Train in Austria, Race in Montecarlo: Generalized RL for Cross-Track F1tenth LIDAR-Based Races

Michael Bosello, Rita Tse, Giovanni Pau

澳門理工學院
Instituto Politécnico de Macau
Macao Polytechnic Institute

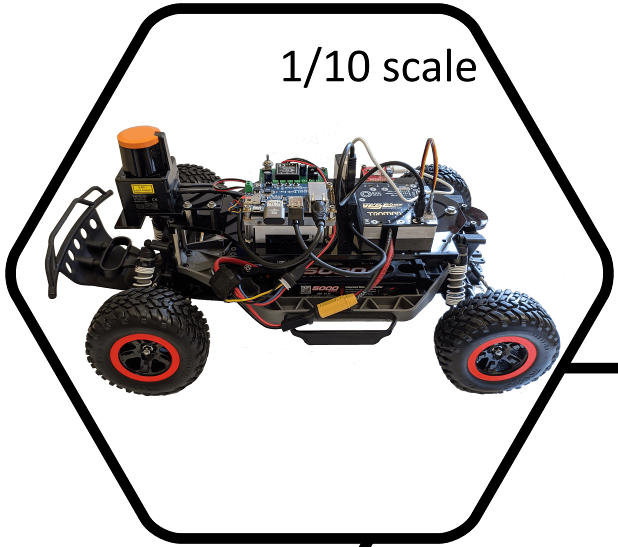
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

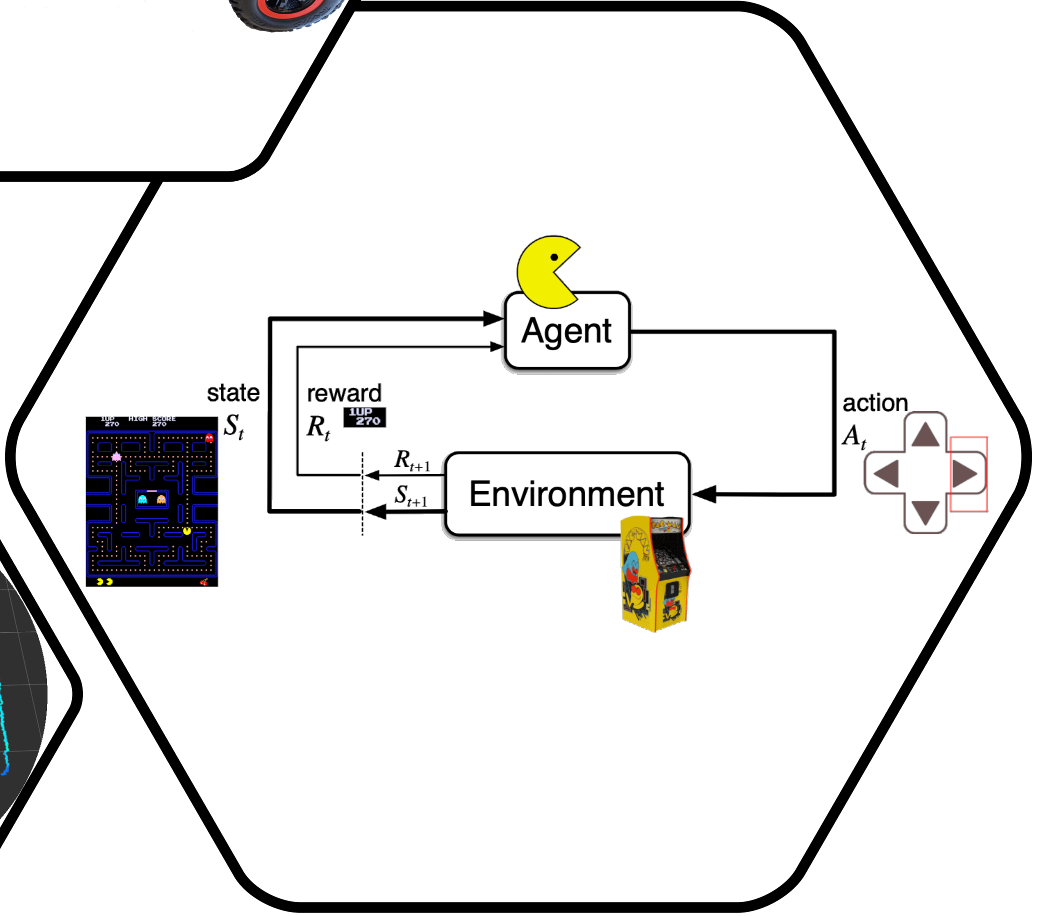
UCLA
Computer Science

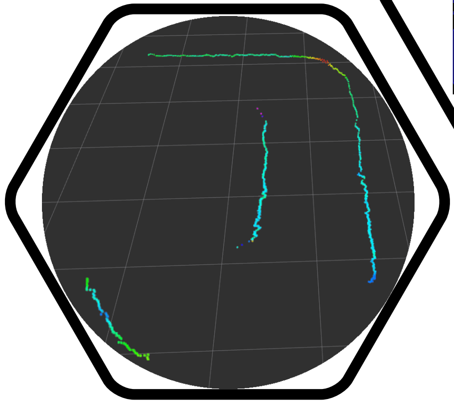
Autonomous Driving

Study Components:

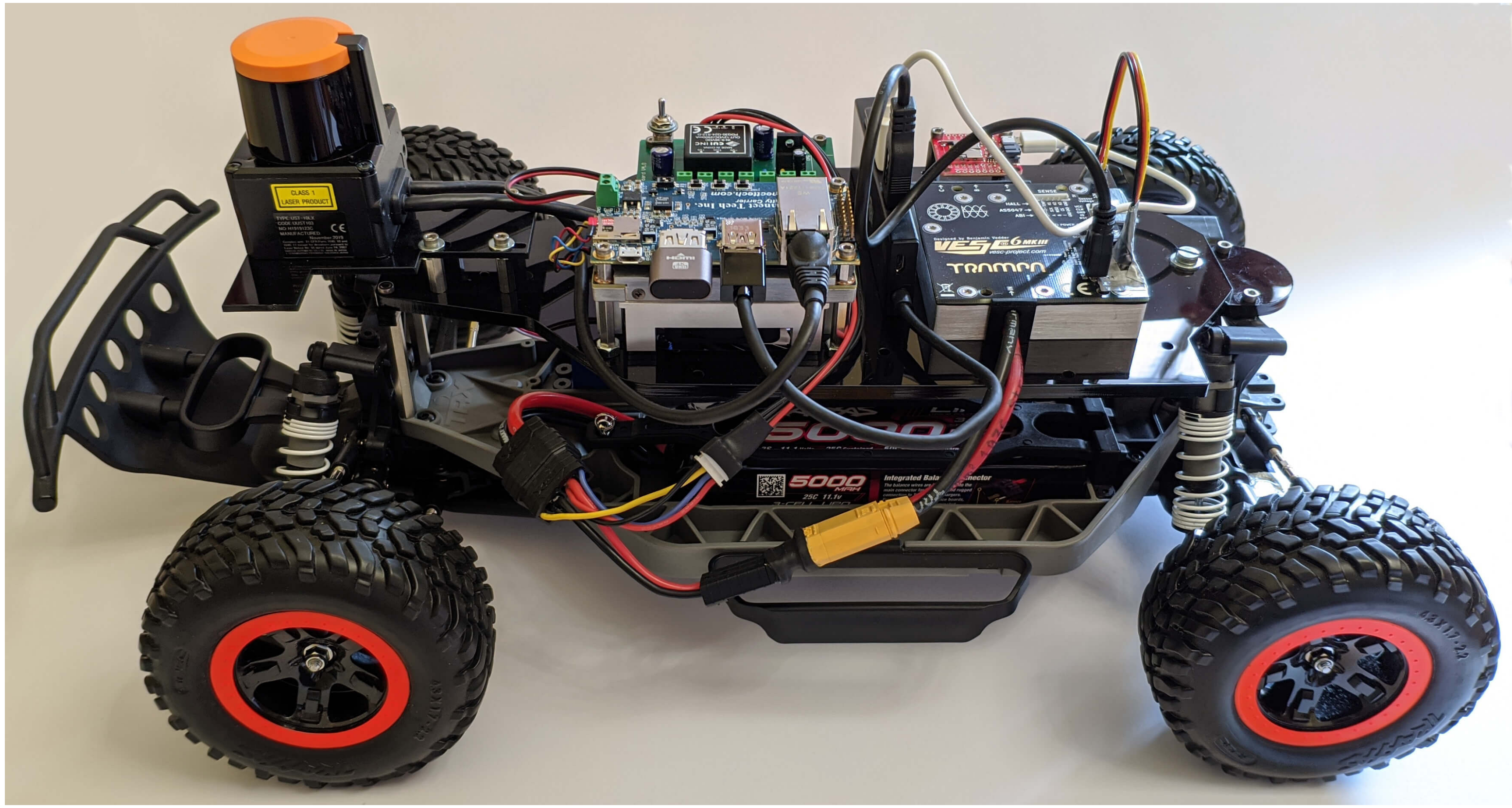
- Physical racing car
- Reinforcement Learning
- LIDAR







F1TENTH: 1/10th scale Autonomous Racing



- The agent still faces challenges of a real driving scene
- Inexpensive
- Safe

Very realistic 1/10 scale car prototype

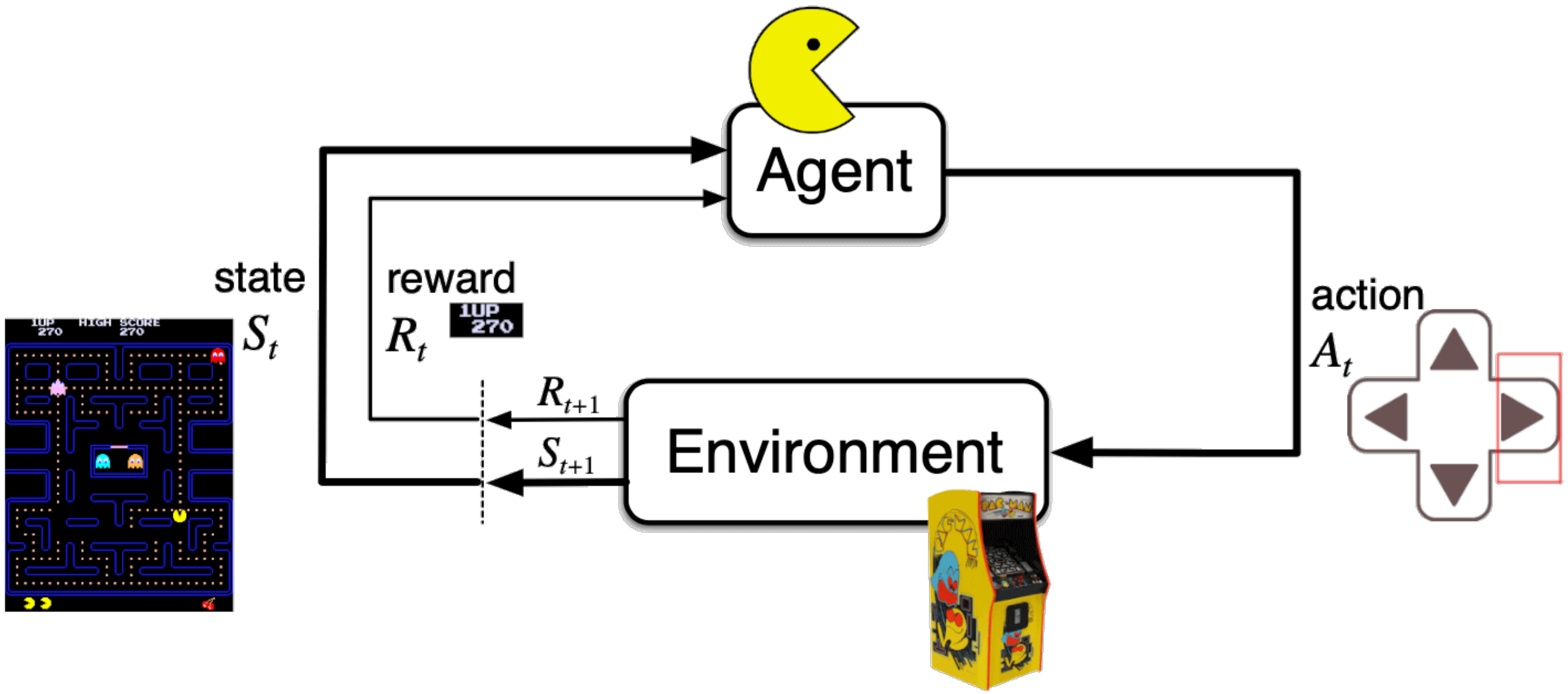
- Hardware/software stacks similar to full-scale solutions
- Ackermann steering
- High speeds

The same code can run on the ad-hoc simulator

Reinforcement Learning Basics

An agent learns how to fulfil a task by interacting with its environment

Sometimes, an agent has only a partial view of the world state



Episode

A complete run from an initial state to a final one

Learning a Policy

A map from states to actions aiming to maximize the cumulative reward (sum of rewards over time)

A way to produce a policy is to estimate the action-value function

- *(state, action)* → *expected return* (expectation of future rewards)
- Given the function, the agent needs only to perform the action with the greatest value

Q-learning

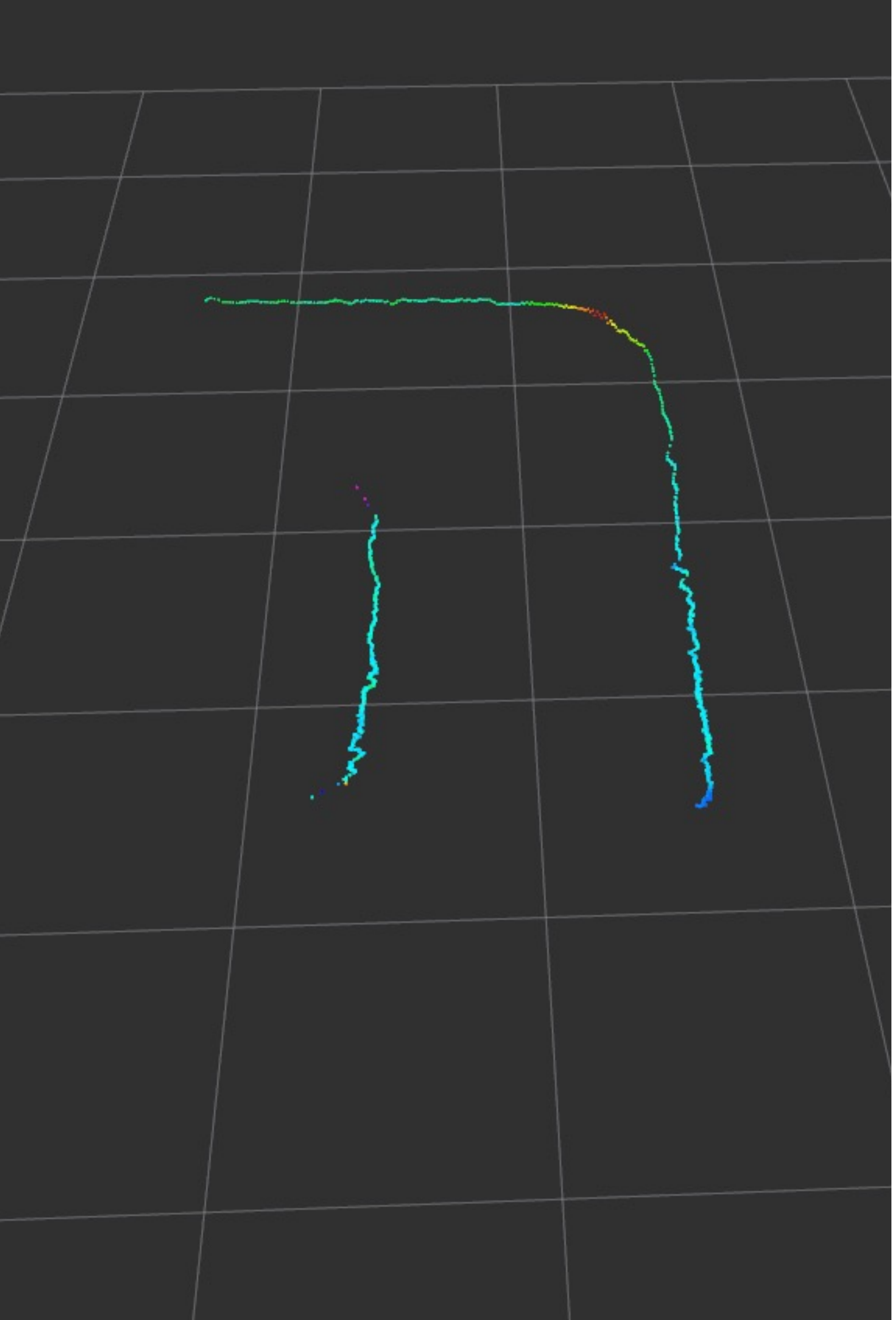
- Is an algorithm that estimates the action-value function
- At every iteration the estimation is refined thanks to the new experience.
- Every time the evaluation becomes more precise, the policy gets closer to the optimal one.

DQN

- When the number of states increases, it become impossible to use a table to store the Q-function
- A neural network can approximate the Q-function
- We also get better generalization and the ability to deal with non-Markovian envs

The exploration/exploitation dilemma

- The agent has to behave optimally but it needs to explore the environment to improve
- ε-greedy policy: the agent behave greedily but there is a (small) ε probability to select a random action.



LIDAR (Light Detection And Ranging)

Sensor for measuring distances

- Emits focused light beams
- Measure the time of flight

- 2D: map of the azimuth at a fixed height
- 3D: point clouds

- LIDAR measurements are greatly affected by reflection
  - When a ray gets reflected, it appears as it is no obstacle in that direction
  - RL training with real LIDAR data has been considered an open problem

Hokuyo UST-10LX

- 270° field of view
- 0.25° angular resolution
- 1081 scan rays
- 10m detection range
- ±40mm accuracy
- 25ms scan speed

Autonomous Driving State of the Art

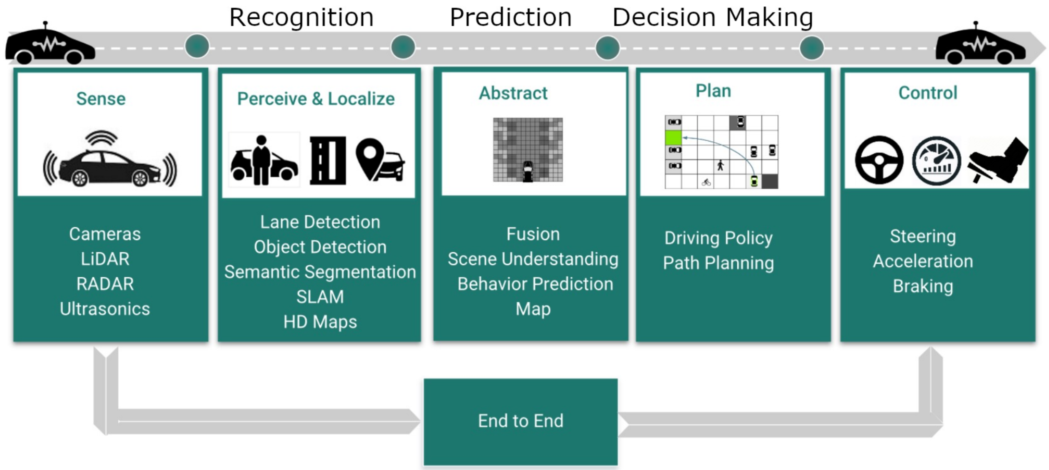
is still an open problem

Environment
- Non-deterministic
- Partially observable
- Dynamic

Supervised Learning
- Based on imitation
- Current approach by major car manufacturer

Drawbacks
- Training data
  - huge amounts of labelled data or human effort
- Covering all possible driving scenarios is very hard

- Single task handling
  - ➢ Use human ingenuity to inject knowledge about the domain
- End-to-end
  - ➢ Let the algorithm optimize towards the final goal without constrains



Why Reinforcement Learning?

- Learns by interacting with the environment through trial-and-error
  - ➢ Does not require explicit supervision from humans
- RL is specifically formulated to handle the agent-environment interaction
  - ➢ Natural approach for learning robotics (and autonomous driving)

- Mainly used on simulated environment
  - to avoid consequences in real-life
  - transferring learning from simulations to the real world is a hard problem
    - simulated and real data have not the same distribution
    - agents trained in a synthesized world often fail to generalize
      - if appropriate domain adaptation measures are not taken.

Related Works: RL and AD

- Multiple works applied RL to racing games using both end-to-end and single task handling
- RL used to solve single tasks like motion planning, overtaking, merging, lane change, lane keep, and automated parking

Sim2real

- Some work on transfer learning
- Growing interest in training RL agents directly in real-world robotic systems
  - We trained an RL agent directly in the real-world using camera frames
  - Use of RL (DDPG) in a real car to learn a policy for line following
    - input comes from a front-facing camera
    - rewards $\propto$ to the distance travelled without human intervention
    - training in a real car using on-board computation
    - model able to learn a road segment in just half an hour

LIDAR and RL

LIDAR is widely used in AD thanks to progress in technology and the use of DL

- Little work in the context of RL-based driving.

- Paper on NN verification
  - Showed that LIDAR faults (due to reflection) prevent state-of-the-art RL from learning
- Study on the use of RL on the F1tenth platform
  - Run on complex F1 racetracks
  - Model-based and several model-free RL algorithm tested

Driver Agent Software

ROS (Robot Operating System)

- LIDAR data and actuator commands updated asynchronously
- Automatic emergency braking (TTC-based)
  - It has priority over RL decisions

Episode

- An episode ends when emergency braking activates
- The car goes backward, then a new episode starts

Action

- Go forward
- Turn right
- Turn left
- (Slowdown action)

➢ Fast learning

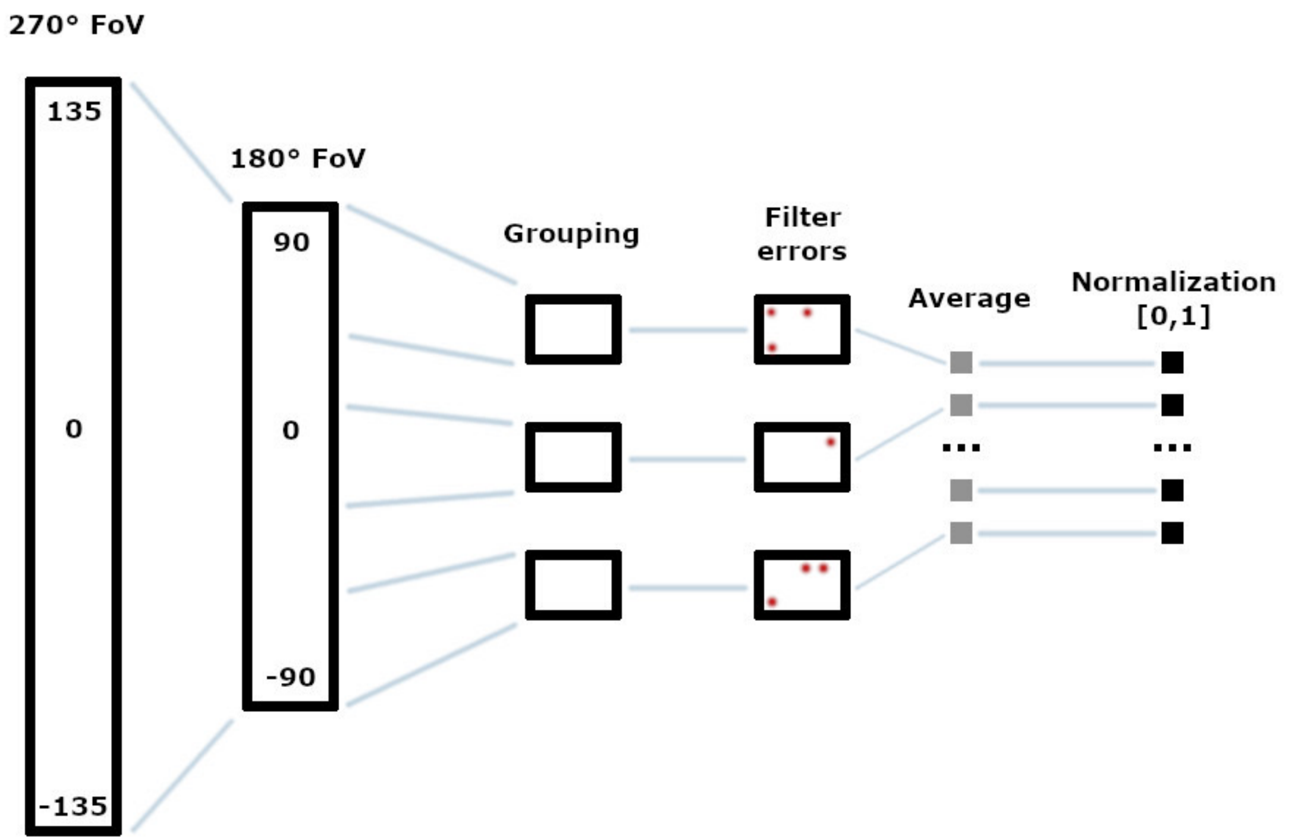
State

- Two subsequent LIDAR measurements (plus the car's velocity)
- Integration over time

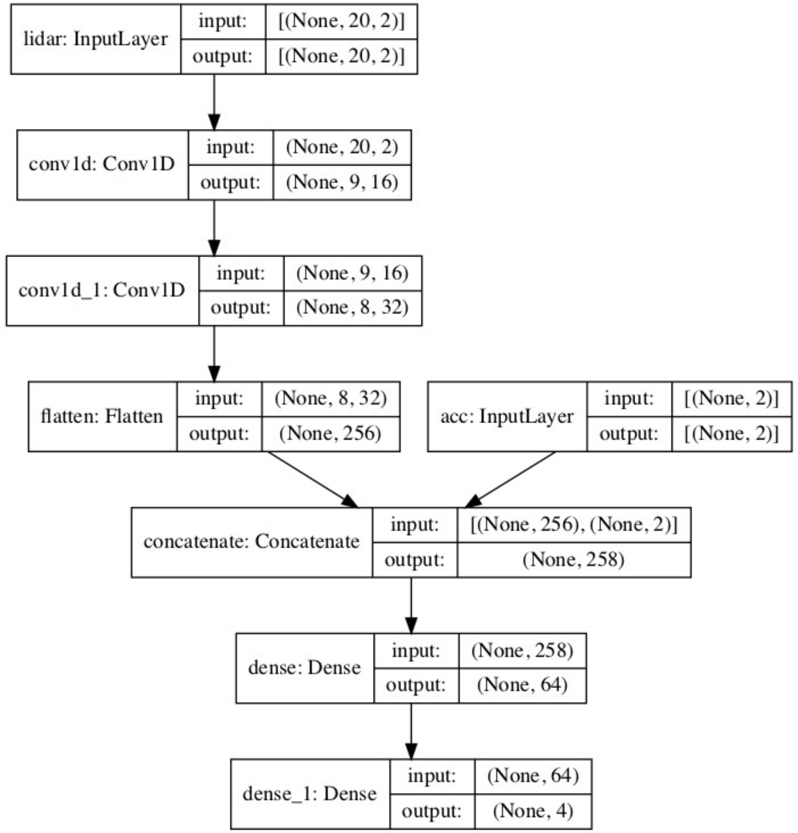
Rewards

- Emergency braking = -1
- Proportional to
  - car velocity (0.09 max)
  - distance to the nearest obstacle (0.01 max)

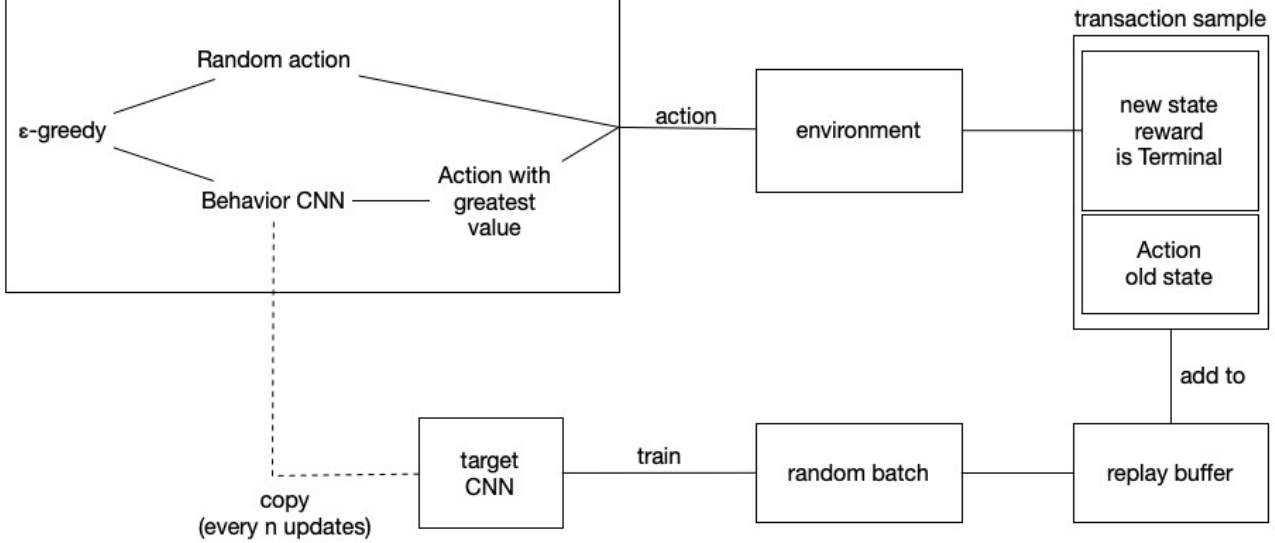
LIDAR Pre-Processing



1D CNN


lidar: InputLayer
input: [(None, 20, 2)]
output: [(None, 20, 2)]
conv1d: Conv1D
input: (None, 20, 2)
output: (None, 9, 16)
conv1d_1: Conv1D
input: (None, 9, 16)
output: (None, 8, 32)
flatten: Flatten
input: (None, 8, 32)
output: (None, 256)
acc: InputLayer
input: [(None, 2)]
output: [(None, 2)]
concatenate: Concatenate
input: [(None, 256), (None, 2)]
output: (None, 258)
dense: Dense
input: (None, 258)
output: (None, 64)
dense_1: Dense
input: (None, 64)
output: (None, 4)

DQN

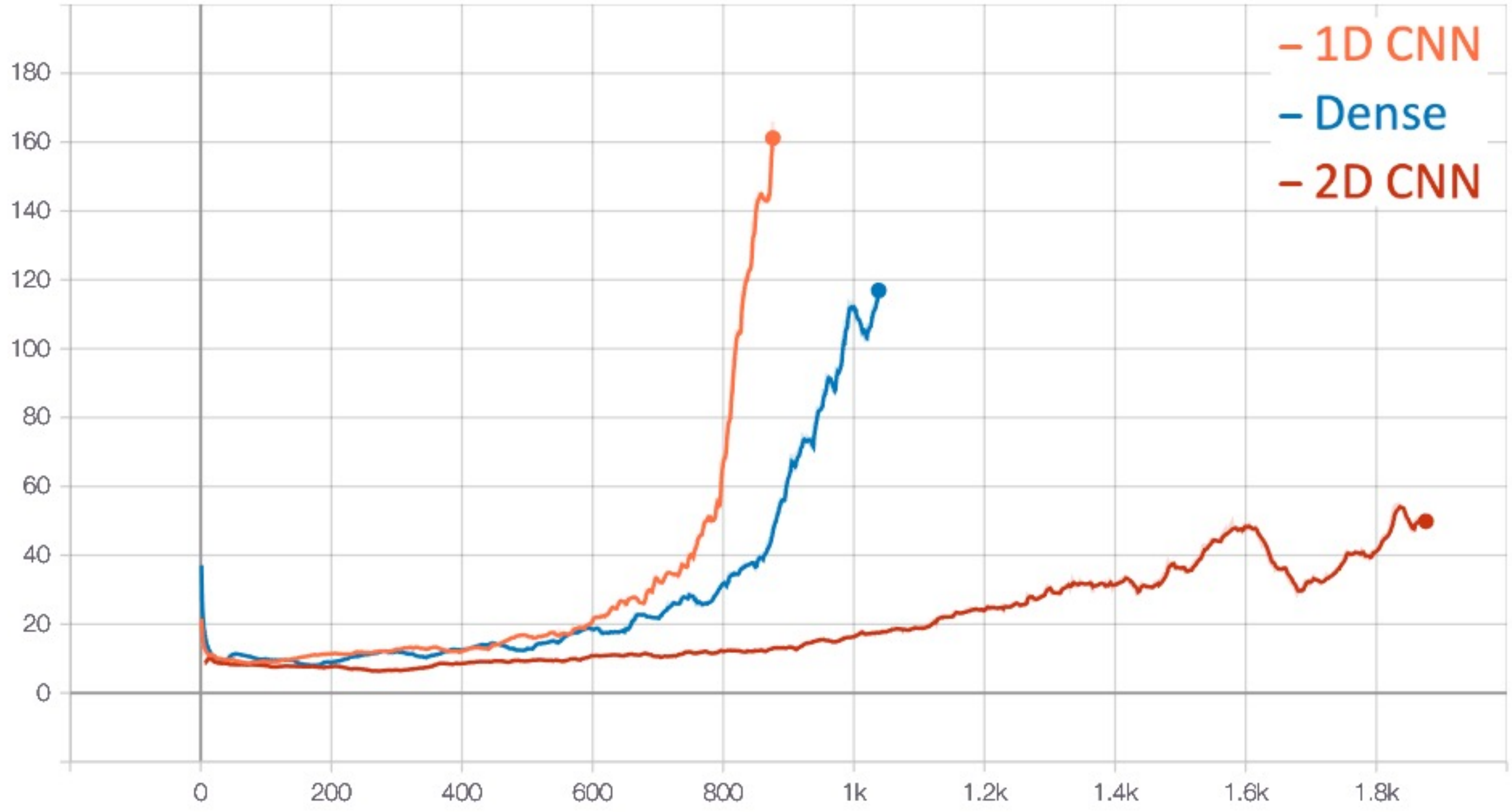

Random action
ε-greedy
Behavior CNN
Action with greatest value
action
environment
transaction sample
new state
reward
is Terminal
Action
old state
add to
replay buffer
random batch
train
target CNN
copy
(every n updates)

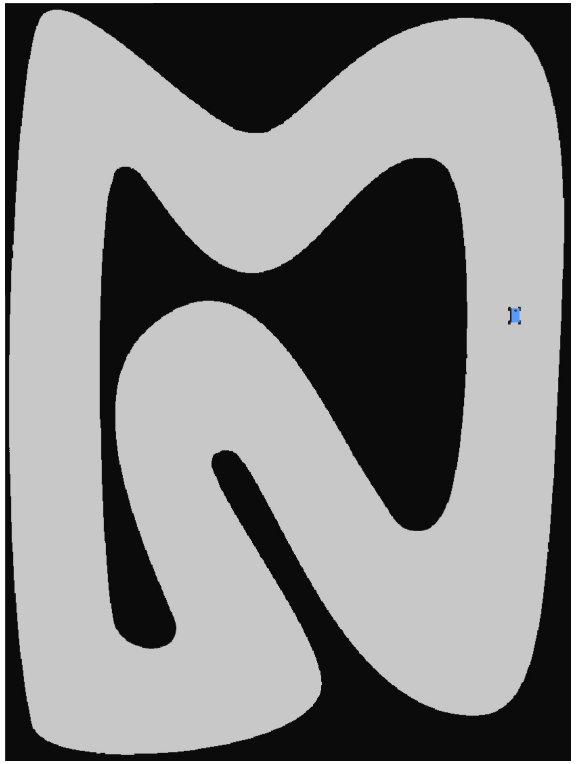
Experiments

- Assessing the approach by analyzing three factors
  - Finding the best NN for LIDAR data
  - Sim2real approaches
  - Performances, efficiency, generalization

NN comparison experiment

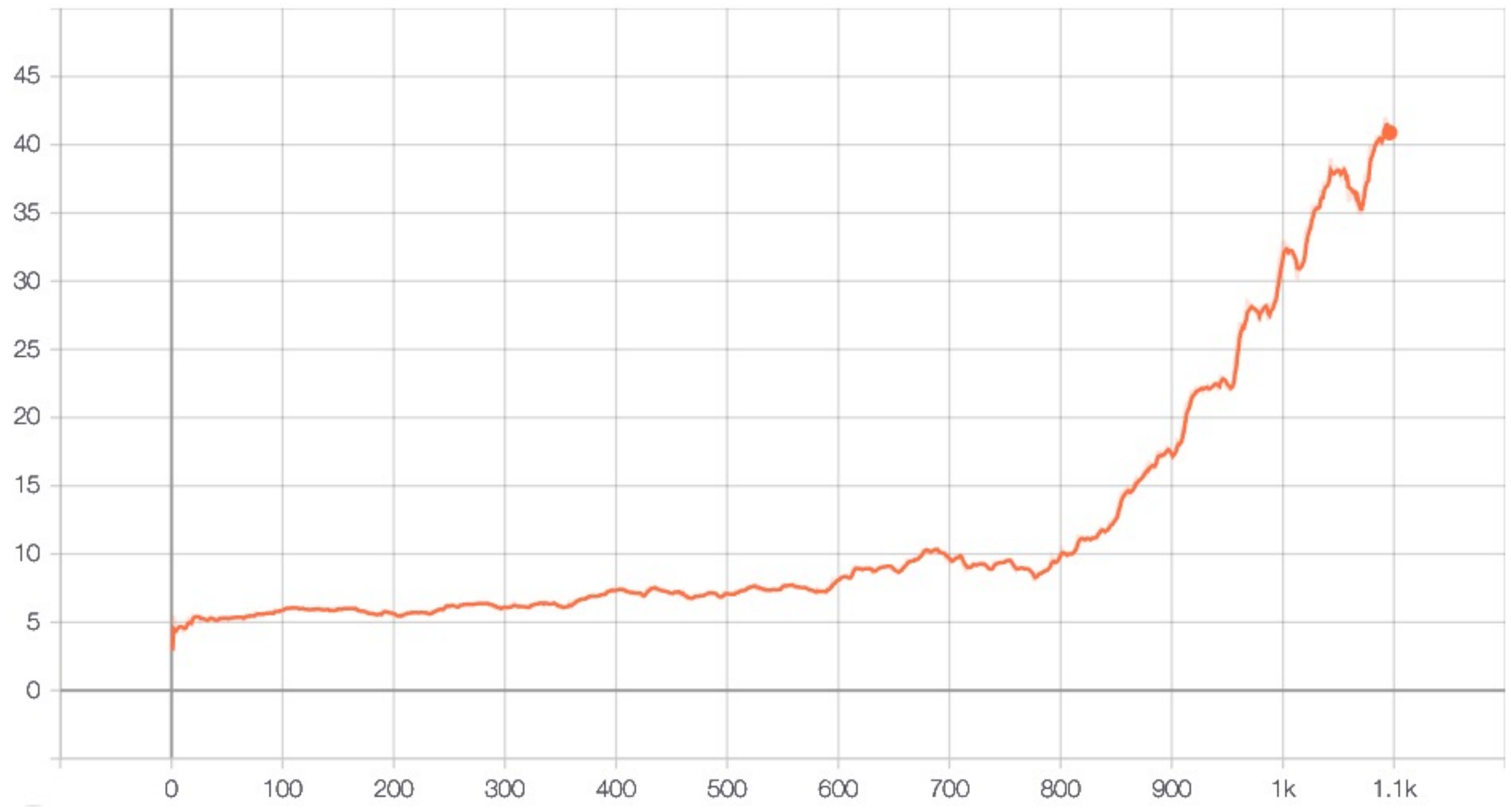
CNNs perform better on structured and spatially related data





Sim2real experiment 1: Training on the physical car

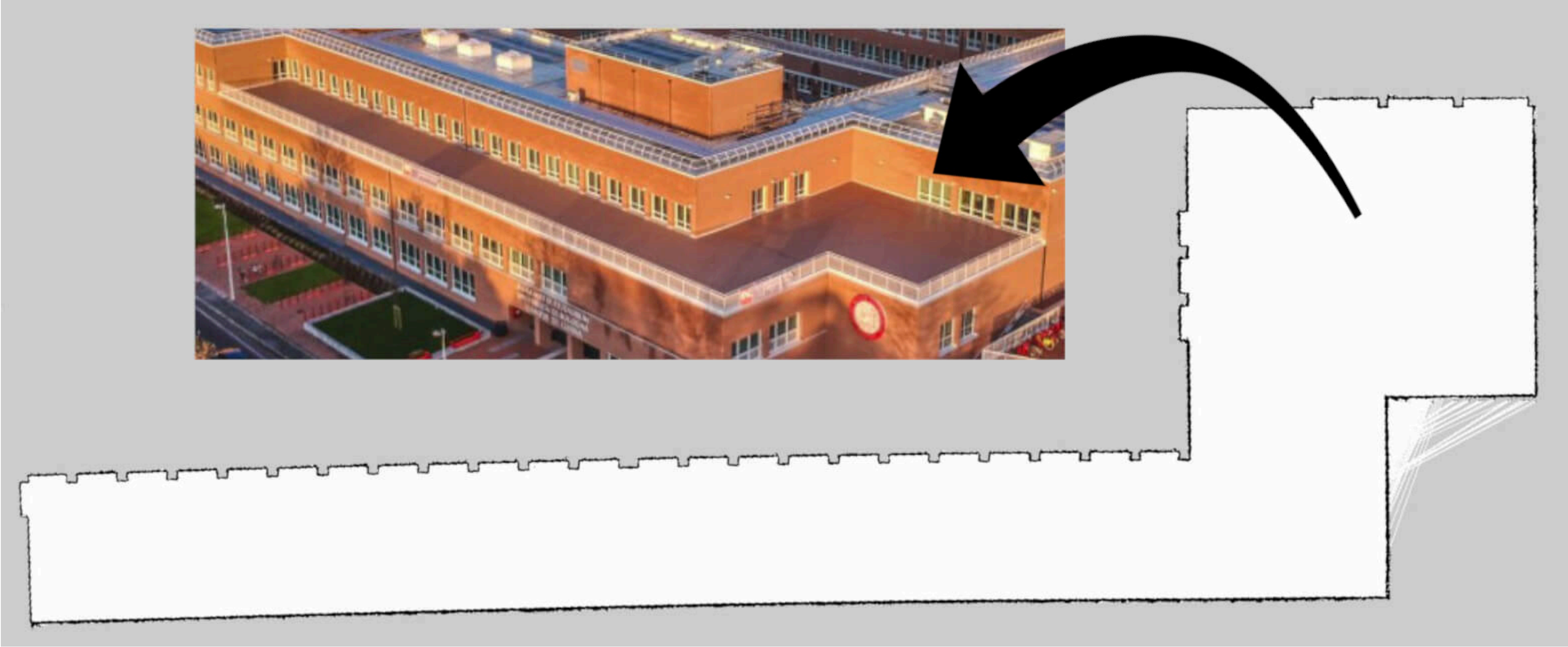
The driver-agent successfully learned a control policy **using real LIDAR data**



Sim2real experiment 2: Transfer learning

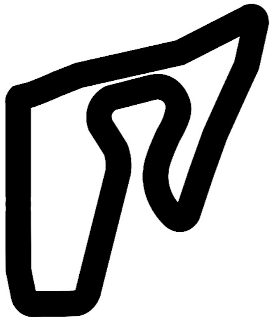
Without any additional training or changes

Challenge: real car dynamics at high speeds

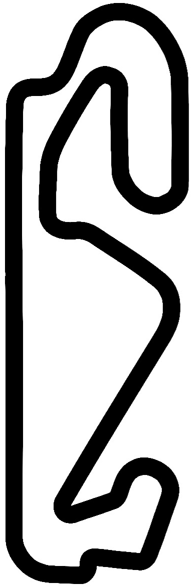


F1 racetracks experiment

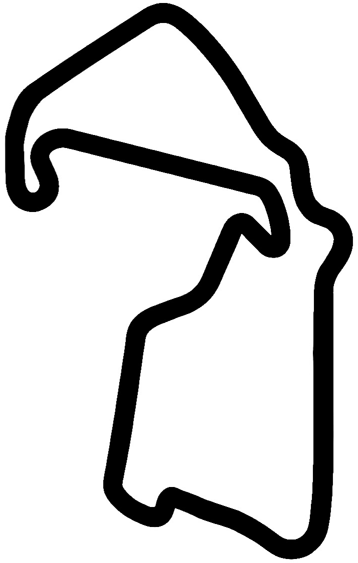
Red Bull Ring (AUT)



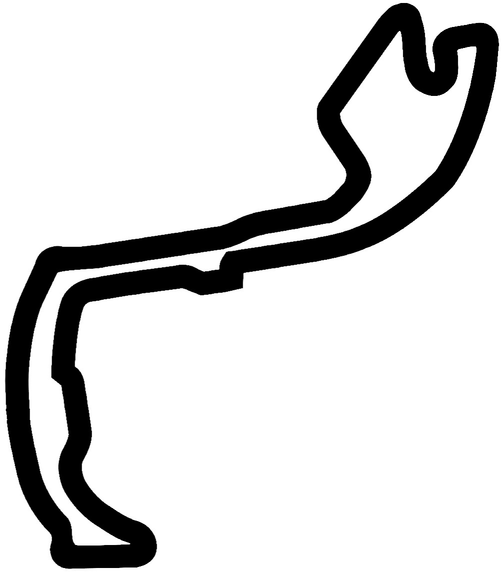
Circuit de Catalunya (BRC)



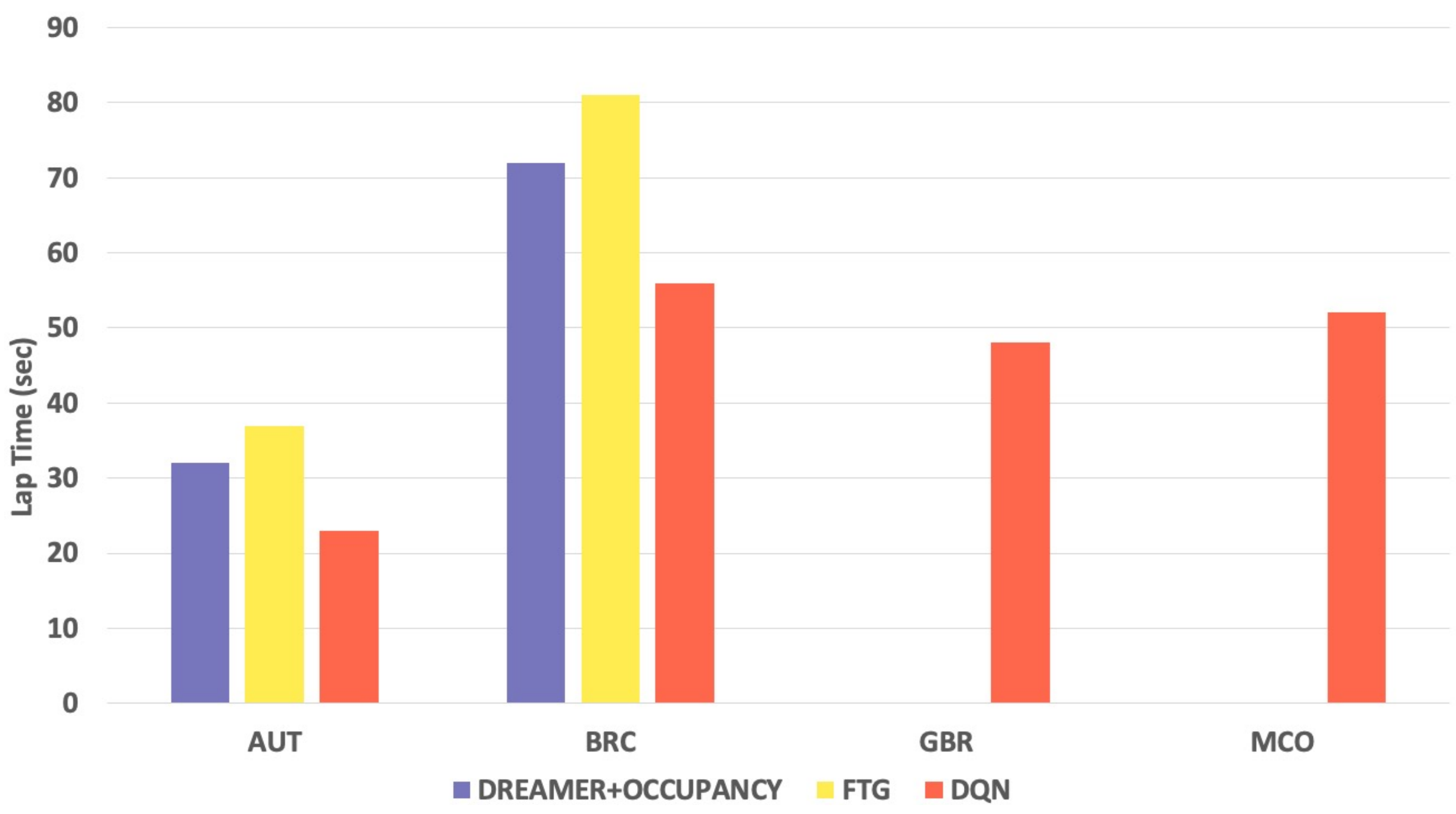
Silverstone Circuit (GBR)



Circuit de Monaco (MCO)



F1 Tracks Performance, Efficiency, and Generalization



D4PG, PPO, LTSTM-PPO, MPO, SAC were not able to complete the task

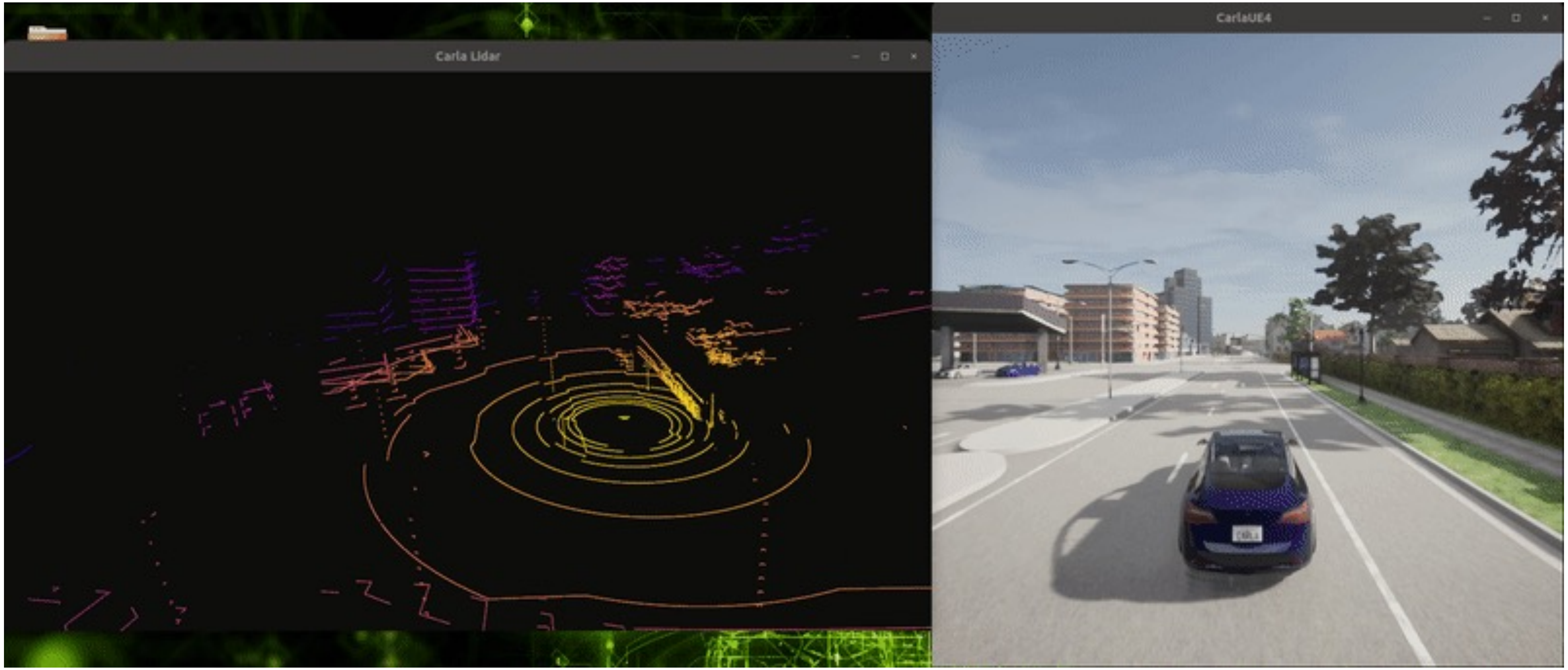
Sample efficiency

- DQN: 550k steps
- Dreamer: 2M steps
- Others: 8M steps

Future Works

Using 3D lidar in urban scenarios

- And transfer learning from simulation (CARLA) to the F1TENTH



Thank You

https://github.com/MichaelBosello/f1tenth-RL

michael.bosello@unibo.it