SOSCON

Unity ML-Agents

Development of AI Agents Using Unity ML-Agents

Hanyang University | Automotive Engineering | Kyushik Min

20191017

SAMSUNG OPEN SOURCE CONFERENCE 2019

Contents

Self Introduction	**01**
Reinforcement Learning	**02**
Unity ML-Agents	**03**
Development Case using Unity ML-Agents	**04**

Self Introduction

Kyushik Min

Ph.D. Candidate of Automotive Department in Hanyang University

- Research Topics

  - Self Driving Car, Driver Assistance System, Artificial Intelligence, Reinforcement Learning

- Career

  - Unity Masters

  - Manager of Reinforcement Learning Korea (Facebook Page)

  - Creative Application Award Winner at ML-Agents Challenge

  - Doing Research and writing papers using ML-Agents

  - Conducting numerous seminars and lectures on ML-Agents

Unity ML-Agents

Reinforcement Learning

Reinforcement Learning

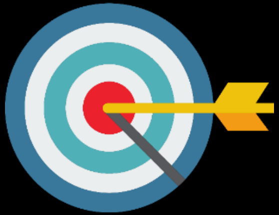
Machine Learning

- Machine Learning

  - Research areas that give computers the ability to learn without explicit programming

- Types of Machine Learning

  - Supervised Learning, Unsupervised Learning, Reinforcement Learning



Supervised Learning

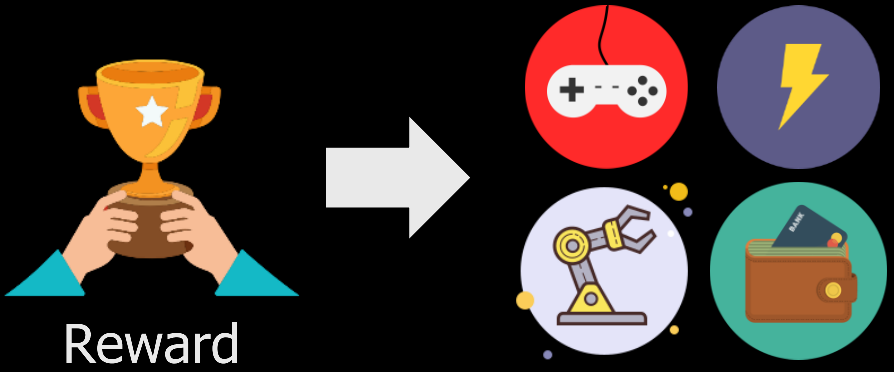
Unsupervised Learning

Reinforcement Learning

Reinforcement Learning

Reinforcement Learning

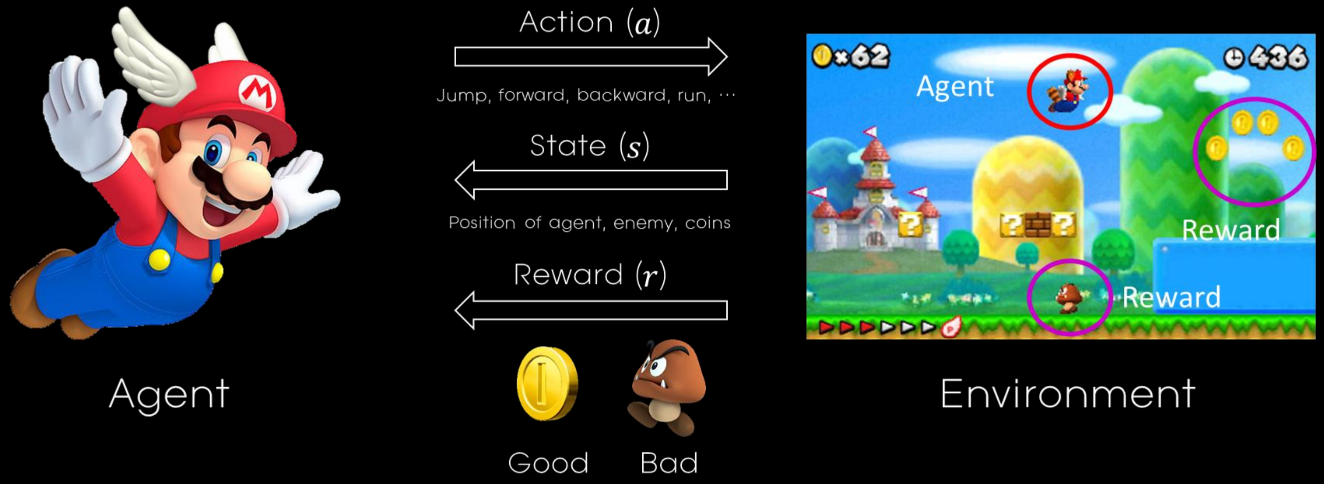
- Learning by Reward
- Perform various experiences through trial and error
- Learn to choose actions in a way that maximizes rewards



Reinforcement Learning

Reinforcement Learning

- Training Process of Reinforcement Learning



Action (a)

Jump, forward, backward, run, ⋯

State (s)

Position of agent, enemy, coins

Reward (r)

Good

Bad

Agent

Environment

Reinforcement Learning

AlphaGo (2016)

Reinforcement Learning

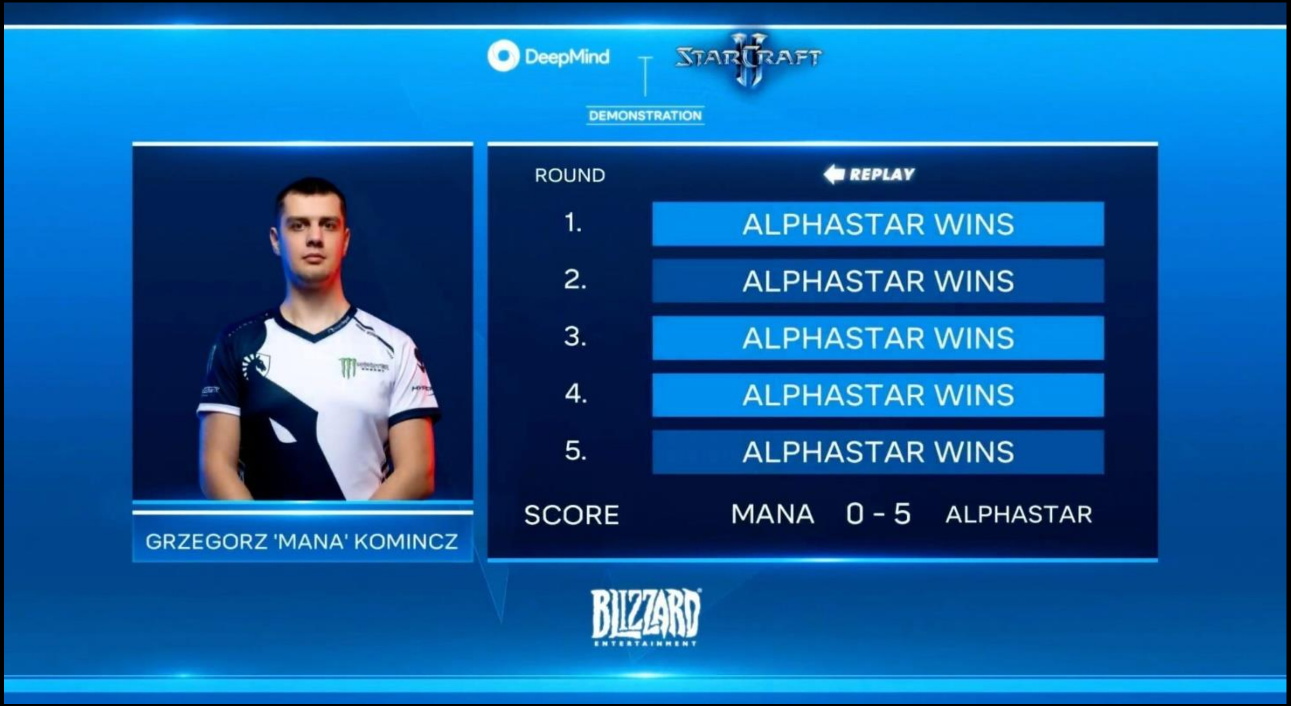
AlphaGo Zero (2017)

Reinforcement Learning

OpenAI Five (2018)

Reinforcement Learning

AlphaStar (2019)



ROUND	
1.	ALPHASTAR WINS
2.	ALPHASTAR WINS
3.	ALPHASTAR WINS
4.	ALPHASTAR WINS
5.	ALPHASTAR WINS
SCORE	MANA 0 - 5 ALPHASTAR

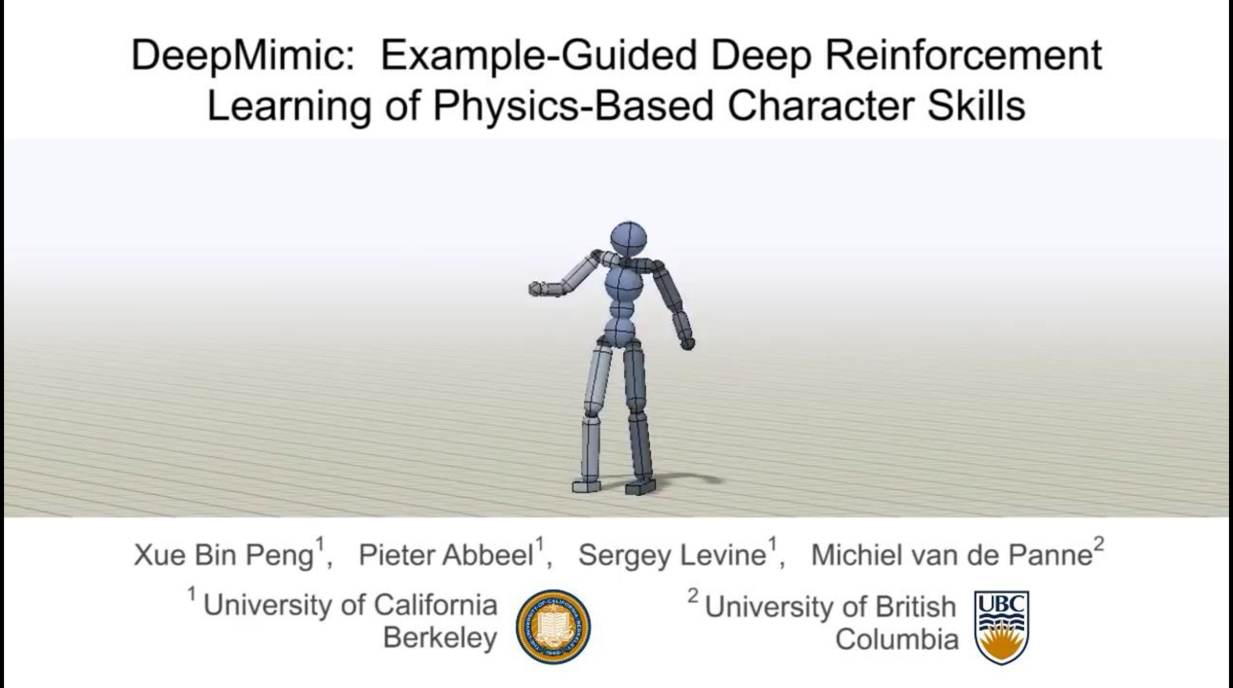
GRZEGORZ 'MANA' KOMINCZ

Reinforcement Learning

Locomotion

Reinforcement Learning

DeepMimic



DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills

Xue Bin Peng[1], Pieter Abbeel[1], Sergey Levine[1], Michiel van de Panne[2]

[1] University of California Berkeley

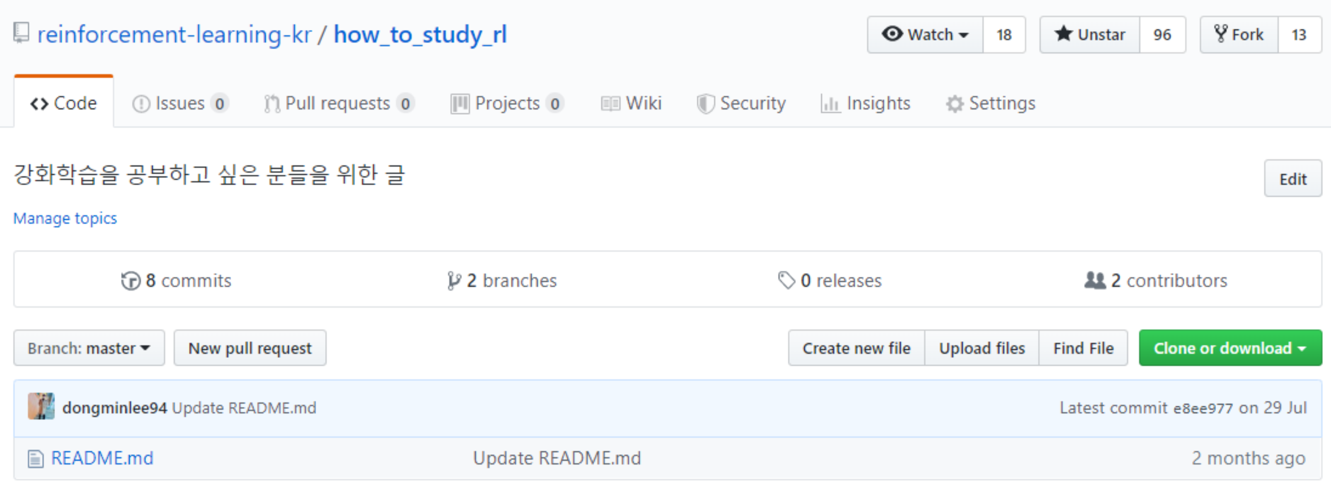
[2] University of British Columbia

Reinforcement Learning Korea

https://www.facebook.com/groups/ReinforcementLearningKR/

Reinforcement Learning

Reinforcement Learning Korea



reinforcement-learning-kr / how_to_study_rl

강화학습을 공부하고 싶은 분들을 위한 글

8 commits | 2 branches | 0 releases | 2 contributors

dongminlee94 Update README.md — Latest commit e8ee977 on 29 Jul

README.md — Update README.md — 2 months ago

https://github.com/reinforcement-learning-kr/how_to_study_rl

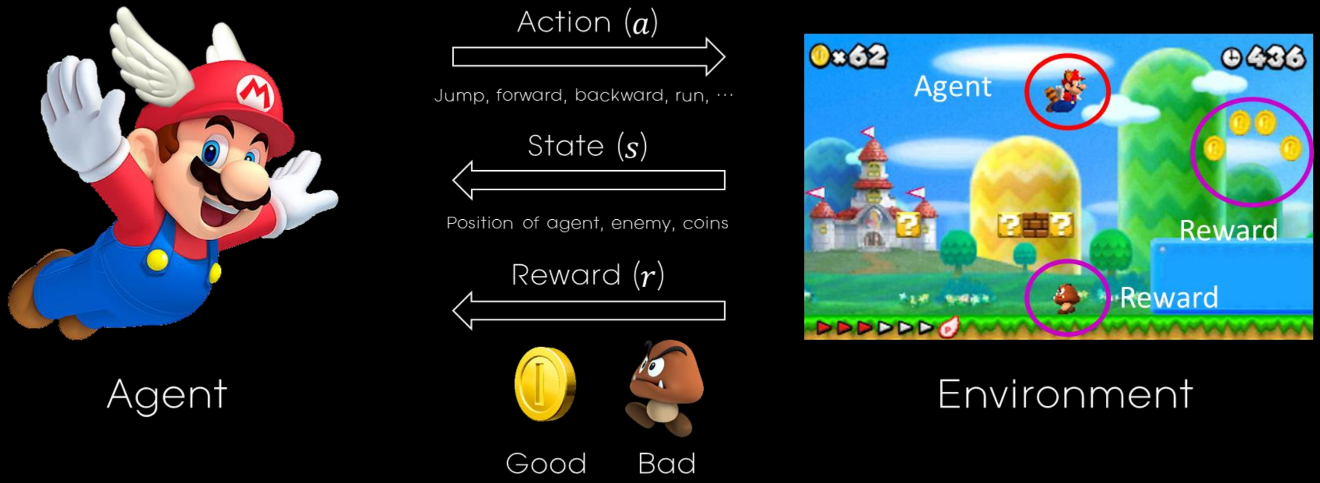
Unity ML-Agents

Unity ML-Agents

Unity ML-Agents

Reinforcement Learning

- Training Process of Reinforcement Learning



Action (a)

Jump, forward, backward, run, …

State (s)

Position of agent, enemy, coins

Reward (r)

Good

Bad

Agent

Environment

Unity ML-Agents

Reinforcement Learning

Agent

Deep Q Network
Rainbow DQN
Deep Deterministic Policy Gradient
Trust Region Policy Optimization
Proximal Policy Optimization

⋮

Environment

OpenAI GYM
Atari
Super Mario
Mujoco
Malmo

⋮

Unity ML-Agents

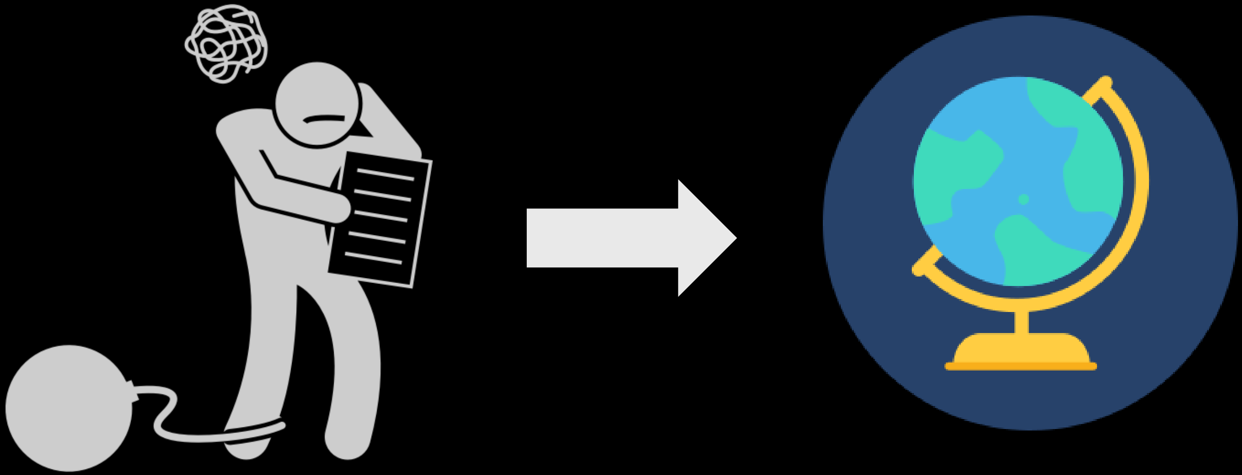
If you are using a created environment…

Difficulties modifying the environment

May not have the required environment

Unity ML-Agents

Need to create RL environment



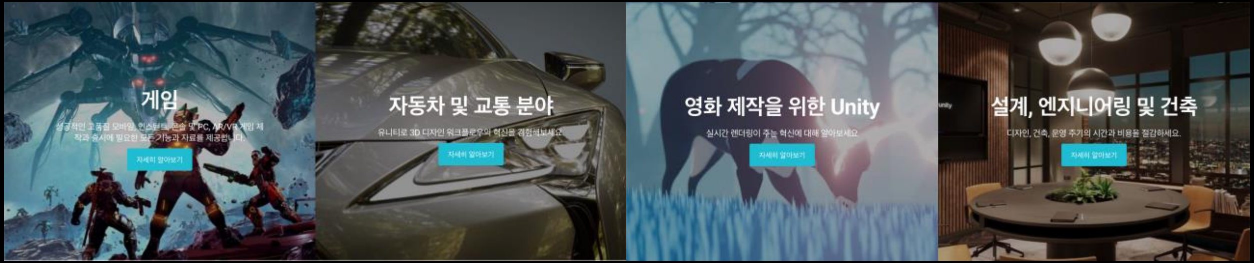
Concerns of people who study RL

Create an environment for testing RL

Unity ML-Agents

Unity

- Game engine that provides the development environment for 2D & 3D video games
- Also applied to various industries such as 3D animation, architectural visualization, VR
- Over 45% of the game engine market, over 5 million registered developers



게임

성공적인 고품질 모바일, 인스턴트, 콘솔 및 PC, AR/VR 게임 제작과 출시에 필요한 모든 기능과 자료를 제공합니다.

자동차 및 교통 분야

유니티로 3D 디자인 워크플로우의 혁신을 경험해보세요.

영화 제작을 위한 Unity

실시간 렌더링이 주는 혁신에 대해 알아보세요

설계, 엔지니어링 및 건축

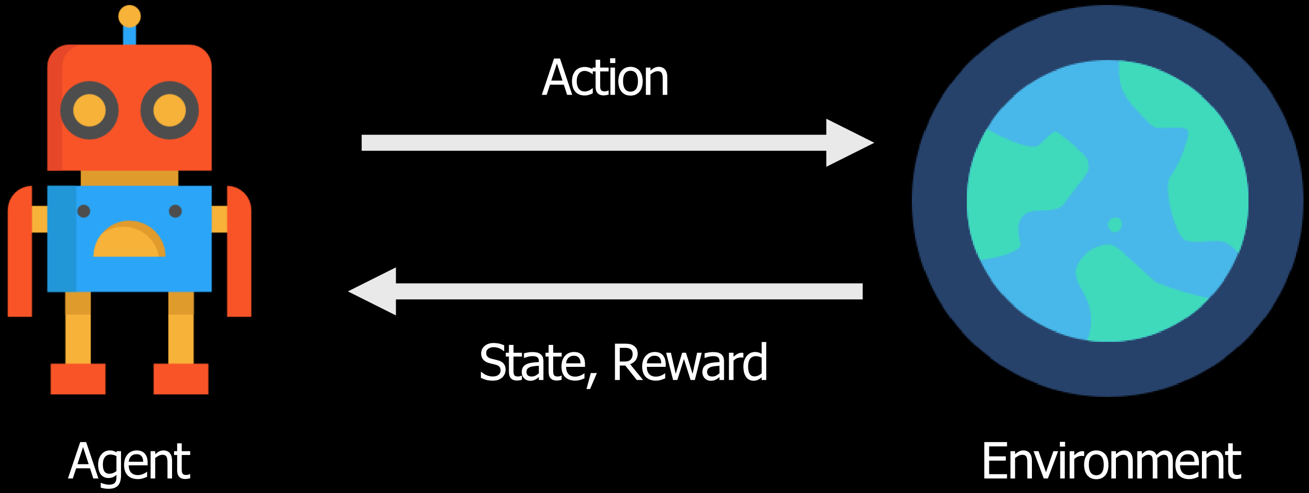
디자인, 건축, 운영 주기의 시간과 비용을 절감하세요.

Unity ML-Agents

Unity

Unity ML-Agents

Agent and Environment



Action

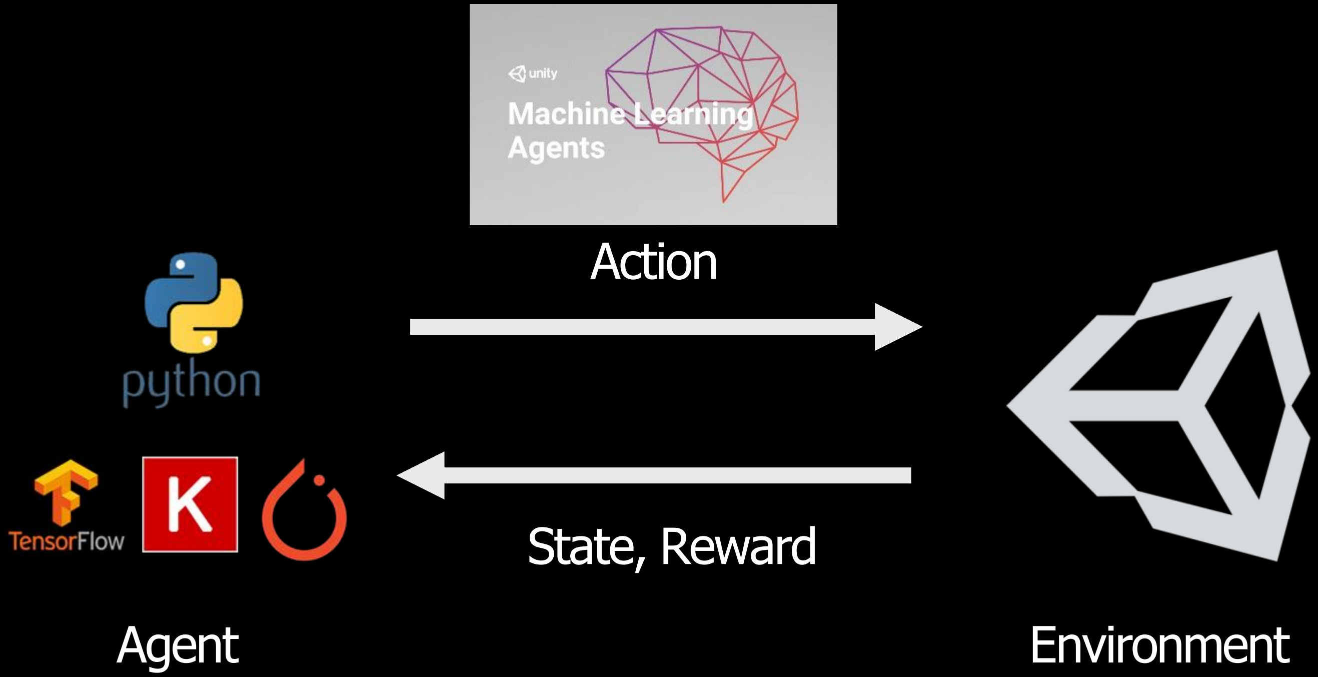
State, Reward

Agent

Environment

Unity ML-Agents

Agent and Environment



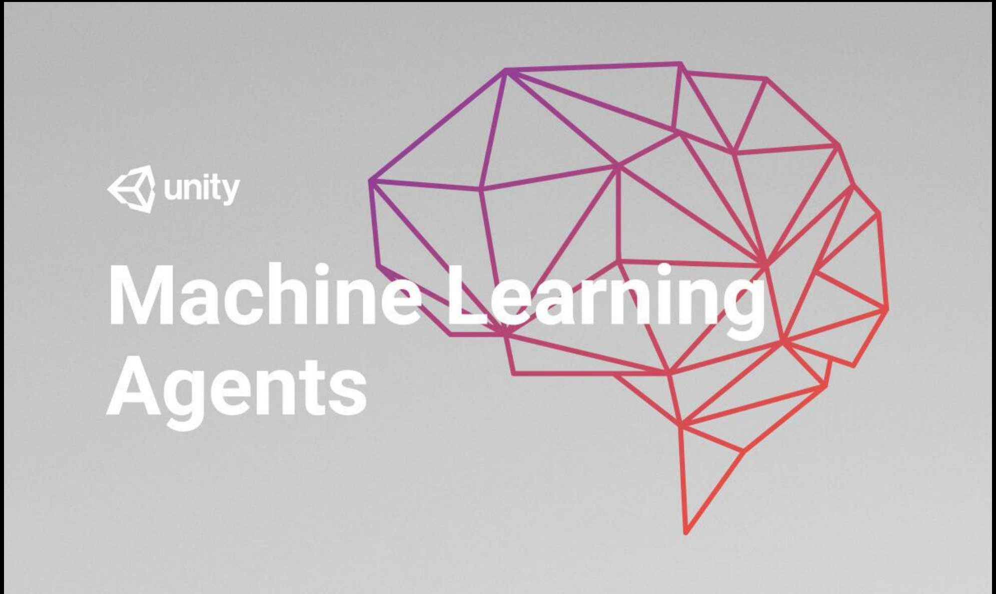
Action

State, Reward

Agent

Environment

Unity ML-Agents



Released 2017.09.19 -> Beta 0.10.0

Unity ML-Agents

Unity ML-Agents

- API to simplify configuration for RL in Unity environments
- Communication between Python and Unity environment (State, Action, Reward)
- Consists of Agent, Brain, Academy

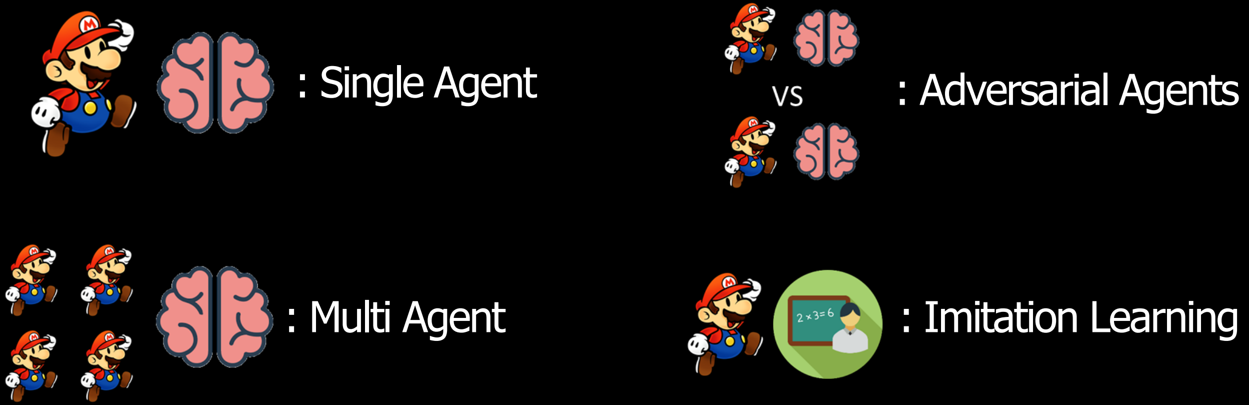
  - Agent: Code for the Agent, configuration for Obs, action, reward

  - Brain: Determine how to control agents (Player, Heuristic, Learning)

  - Academy: Integrated management of brain, various settings for environment

Unity ML-Agents

Unity ML-Agents



: Single Agent

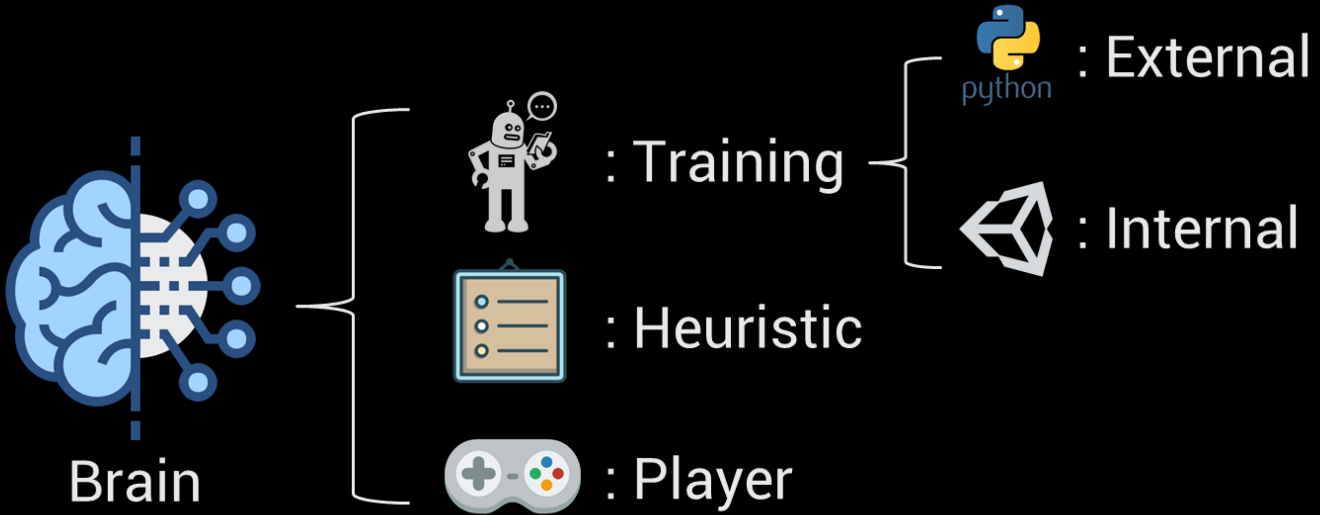
: Adversarial Agents

: Multi Agent

: Imitation Learning

Unity ML-Agents

Unity ML-Agents



Brain

: Training

: External

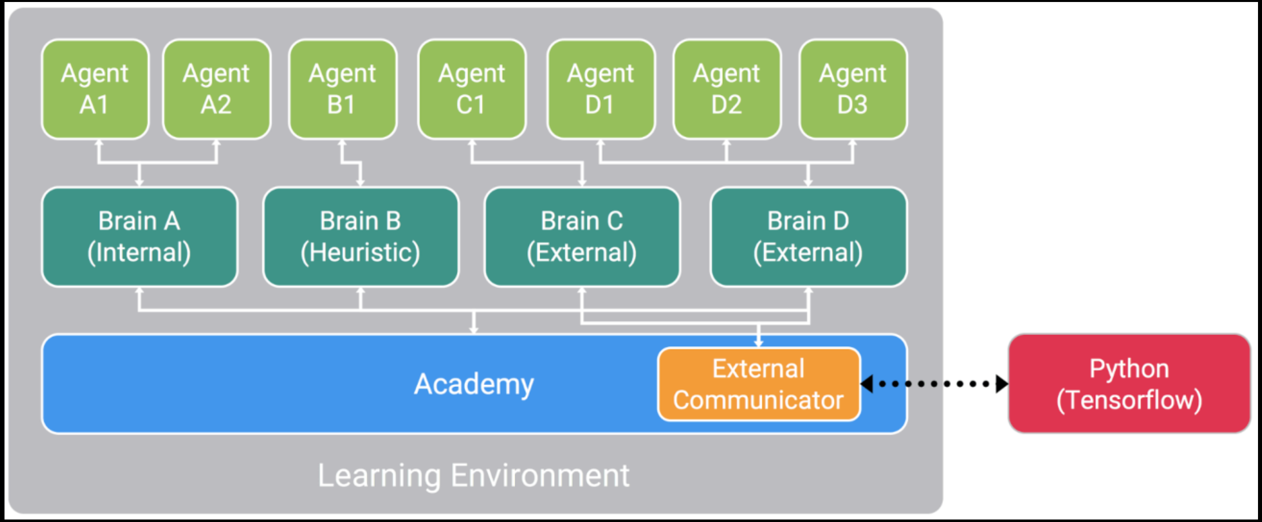
: Internal

: Heuristic

: Player

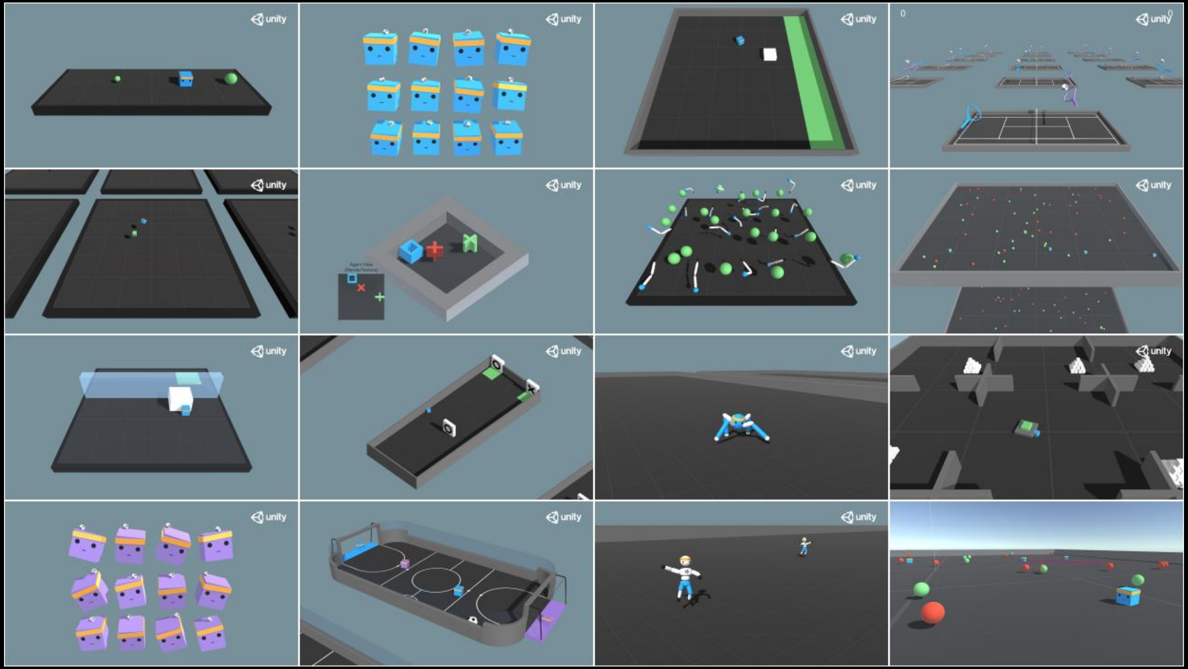
Unity ML-Agents

Unity ML-Agents



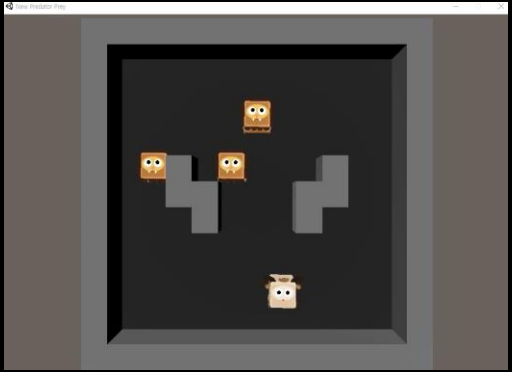
Unity ML-Agents

Unity ML-Agents



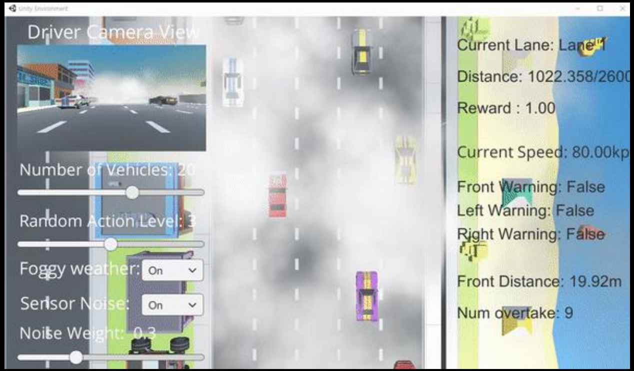
Unity ML-Agents

Unity ML-Agents



Distance: 42.39

Reward : 0.085



Number of Vehicles: 20

Random Action Level: 3

Foggy weather: On

Sensor Noise: On

Noise Weight: 0.3

Current Lane: Lane 1

Distance: 1022.358/2600

Reward : 1.00

Current Speed: 80.00kp

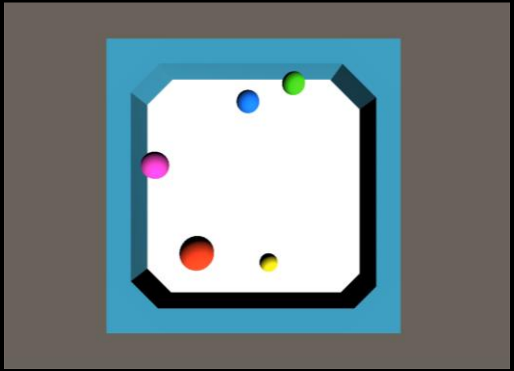
Front Warning: False

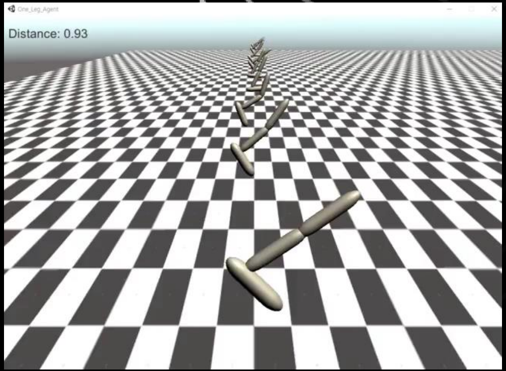
Left Warning: False

Right Warning: False

Front Distance: 19.92m

Num overtake: 9





Distance: 0.93

Unity ML-Agents

Development Case using Unity ML-Agents

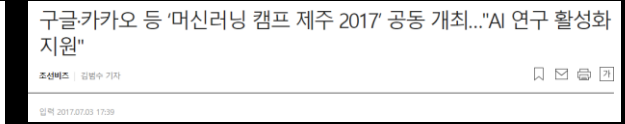
Machine Learning Camp Jeju 2017

Google 한국 블로그

Google에 대한 다양한 소식을 확인해 보세요!

구글이 머신러닝 캠프 제주 2017과 함께 합니다!

2017년 7월 3일 월요일



구글·카카오 등 '머신러닝 캠프 제주 2017' 공동 개최..."AI 연구 활성화 지원"

조선비즈 | 김범수 기자

가

입력 2017.07.03 17:39

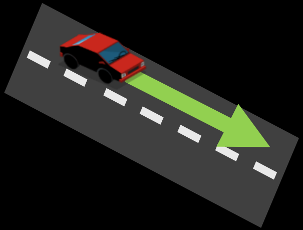
'머신러닝 캠프 제주 2017' 인공지능기술 대중화 캠프 개최

정동희 기자 | 승인 2017.07.03 15:01 | 댓글 0

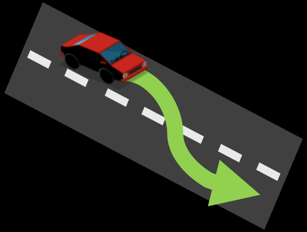
Development Case using Unity ML-Agents

Project Proposal

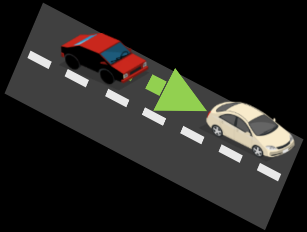
- Various Advanced Driver Assistance Systems (ADAS) are already commercialized, including lane keeping and lane changes.
- Autonomous driving is possible with a combination of ADAS
- The challenge is to determine which ADAS controls the vehicle in every state



Lane Keeping



Lane Change

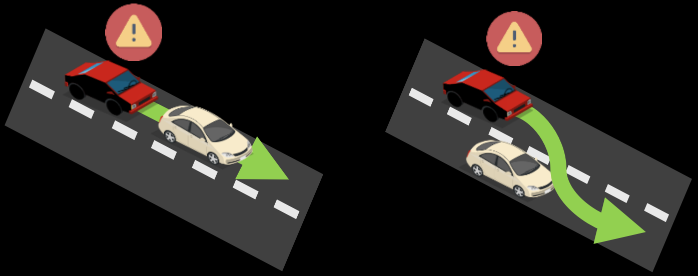


Cruise Control

Development Case using Unity ML-Agents

Project Proposal

- In case of RL, prediction of action selection is hard
- Applying collision avoidance systems such as AEB and lane change prevention
- Use sensors such as cameras, LIDAR, and RADAR that are used in autonomous vehicles
- But there is no simulator that satisfies the desired condition

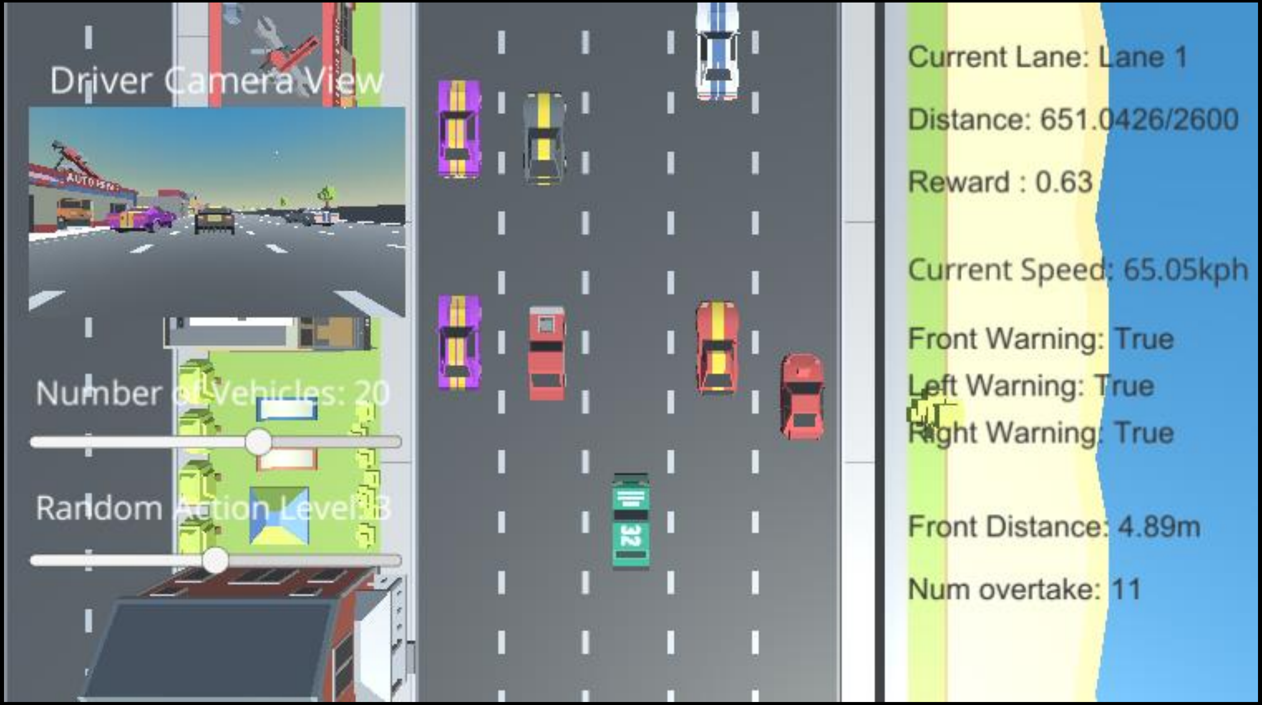


Collision Avoidance Systems

Camera, LIDAR, RADAR

Development Case using Unity ML-Agents

Simulator



Driver Camera View

Number of Vehicles: 20

Random Action Level: 3

Current Lane: Lane 1

Distance: 651.0426/2600

Reward : 0.63

Current Speed: 65.05kph

Front Warning: True

Left Warning: True

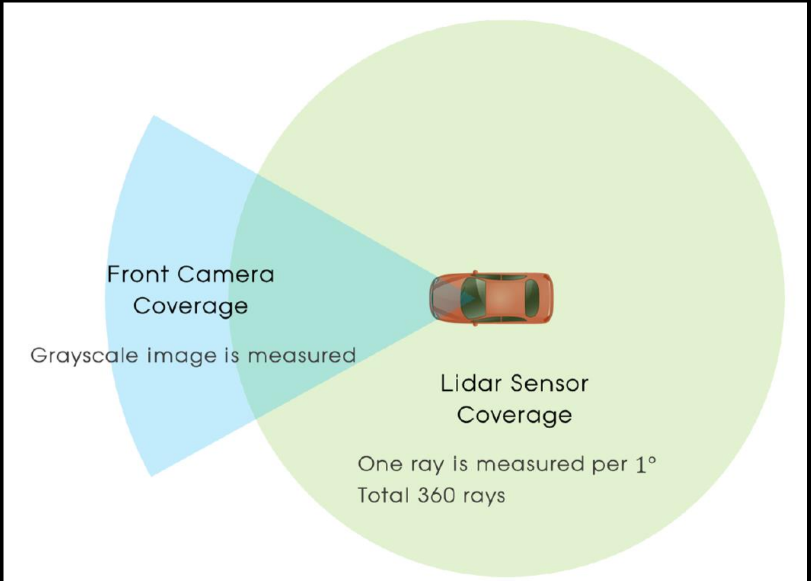
Right Warning: True

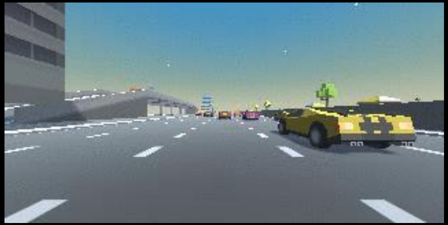
Front Distance: 4.89m

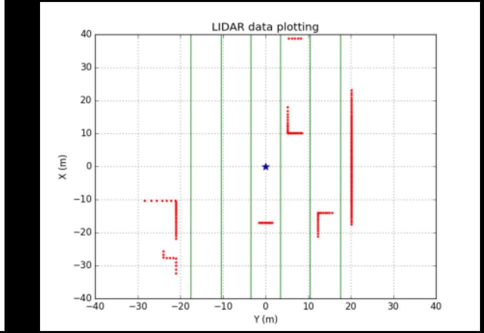
Num overtake: 11

Development Case using Unity ML-Agents

Simulator (Observations)

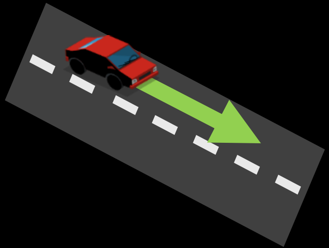




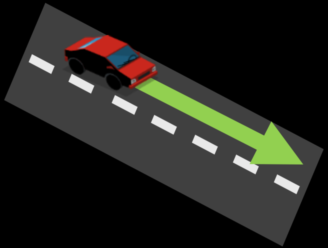


Development Case using Unity ML-Agents

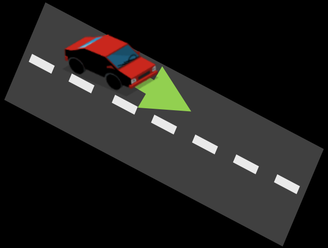
Simulator (Actions)



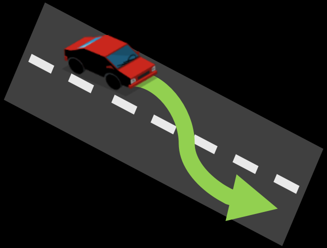
Keep Current State



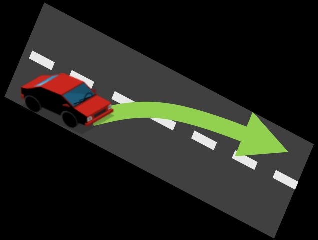
Acceleration



Deceleration



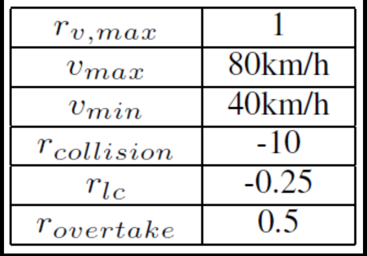
Lane Change (Right)



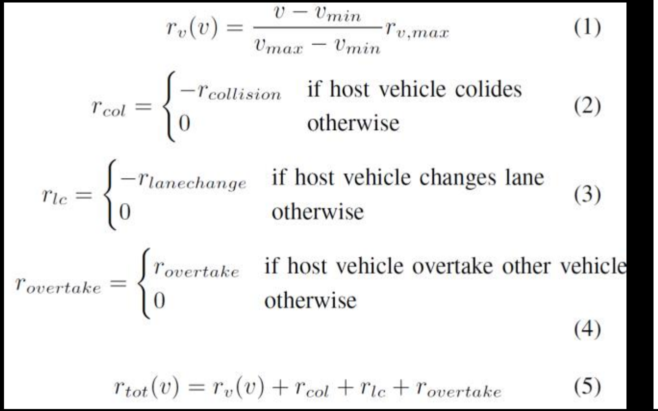
Lane Change (Left)

Development Case using Unity ML-Agents

Simulator (Reward)



$r_{v,max}$	1
v_{max}	80km/h
v_{min}	40km/h
$r_{collision}$	-10
r_{lc}	-0.25
$r_{overtake}$	0.5



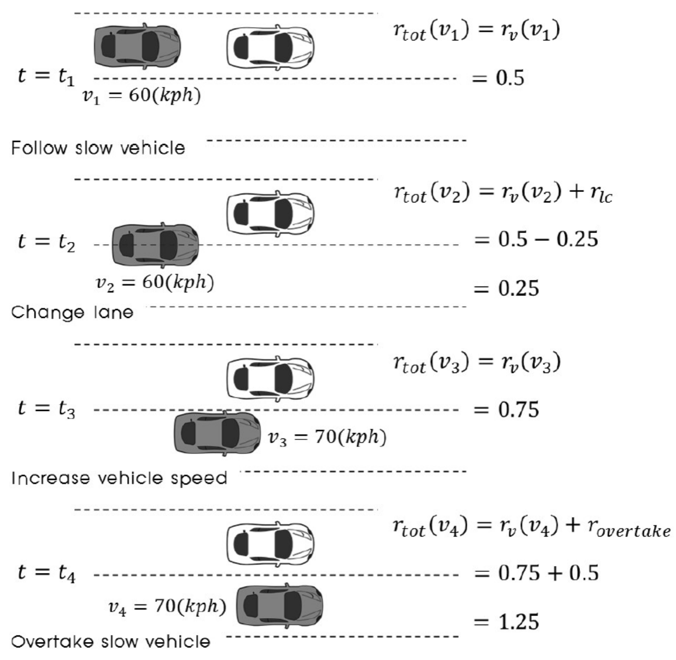
$$r_v(v) = \frac{v - v_{min}}{v_{max} - v_{min}} r_{v,max} \qquad (1)$$

$$r_{col} = \begin{cases} -r_{collision} & \text{if host vehicle colides} \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

$$r_{lc} = \begin{cases} -r_{lanechange} & \text{if host vehicle changes lane} \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

$$r_{overtake} = \begin{cases} r_{overtake} & \text{if host vehicle overtake other vehicle} \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

$$r_{tot}(v) = r_v(v) + r_{col} + r_{lc} + r_{overtake} \qquad (5)$$



$t = t_1$ $v_1 = 60(kph)$

$$r_{tot}(v_1) = r_v(v_1) \\ = 0.5$$

Follow slow vehicle

$t = t_2$ $v_2 = 60(kph)$

$$r_{tot}(v_2) = r_v(v_2) + r_{lc} \\ = 0.5 - 0.25 \\ = 0.25$$

Change lane

$t = t_3$ $v_3 = 70(kph)$

$$r_{tot}(v_3) = r_v(v_3) \\ = 0.75$$

Increase vehicle speed

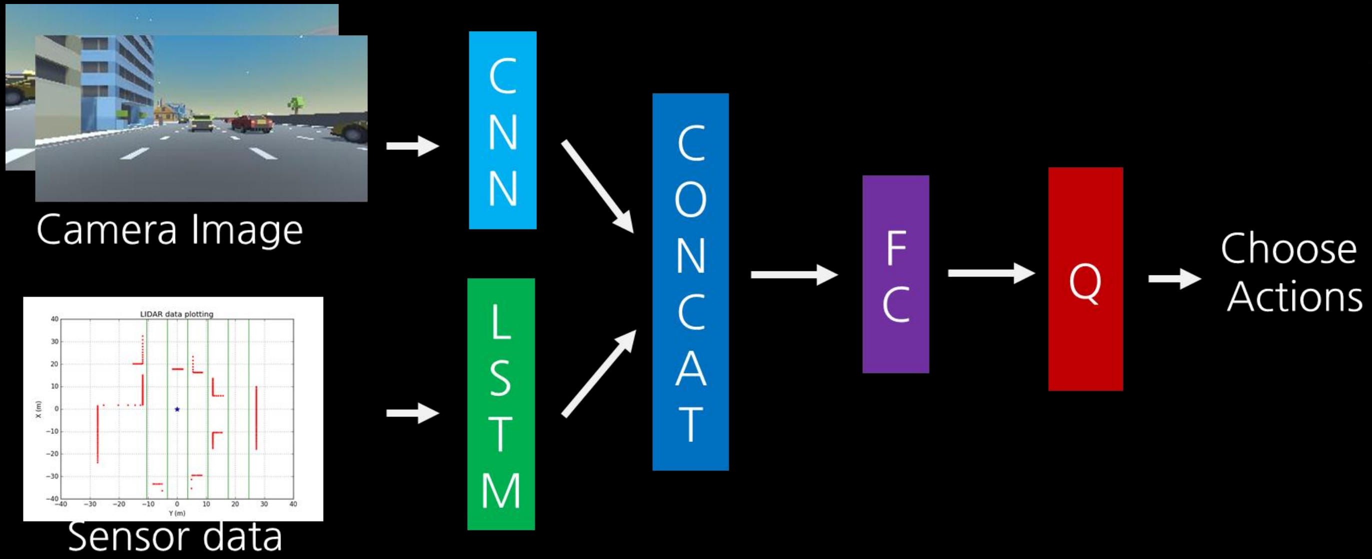
$t = t_4$ $v_4 = 70(kph)$

$$r_{tot}(v_4) = r_v(v_4) + r_{overtake} \\ = 0.75 + 0.5 \\ = 1.25$$

Overtake slow vehicle

Development Case using Unity ML-Agents

Network Architecture



Camera Image

Sensor data

CNN

LSTM

CONCAT

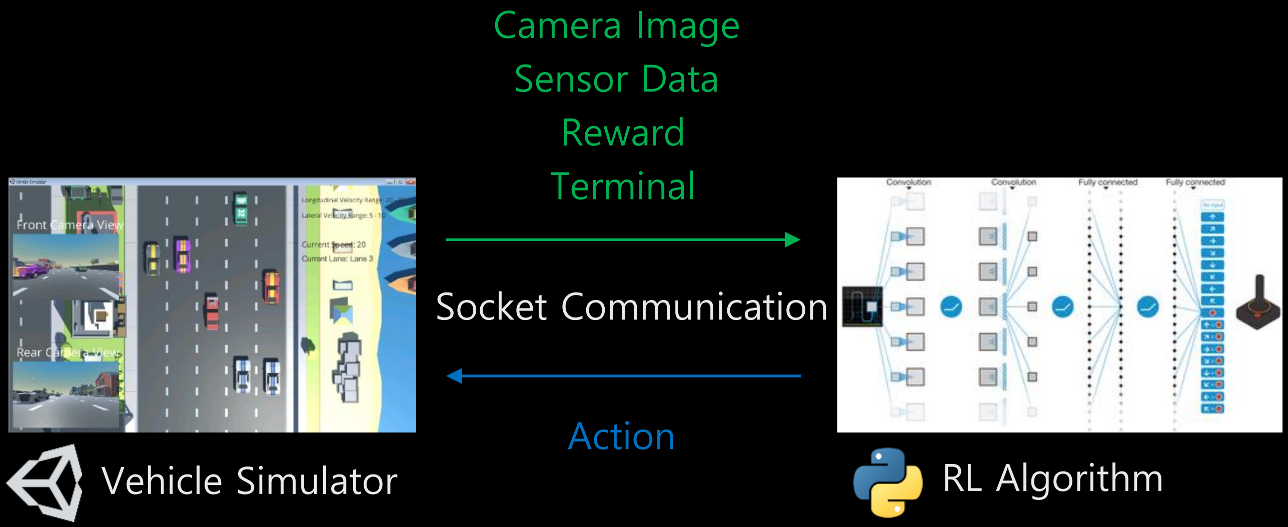
FC

Q

Choose Actions

Development Case using Unity ML-Agents

Communication between Python & Unity



Camera Image

Sensor Data

Reward

Terminal

Socket Communication

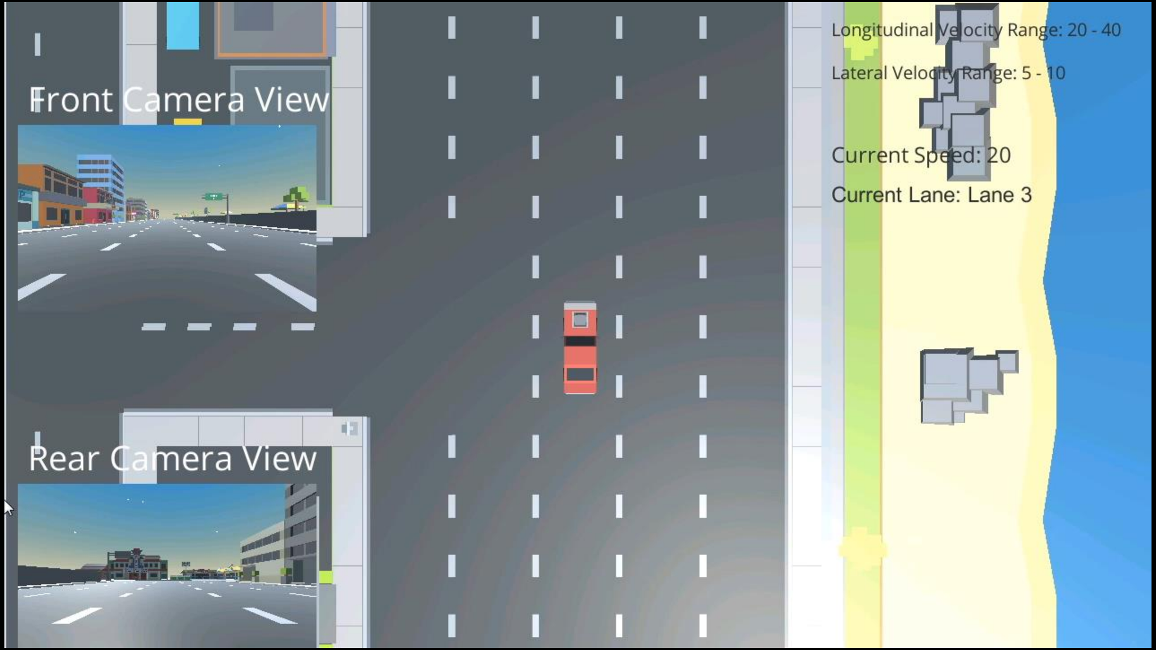
Action

Vehicle Simulator

RL Algorithm

Development Case using Unity ML-Agents

Result



Longitudinal Velocity Range: 20 - 40

Lateral Velocity Range: 5 - 10

Current Speed: 20

Current Lane: Lane 3

Development Case using Unity ML-Agents

Communication between Python & Unity

- Implemented using Socket communication, but there are many unstable parts and bugs

  - Problem with communication interruption

  - Lots of coding is required for small changes in the environment

  - Synchronization Problems Between Communications

  - Issue with speed differences between Unity and Python code

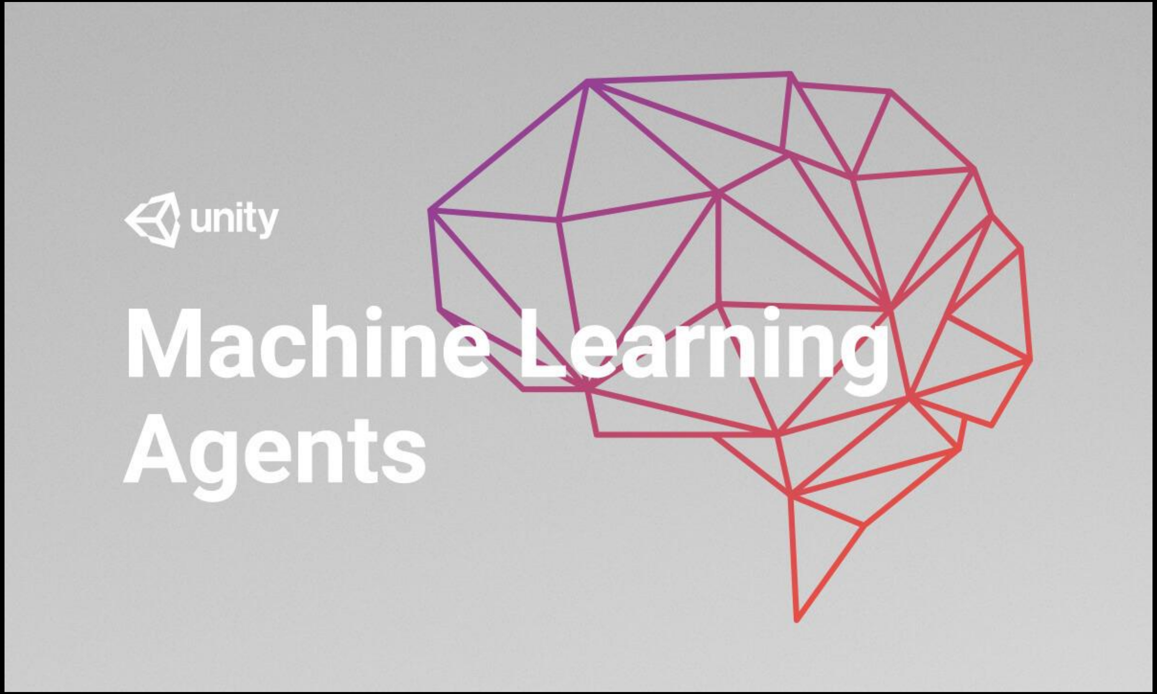
- Trying to solve problems for about 1~2 months

  - About 70% of all problems were solved

  - It was scheduled for release on Github

Development Case using Unity ML-Agents

ML-Agents released (2017.09.19)



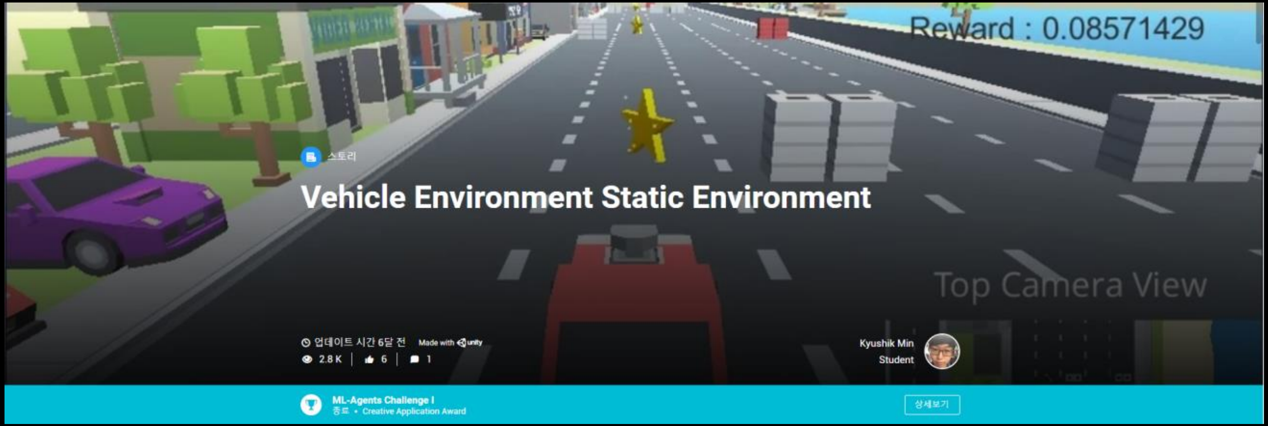
Development Case using Unity ML-Agents

ML-Agents Challenge

Development Case using Unity ML-Agents

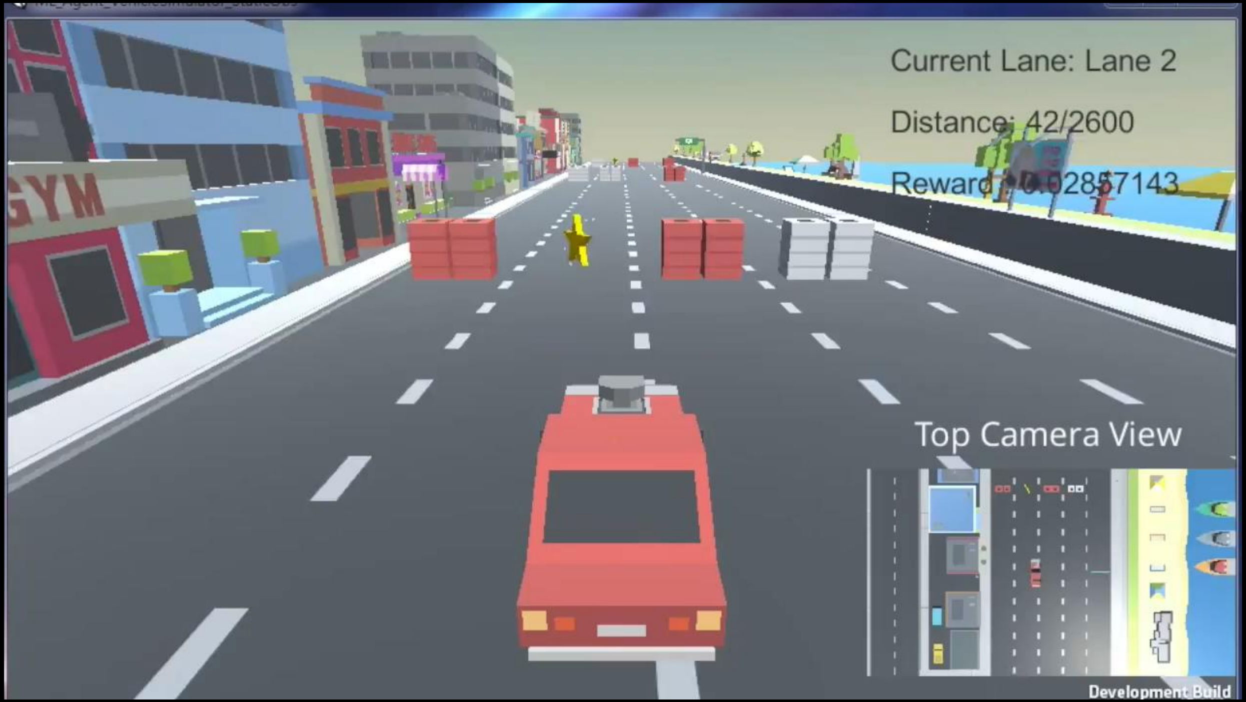
ML-Agents Challenge

- Apply ML-Agents to the environment created in Jeju Camp
- Made with a simpler environment (static obstacles)



Development Case using Unity ML-Agents

ML-Agents Challenge



Current Lane: Lane 2

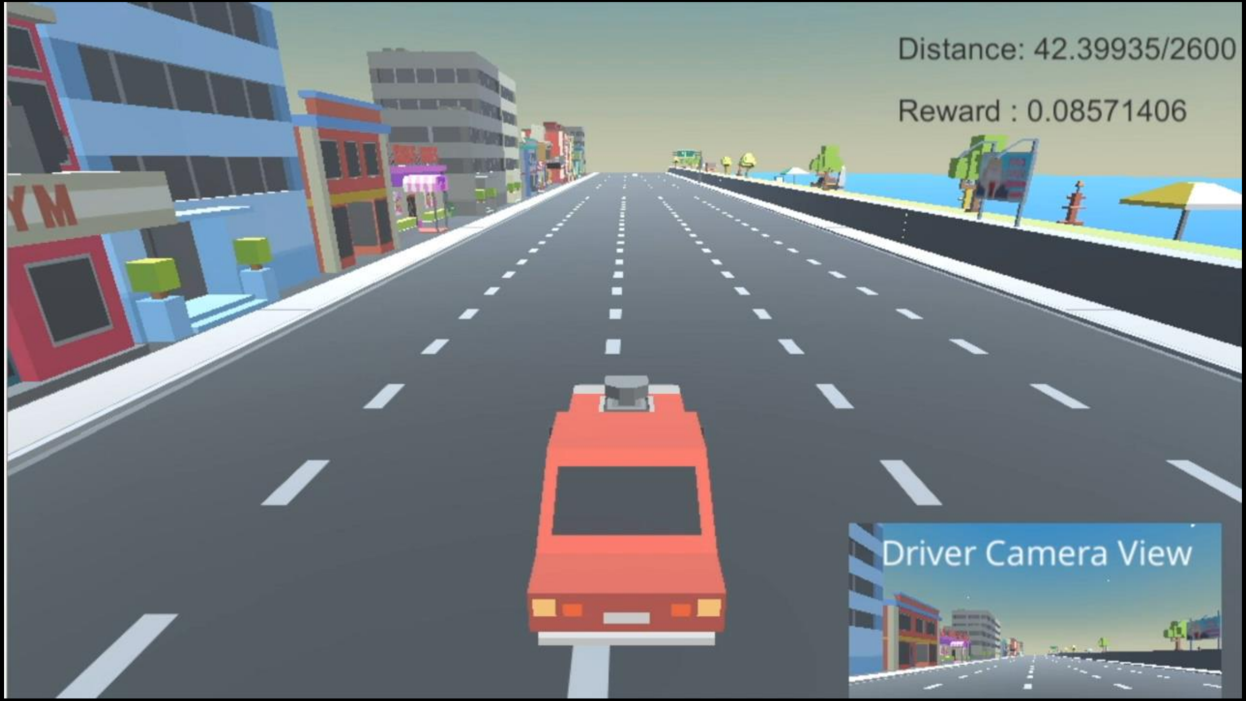
Distance: 42/2600

Reward : 0.02857143

Top Camera View

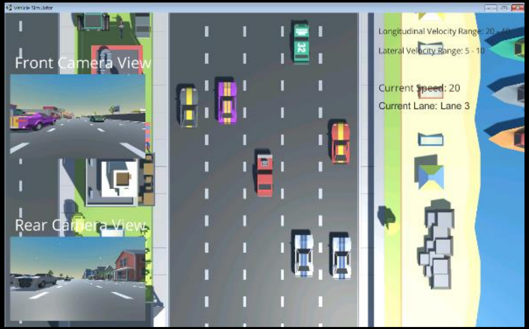
Development Case using Unity ML-Agents

ML-Agents Challenge



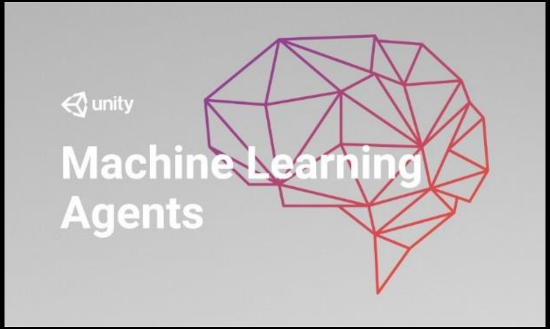
Development Case using Unity ML-Agents

2018 IEEE IV Conference



Longitudinal Velocity Range: 20 - 40
Lateral Velocity Range: 5 - 10
Current Speed: 20
Current Lane: Lane 3

Vehicle Simulator

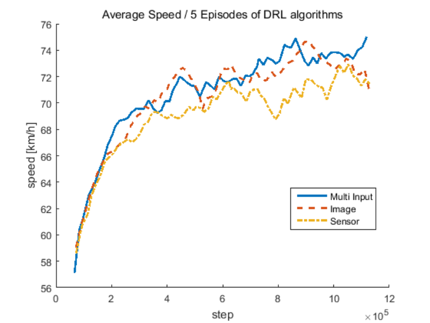


Unity ML-agents

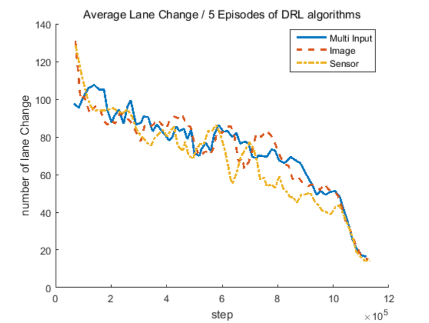
Development Case using Unity ML-Agents

2018 IEEE IV Conference

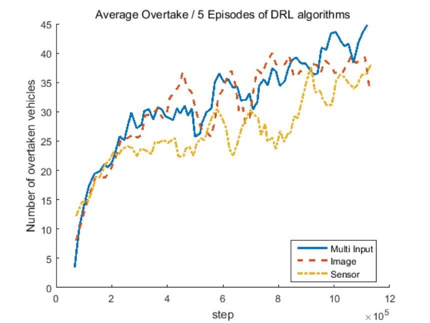
Speed



Lane Change



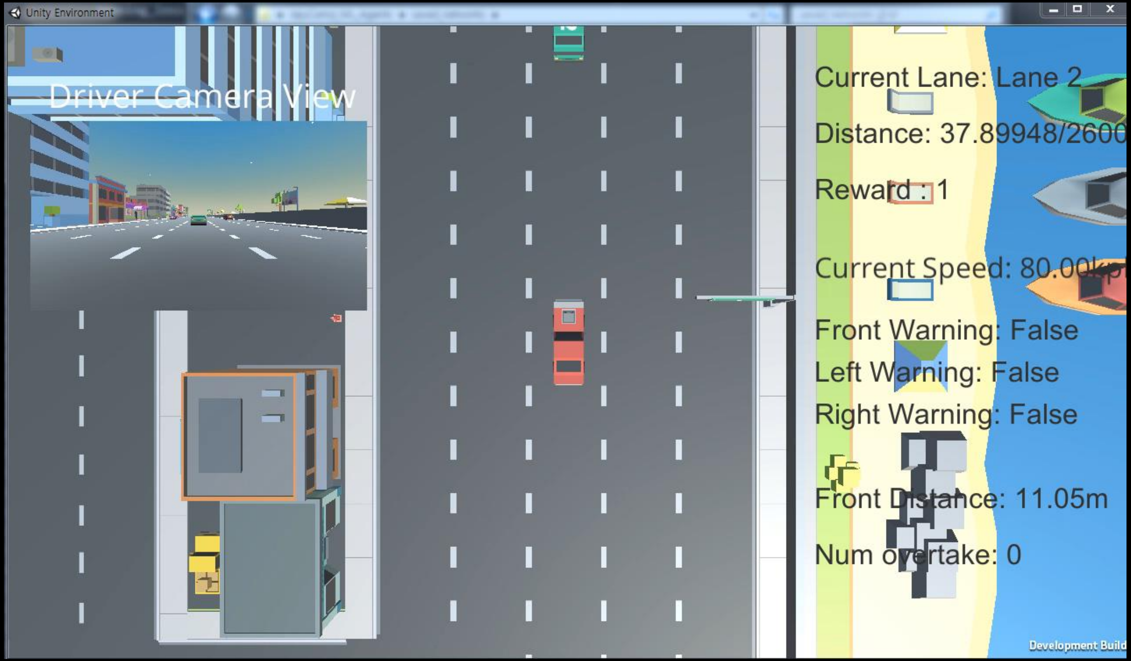
Overtake



Input Configuration	Average Speed (km/h)	# of Average Lane Change	# of Average Overtaking
Camera Only	71.0776	15	35.2667
LIDAR Only	71.3758	**14.2667**	38.0667
Multi-Input	**75.0212**	19.4	**44.8**

Development Case using Unity ML-Agents

2018 IEEE IV Conference



Current Lane: Lane 2

Distance: 37.89948/2600

Reward : 1

Current Speed: 80.00kp

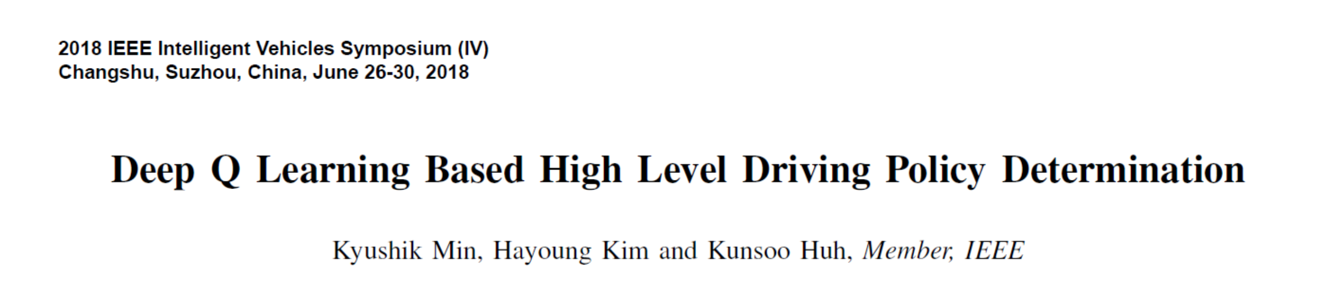
Front Warning: False
Left Warning: False
Right Warning: False

Front Distance: 11.05m

Num overtake: 0

Development Case using Unity ML-Agents

2018 IEEE IV Conference



2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Deep Q Learning Based High Level Driving Policy Determination

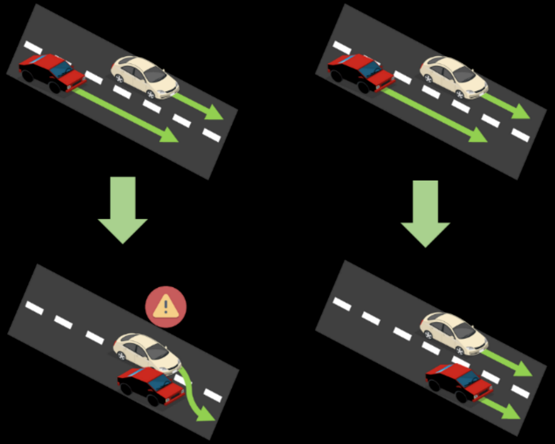
Kyushik Min, Hayoung Kim and Kunsoo Huh, *Member, IEEE*

IV2018
2018.06
26-29
Chang Shu
The 29th IEEE Intelligent Vehicles Symposium
Sponsored by the IEEE Intelligent Transportation Systems Society (ITSS)
IEEE
VIPA
QAII

Development Case using Unity ML-Agents

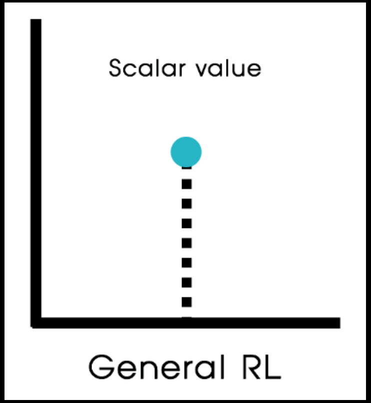
IEEE T-IV

- Driving situation is stochastic environment
- Even the same action in the same state can have different results!
- General RL: predicting the value as one scalar value



Same State
&
Same Action

Different
Result



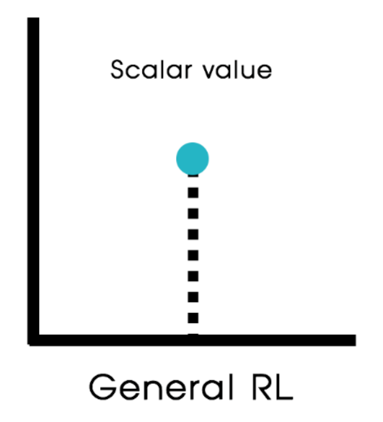
Development Case using Unity ML-Agents

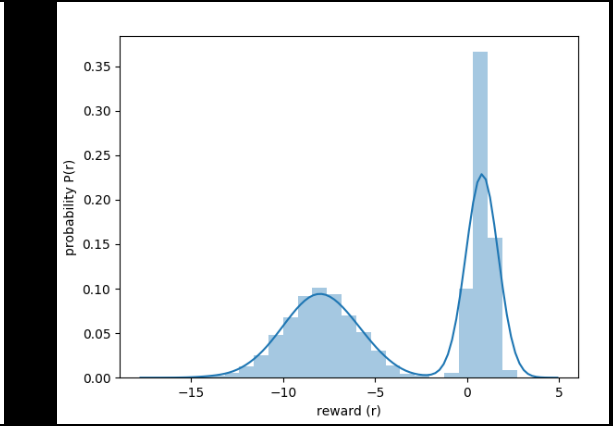
IEEE T-IV

- Distributional RL

  - Predict the value which agent will receive in the future as a probability distribution

  - Better performance in stochastic environments





Development Case using Unity ML-Agents

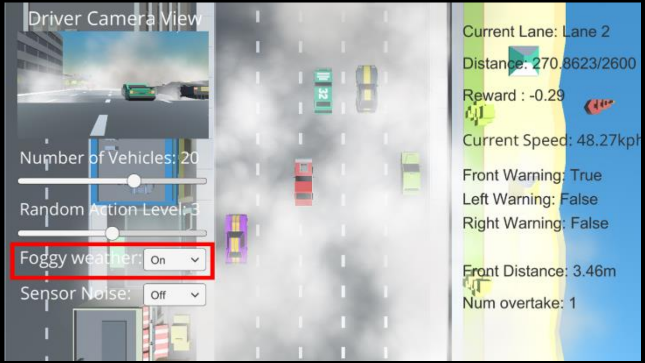
IEEE T-IV

- Simulation Environment

  - Add fog and sensor noise to verify the robustness of Distributional RL

  - Sensor noise equation: $d = d + \alpha * Random.Range(-d, d)$ (α: noise weight)

  - Add sensor noise and fog only when performing post-training algorithm validation



Driver Camera View

Number of Vehicles: 20

Random Action Level: 3

Foggy weather: On

Sensor Noise: Off

Current Lane: Lane 2

Distance: 270.8623/2600

Reward : -0.29

Current Speed: 48.27kph

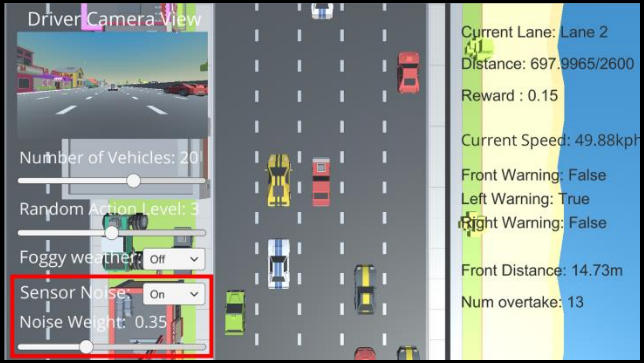
Front Warning: True

Left Warning: False

Right Warning: False

Front Distance: 3.46m

Num overtake: 1



Driver Camera View

Number of Vehicles: 20

Random Action Level: 3

Foggy weather: Off

Sensor Noise: On

Noise Weight: 0.35

Current Lane: Lane 2

Distance: 697.9965/2600

Reward : 0.15

Current Speed: 49.88kph

Front Warning: False

Left Warning: True

Right Warning: False

Front Distance: 14.73m

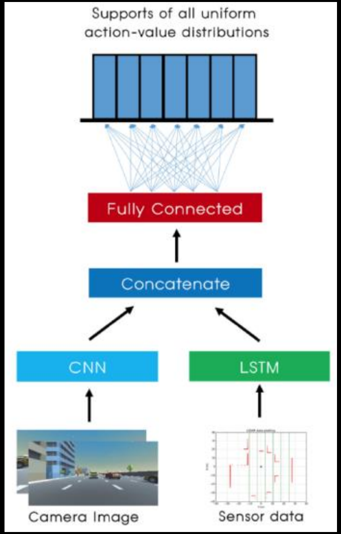
Num overtake: 13

Development Case using Unity ML-Agents

IEEE T-IV

- Network Architecture

  - Use QR-DQN, one of the Distributional RL algorithms



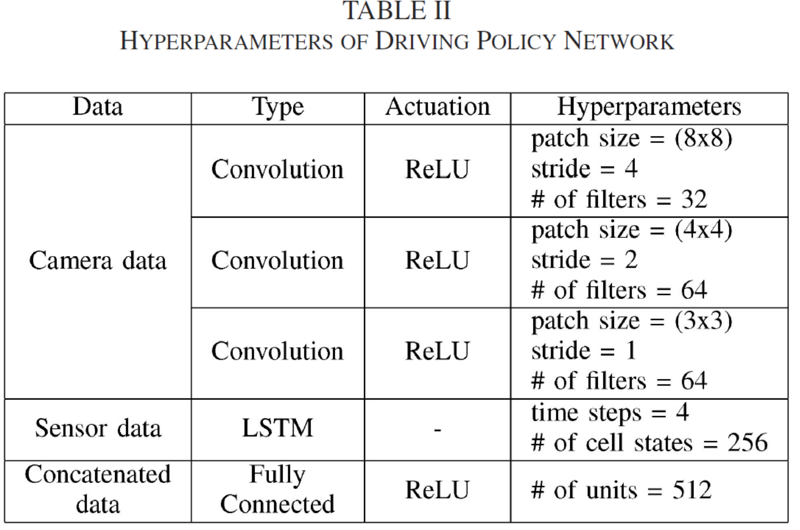


TABLE II
HYPERPARAMETERS OF DRIVING POLICY NETWORK

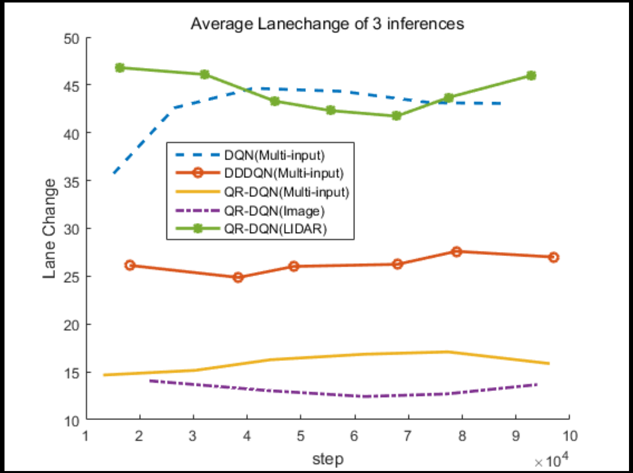
Data	Type	Actuation	Hyperparameters
Camera data	Convolution	ReLU	patch size = (8x8) stride = 4 # of filters = 32
	Convolution	ReLU	patch size = (4x4) stride = 2 # of filters = 64
	Convolution	ReLU	patch size = (3x3) stride = 1 # of filters = 64
Sensor data	LSTM	-	time steps = 4 # of cell states = 256
Concatenated data	Fully Connected	ReLU	# of units = 512

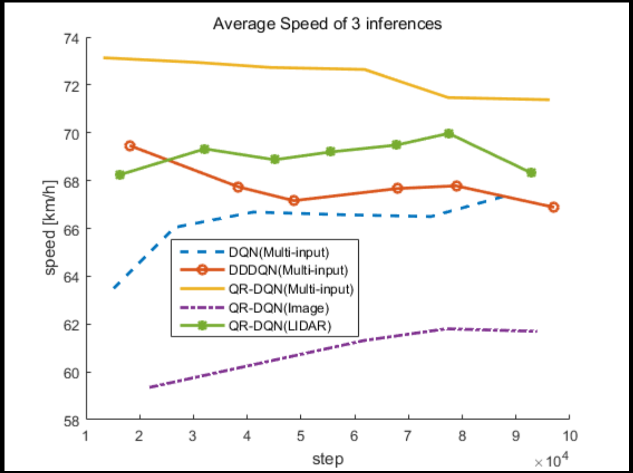
Development Case using Unity ML-Agents

IEEE T-IV

- Result

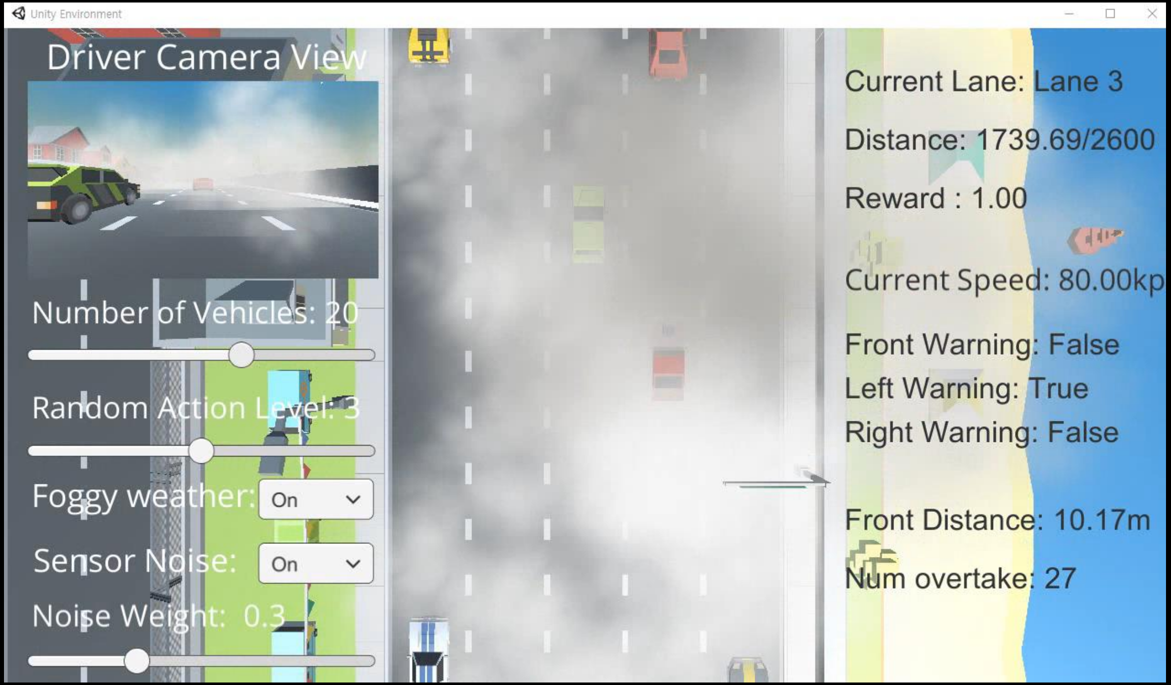
  - Fast average speed, minimum unnecessary lane changes





Development Case using Unity ML-Agents

IEEE T-IV



Number of Vehicles: 20

Random Action Level: 3

Foggy weather: On

Sensor Noise: On

Noise Weight: 0.3

Current Lane: Lane 3

Distance: 1739.69/2600

Reward : 1.00

Current Speed: 80.00kp

Front Warning: False

Left Warning: True

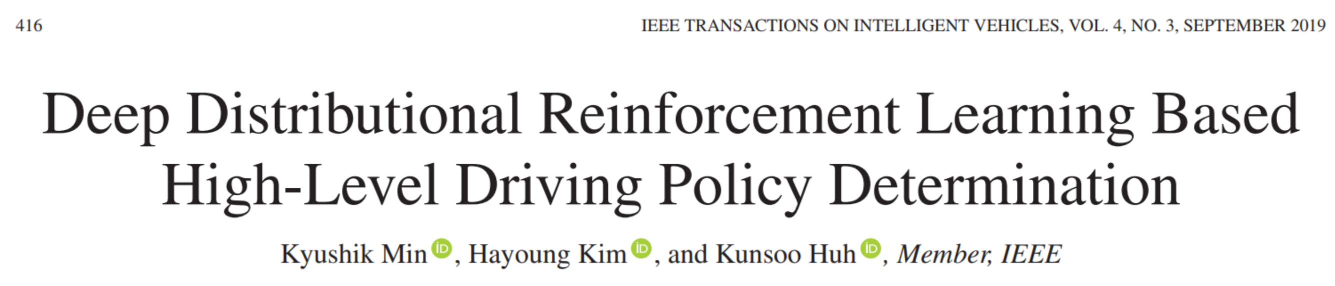
Right Warning: False

Front Distance: 10.17m

Num overtake: 27

Development Case using Unity ML-Agents

IEEE T-IV



416 IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, VOL. 4, NO. 3, SEPTEMBER 2019

Deep Distributional Reinforcement Learning Based High-Level Driving Policy Determination

Kyushik Min, Hayoung Kim, and Kunsoo Huh, *Member, IEEE*

IEEE Transactions on Intelligent Vehicles. Vol. 4, No. 3, Sep 2019

Development Case using Unity ML-Agents

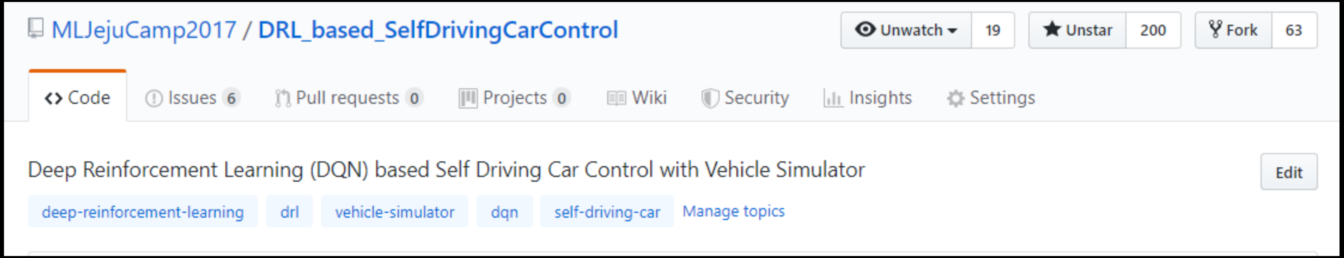
Github

- Upload the following items to Github!

  - RL Algorithms

  - Built Unity Environment

  - Unity Files



MLJejuCamp2017 / DRL_based_SelfDrivingCarControl

Deep Reinforcement Learning (DQN) based Self Driving Car Control with Vehicle Simulator

deep-reinforcement-learning drl vehicle-simulator dqn self-driving-car Manage topics

https://github.com/MLJejuCamp2017/DRL_based_SelfDrivingCarControl

Development Case using Unity ML-Agents

2019 IEEE ISPACS

- Multi-Agent Traffic Control Environment

  - Difficult to change lanes to desired lane in complex road situations

  - Overall traffic may slow down during lane changes

  - Lane changes in complex situations can lead to accidents

- Goal

  - Control multiple vehicles at the same time to move a specific vehicle to the target lane!

  - Minimize the overall vehicle speed reduction

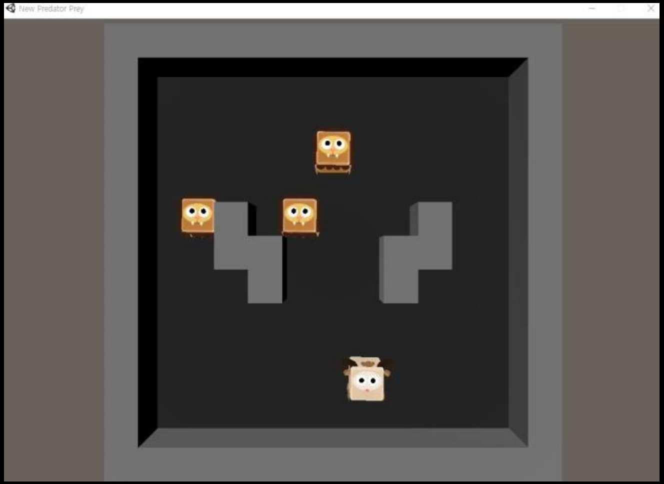
  => Multi-Agent Reinforcement Learning

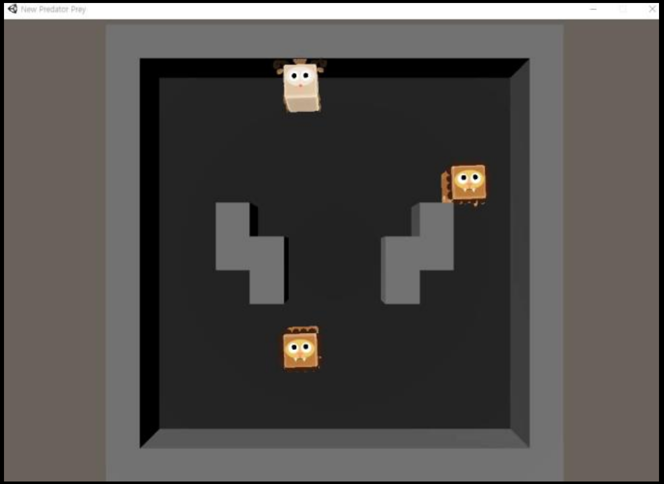
Development Case using Unity ML-Agents

2019 IEEE ISPACS

- Predator-Prey Environment

  - Predators are learned to hunt the prey and Prey are leaned to runs away from the predators

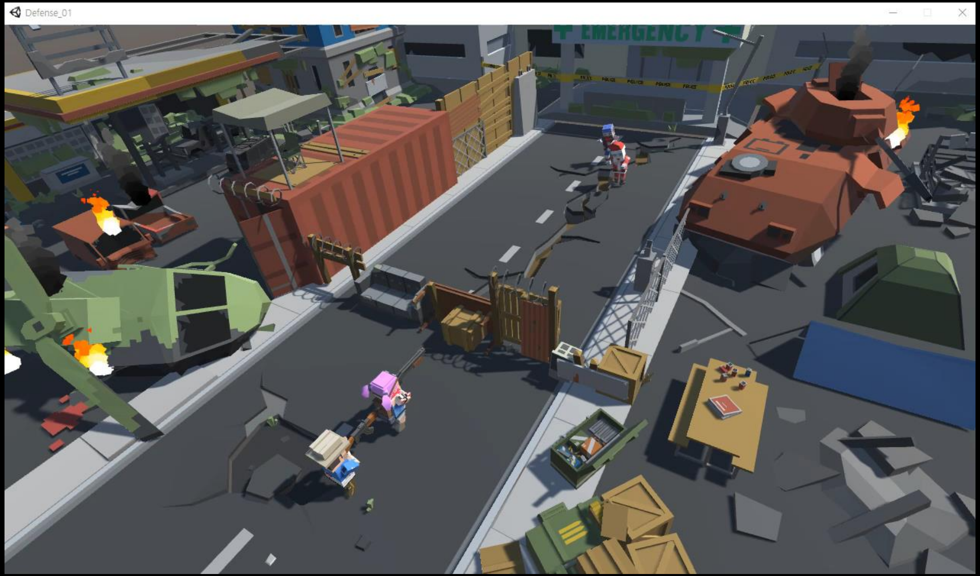




Development Case using Unity ML-Agents

2019 IEEE ISPACS

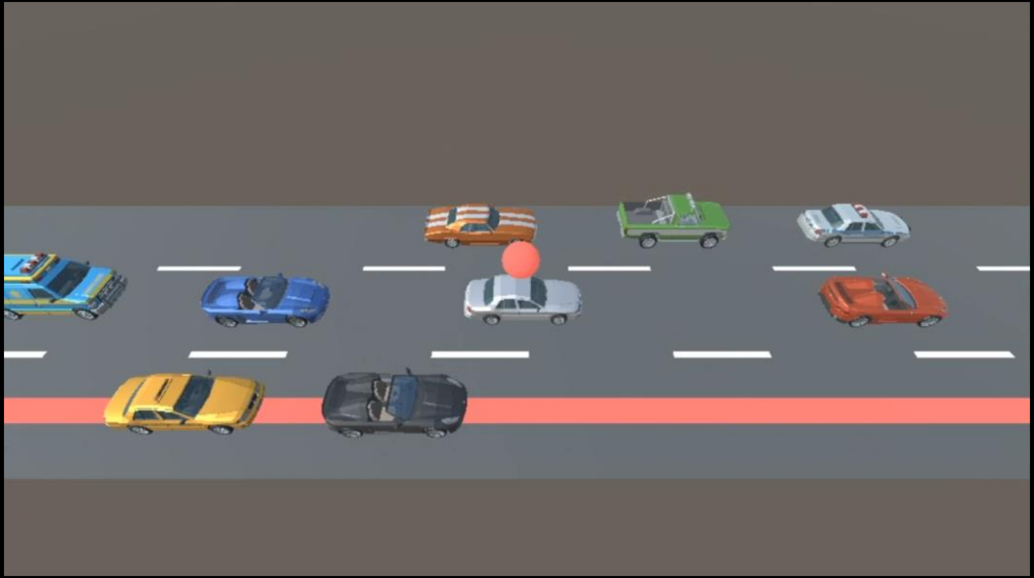
- Zombie Defense Environment



Development Case using Unity ML-Agents

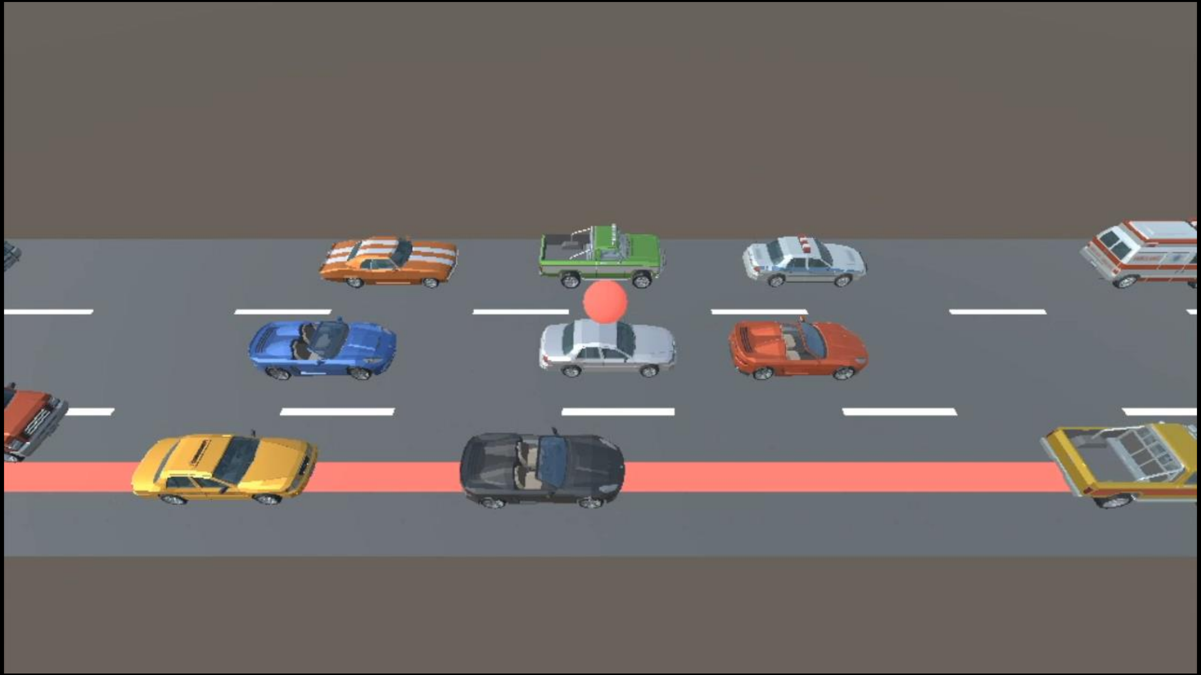
2019 IEEE ISPACS

- Multi-Agent Traffic Control Environment



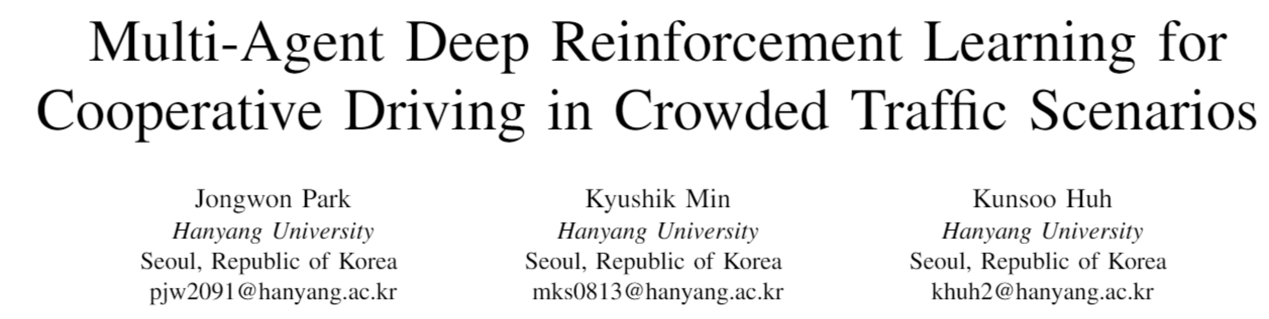
Development Case using Unity ML-Agents

2019 IEEE ISPACS



Development Case using Unity ML-Agents

2019 IEEE ISPACS



Multi-Agent Deep Reinforcement Learning for Cooperative Driving in Crowded Traffic Scenarios

Jongwon Park
Hanyang University
Seoul, Republic of Korea
pjw2091@hanyang.ac.kr

Kyushik Min
Hanyang University
Seoul, Republic of Korea
mks0813@hanyang.ac.kr

Kunsoo Huh
Hanyang University
Seoul, Republic of Korea
khuh2@hanyang.ac.kr

ISPACS 2019

Beitou, Taipei
Dec. 3-6, 2019

Development Case using Unity ML-Agents

Extra Research

Conclusion

- Are you curious about reinforcement learning? Come to RLKorea!
- Unity ML-Agents make it easy to create RL environments!

  - Easy environment creation using Unity

  - Stable communication between Unity environment and Python code

  - Support for creating a variety of RL environments (Multi-Agent, Curriculum,…)
- Perform various studies using ML-Agents

  - Create a variety of game and vehicle environments

  - RL performance verification using ML-Agents

THANK YOU

SOSCON2019

SAMSUNG OPEN SOURCE CONFERENCE 2019