GUNADARMA UNIVERSITY

PELATIHAN AGEN *DEEP REINFORCEMENT LEARNING* BERBASIS *CNN* MENGGUNAKAN *PROXIMAL POLICY OPTIMIZATION* (*PPO*) DAN *VECTORIZED ENVIRONMENT* PADA BERAGAM *ENVIRONMENT* DARI *OPENAI-GYM*



Nama : **Harits Raharjo Setiono**

NPM : **10122581**

Dosen Pembimbing : **Dr. Witta Listiya Ningrum, S.Kom., MMSI.**

UG | PENULISANILMIAH UNIVERSITASGUNADARMA

LATAR BELAKANG

- Perkembangan DRL Terus berkembang dengan pesat, DRL memungkinkan agen belajar dari trial-error tanpa aturan eksplisit.
- Pelatihan visual membutuhkan daya komputasi tinggi dan proses lambat. Optimalisasi dilakukan menggunakan vectorized environment dan preprocessing visual.

RUMUSAN MASALAH

- Bagaimana metode merancang sistem DRL berbasis CNN + PPO untuk vectorized environment?
- Sejauh mana vectorized environment mempercepat pelatihan serta hasil yang diperoleh

RUANG LINGKUP & TUJUAN

RUANG LINGKUP

- Penerapan algortima PPO & CNN dari Stable-Baselines3, serta Pelatihan paralel menggunakan vectorized environment.
- Fokus pada game visual 2D: Pong, Breakout, Space Invaders, Flappy Bird, Super Mario Bros.
- Tidak mencakup transfer learning, game 3D, atau hyperparameter tuning.

TUJUAN

- Merancang sistem pelatihan agen DRL berbasis visi dengan menggunakan PPO sebagai algoritma pengambilan aksi, Serta menggunakan pelatihan paralel melalui Vectorized Environment
- Menguji efektivitas agen di berbagai environment dengan mengevaluasi berdasarkan skor rata-rata, durasi, dan durasi performa.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan sistematis yang dirancang untuk pengembangan dan pelatihan agen deep reinforcement learning berbasis algoritma Convolutional Neural Network (CNN) dan Proximal Policy Optimization (PPO).



PERENCANAAN

Pada tahap ini, Dimulai dengan perencanaan sistem, yang mencakupi aspek aspek berikut:

Target environment:

- Space Invaders, Breakout, Pong, Flappy Bird, Super Mario Bros.

Metode pelatihan:

- PPO (Proximal Policy Optimization) + CNN (Convolutional Neural Network) + Vectorized Environment.

Tools:

- Python, Stable-Baselines3, Gymnasium, PyTorch.

PRE-PROCESSING

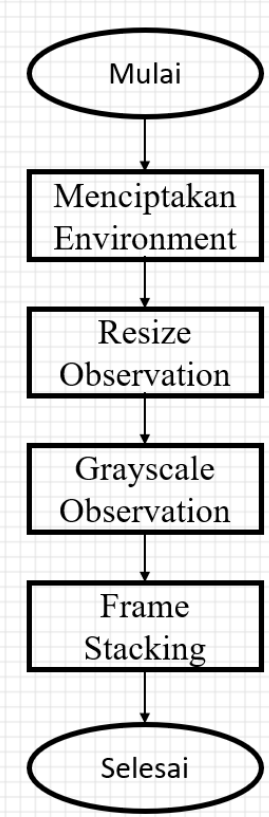
Tujuan Pre-processing

- Menyederhanakan input visual dari environment.
- Menyediakan konteks temporal melalui frame stacking.

Transformasi Input

- Resize layar: 84×84
- Grayscale conversion
- Frame stacking (4 frame terakhir)

```
env = ResizeObservation(env, (84, 84))
env = GrayscaleObservation(env, keep_dim=True)
env = FrameStack(env, num_stack=4)
```



Perancangan Model

Perencanaan Sistem

- PPO dengan CnnPolicy dari Stable-Baselines3.
- CNN terdiri dari 3 convolutional layers + 1 fully connected layer.

Inisialisasi PPO

- Inisialiasi PPO menggunakan syntax disamping

```
model = PPO(policy="CnnPolicy", env=train_env)
```

Parallel Environment Setup

- Menggunakan SubprocVecEnv sebanyak 8 environment.

```
train_env = SubprocVecEnv([make_train_env() for _ in range(8)])
```

Frame Stacking

- Menggabungkan beberapa frame terakhir menjadi satu observasi untuk informasi temporal.

```
train_env = VecFrameStack(train_env, n_stack=4)
```

EVALUASI

```
model = PPO(tensorboard_log=log_dir, verbose=1)
model.learn(total_timesteps=5_000_000, progress_bar=True)
```

Monitoring & Logging

Evaluasi dilakukan selama pelatihan menggunakan:

- TensorBoard untuk metrik reward & episode length.
- Verbose logging dan progress_bar=True untuk transparansi progres.

Evalcallback

EvalCallback, CheckpointCallback digunakan untuk:

- Menyimpan model otomatis.
- Mencegah kehilangan progres.
- Menganalisis performa di tiap tahap pelatihan.

PELATIHAN MODEL

Proses Pelatihan PPO

Langkah-langkah utama:

- Agen menerima input visual (grayscale, 84×84, 4 frame).
- PPO menghasilkan aksi → dieksekusi di environment.
- Reward dikumpulkan → dijadikan dasar update policy.

Optimasi Kebijakan

- PPO melakukan update setelah batch pengalaman terkumpul.

Monitoring Tambahan

- Visualisasi reward, loss, entropy via TensorBoard.
- Checkpoint memastikan hasil aman dan bisa dievaluasi ulang.

KESIMPULAN

Efisiensi Vectorized Environment

- Vectorized (8 env): 2h 15m
- Non-vectorized: 4h 52m
- Penghematan: 52.71% atau penghematan waktu sekitar 2 jam 36 menit

Performa PPO (Reward %)

Env	Reward	Length
Space Invaders	+38.67%	+72.66%
Breakout	-50.00%	-92.92%
Pong	+111.36%	+25.74%
Flappy Bird	+98.05%	+65.35%
Super Mario Bros	-78.17%	-96.34%

Kesimpulan

- PPO efektif di game visual sederhana.
- Kompleksitas environment sangat memengaruhi hasil.
- Vectorized env sangat direkomendasikan untuk efisiensi.