Reinforcement Learning in Games

(Computer Graphics Seminar Fall 2018)

BY: MAHIR GULZAR
Agenda

• Detour (ML in CG)
• RL emergence & approaches
• Unity ML-agents & Case studies
• Use cases (Papers)
• Conclusions
Machine Learning in Computer Graphics & Games

• Light simulation programs path tracing (De-noising)
• Audio-Driven Facial Animation
• Automatic pose estimation for animations

and many more...
De-noising

- SSPs (Sample per pixel)
- Depends on scene lighting
Audio driven facial animations

- Mapping of input audio waveforms to 3D vertex coordinates of a face model
Automatic real-time pose synthesis in animations

Unlike traditional systems, this approach decides automatically and in real-time how to combine tiny fragments into a sequence that matches the controller input, the environmental context, and any gameplay requests.
Back on track..
Reinforcement learning (RL) is the subfield of machine learning concerned with decision making and motor control. It studies how an agent can learn how to achieve goals in a complex, uncertain environment. In general encompassing all problems that involve making a sequence of decisions.

- Controlling a robot’s motors so that it’s able to run and jump,
- Playing video games and board games.
What is Reinforcement Learning or Deep RL?
Emergence in Games

Came to mainstream discussion after the release of this paper “Playing Atari with Deep Reinforcement Learning” by Deep Mind.
Ps: It wasn’t owned by Google at that time.

Psst!.. Not to forget AlphaGo
How about Dota 2?

- Human vs Bot (1v1)
- Humans vs Bots (5v5)
## Approaches

<table>
<thead>
<tr>
<th>Deepmind (Atari)</th>
<th>OpenAI (Dota Bot)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q-Learning</td>
<td>PPO (Proximal Policy Optimization)</td>
</tr>
<tr>
<td>CNN (Conv. Neural nets)</td>
<td>LSTM (Long short-term memory)</td>
</tr>
<tr>
<td>Vectorized Images</td>
<td>Dota Bot Api</td>
</tr>
</tbody>
</table>

Can you guess which one is more feasible in game development?
Not making any sense?

IT'S NOT CONFUSING

BUT I'M CONFUSED
Approaches

- Image vectorization

- Selective state space
  - A vector representing current position of the player
  - A quaternion or rotational vector of enemy, goal or player etc
A Starting point for game dev researchers

Machine Learning Agents
Unity ml-agents

Train intelligent agents with reinforcement learning and evolutionary methods via a simple Python API

- Academic researchers to study complex behaviors from visual content and realistic physics
- Industrial and enterprise researchers to implement large-scale parallel training regimes for robotics, autonomous vehicles, and other industrial applications
- Game developers to tackle challenges, such as using agents to dynamically adjust the game-difficulty level
Key Components

- Learning environment (Unity scenes and game characters)
  - Agents
  - Brains
  - Academy
- Python API (contains all the machine learning algorithms that are used for training)
- External Communicator
Block Diagram Multiple-Brains

Agent A1 → Brain A (Internal) → Academy
Agent A2 → Brain B (Heuristic) → Academy
Agent B1 → Brain C (External) → Academy
Agent C1 → Brain D (External) → Academy
Agent D1
Agent D2
Agent D3

External Communicator

Learning Environment

Python API
Training Types

- Built-in Training (Train a brain externally and apply it internally)
- Curriculum Training (extension to built-in training)
- Imitation Training (Assist agent by recording your behavior)
Some flexible training scenarios

- Single Agent: A single agent linked to a single brain
- Simultaneous Single-Agent: Multiple independent agents linked to a single brain
- Adversarial Self-Play: Two agents with inverse rewards linked to a single brain
- Cooperative Multi-Agent: Multiple interacting agents with a shared reward linked to either single or multiple different brains
More cool stuff

• Puppo The Corgi:

• Satellite AI
Use cases

In terms of developing game AI using RL, there are very few...

Can you guess why?

• Game Production Model:
  No need for super-human level intelligence to beat humans in games. Instead we make AI in such a way that human can play games with more engagement. (sense of achievement for users)

• Cost:
  Certainly there is a potential in adjusting agent behavior w.r.t to human (next slides) using RL but RL is costly. (DDA)

• Research Trends:
  More towards solving control problems (Robotics) or exploiting RL algorithms using games but not for using in games.
Use cases

M. Csikszentmihalyi flow model
Use cases

RL for Game Personalization:

- **Goal:** Increase user engagement in a game of Pong.
- **Proposition:** A composite reward function
Use cases

RL for Game Personalization:

• Step 1: GPP RL agent uses game score as its reward function
• Step 2: Use the policy of the GPP RL agent and fine-tunes it using the gamer engagement reward function

\[
R_t = \begin{cases} 
W_1 & \text{if RL agent loses the point} \\
W_2 & \text{for every rally count increment} \\
W_3 & \text{if rally ends}
\end{cases}
\]

where,

- \(R_t\): Reward value at time step \(t\)
- \(W_1, W_2, W_3\) are tunable scalar parameters

• \(w1 = -1\)
• \(w2 = 0.5\)
• \(w3 = -0.5\)
Use cases

RL for Game Personalization:
Use cases

DeepMimic: Physical based animations:

- You could say that it is about creating better cartoon animation without the help of a human animator.

- It could be a way of getting virtual actors to do things that real actors would find difficult or dangerous.
Use cases

DeepMimic: Physical based animations:

Instead of mimicking the whole animation, the reward function is written in such a way that it takes into consideration both mocap and goal task

\[ r_t = \omega^I r^I_t + \omega^G r^G_t. \]

where,

- \( r_t \) is reward at certain time step
- \( r^I \) is imitation reward
- \( r^G \) is goal task reward
References:

• Nvidia Optix AI- Accelerated Denoiser
  https://developer.nvidia.com/optix-denoiser

• Audio-Driven Facial Animation by Joint End-to-End Learning of Pose and Emotion

• Unity Kinematica and machine learning

• Unity Machine learning Agents
  https://github.com/Unity-Technologies/ml-agents/blob/0.5.0/docs/ML-Agents-Overview.md
References:


• Understanding LSTM Networks: http://colah.github.io/posts/2015-08-Understanding-LSTMs/#lstm-networks
References:

• Open Five (OpenAI bot for Dota-2):
  https://blog.openai.com/openai-five/
• Reinforcement Learning for Game Personalization on Edge Devices:
  https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8356853
• Flow in Games by Jenova Chen:
• TRPO in depth research paper review:
  https://www.youtube.com/watch?v=CKaN5PgkBc
• Path Tracing: A brief explanation
  https://chunky.llbit.se/path_tracing.html
That's all Folks!