*, “Effectiveness of low speed autonomous emergency braking in real-world rear-end crashes,” *Accid. Anal. Prev.*, vol. 81, pp. 24–29, Aug. 2015, doi: 10.1016/j.aap.2015.03.029.