Final Project for the **Complex Systems: Models and Simulations** and **Artificial Intelligence** courses
at MSc of CS at **University of Milano-Bicocca**

AUTONOMOUS EXPLORATION AGENT

Authors:
Federico Bottoni
Nassim Habbash

Task

Definition of a **single-agent obstacle-ridden**, **procedurally generated** environment

Training of the agent with Proximal Policy Optimization for a **target localization objective**

Evaluation and comparison of the agent's **model variations**

Tools

Environment and agent architecture

unity

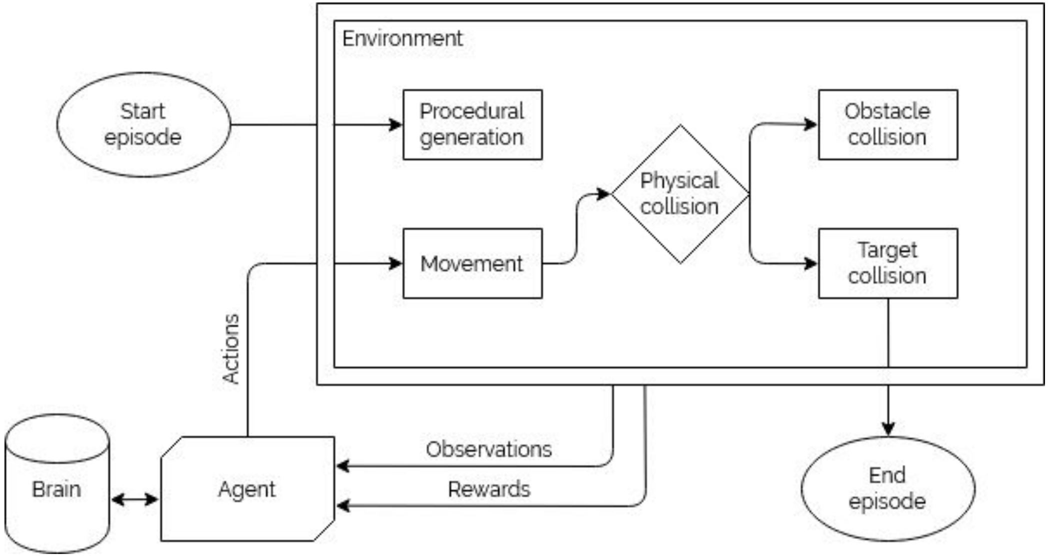
Reinforcement Learning

Unity's ML-Agents

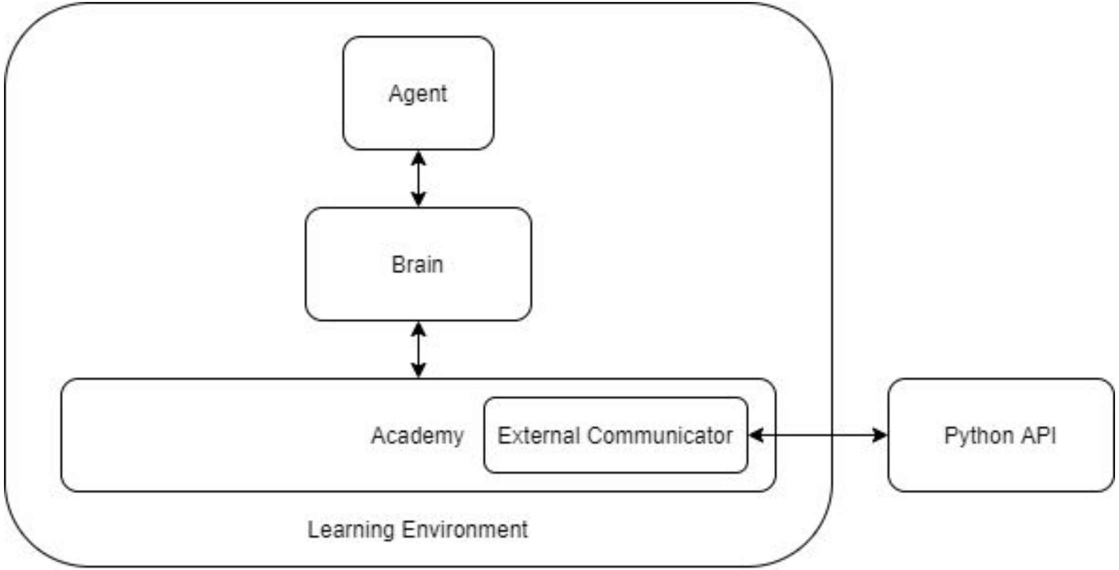
TensorFlow

System Architecture

Schema illustrating the **interactions** between the main **actors** of the system



System Architecture



ML-Agent Learning pipeline

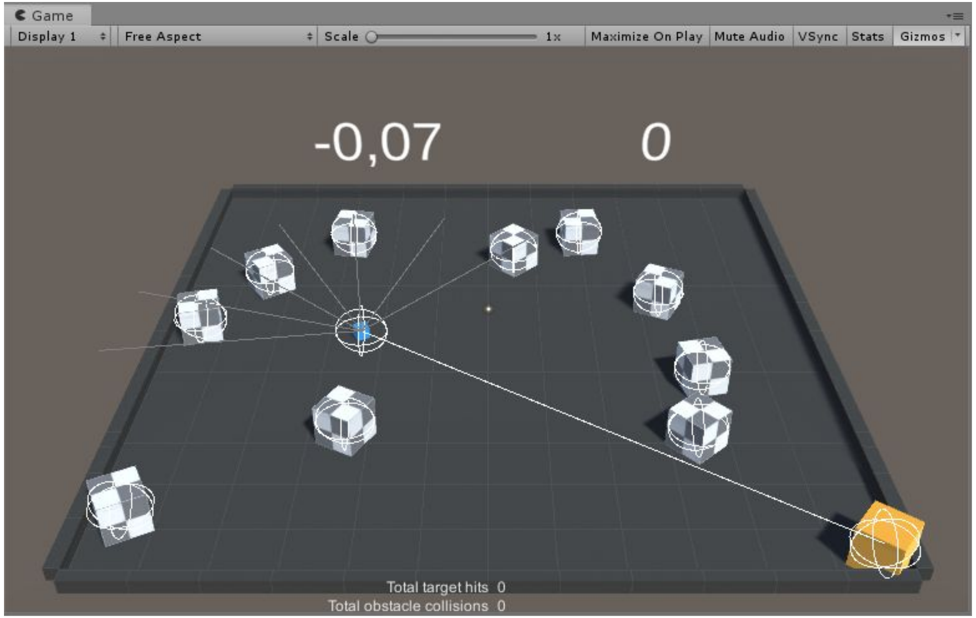
Simulation Main Phases

- **Environment procedural generation**
  - Parametrized spawning and positioning of the **Agent**, **Obstacles** and **Target**
- **Agent's Actions and Movement**
  - Agent movement controller built with Unity's **Physics System**
  - Actions inferred by ML-Agents **Brain** interface
  - Action space: **3D Discrete** (2D maneuvering case)
- **Collisions**
  - **Penalty** function for obstacle collision
  - **Reward** function and episode completion for target collision

Environment Procedural Generation

Constrained generation of the scene ensuring an approximately **uniform distribution** of the object in the environment according to different parameters, such as:

- Number of obstacles
- Minimum Target distance from the Agent
- Minimum distance between objects in the environment



Environment Procedural Generation

Agent parameters	
Dimensions	**1x1x1**
Max linear velocity	**5**
Max angular velocity	**5/3π**
Environment area parameters	
Level area	**50x50**
Obstacle dimensions	**8x8x8**
Target dimensions	**8x8x8**

2D Sensor parameters	
# LIDAR	**14**
Maximum range	**20**
Field of view	**[-2/3π, 2/3π]**
3D Sensor parameters	
# LIDAR	**42**
Maximum Range	**40**
Horizontal field of view	**[-2/3π, 2/3π]**
Vertical field of view	**[-1/3π, 1/3π]**

Static environmental parameters

2D Maneuvering System

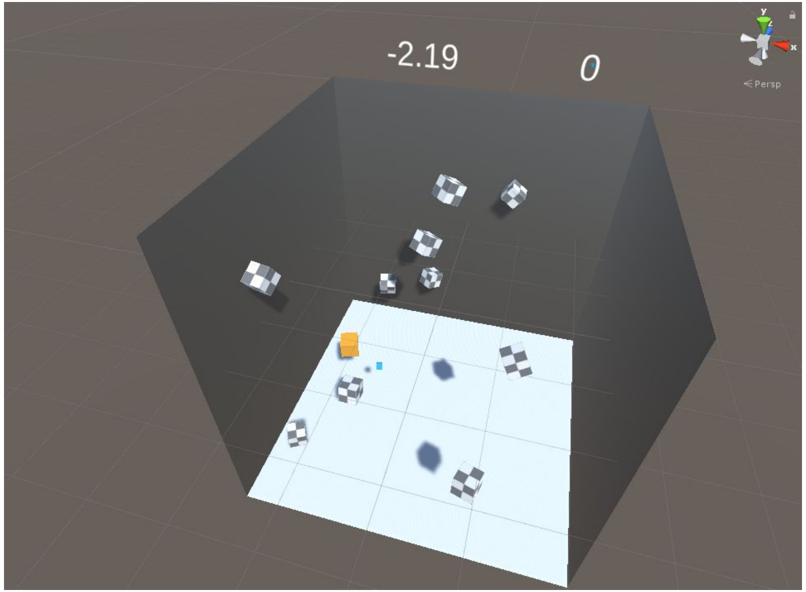
Movement

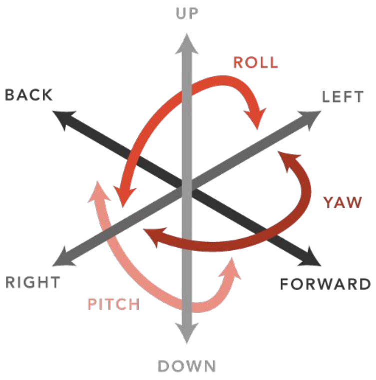
- Simple rover physics **approximation**
- Movement through physically grounded **force application**
- **Instant velocity** change to the agent's body (ForceMode.VelocityChange)
- **Soft clamping** to the max velocity

Actions

- **Decisions** coming from the Brain (model)
- **3D Action Space**:
  - x, z axis **translation** movement
  - yaw axis **rotation** movement

3D Maneuvering System





Movement - Physics system:

- **No gravity**

Actions - Augmented to a **5D** action space:

- x, y, z axis **translation**
- yaw and pitch **rotation**

Learning System

- Uses **Proximal Policy Optimization** as a RL algorithm (Policy Gradient)
- The **reward signal** defines the goal of the task
- **Curriculum learning** scales the difficulty of the task according to the **cumulative reward** reached by the agent
- The same agent's **Brain** is trained on **parallel environments**

Reward Signal - Extrinsic Reward

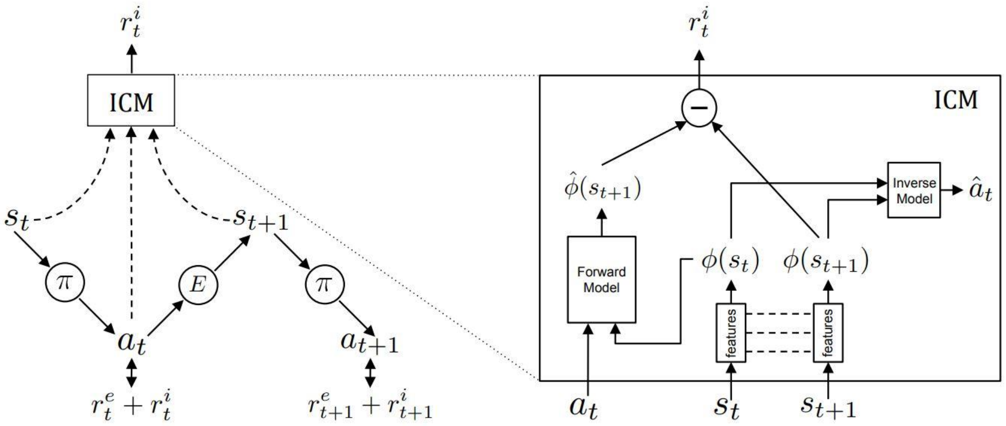
Given in **response** to the **actions** made by the agent in the environment, comprised of two components:

- Reward: *Positive reward - Penalty function*
  - Penalty function: *α*obstacle_collisions+β*time*
  - Positive reward: *γ*target_collisions*

Reward Signal - Intrinsic Reward

Representing the **curiosity** of the agent.

The more **unexpected** the action taken by the agent, the **higher** the curiosity signal.



Alexei A. Efros Trevor Darrell Deepak Pathak, Pulkit Agrawal. "Curiosity-driven exploration by self-supervised prediction" 2017.

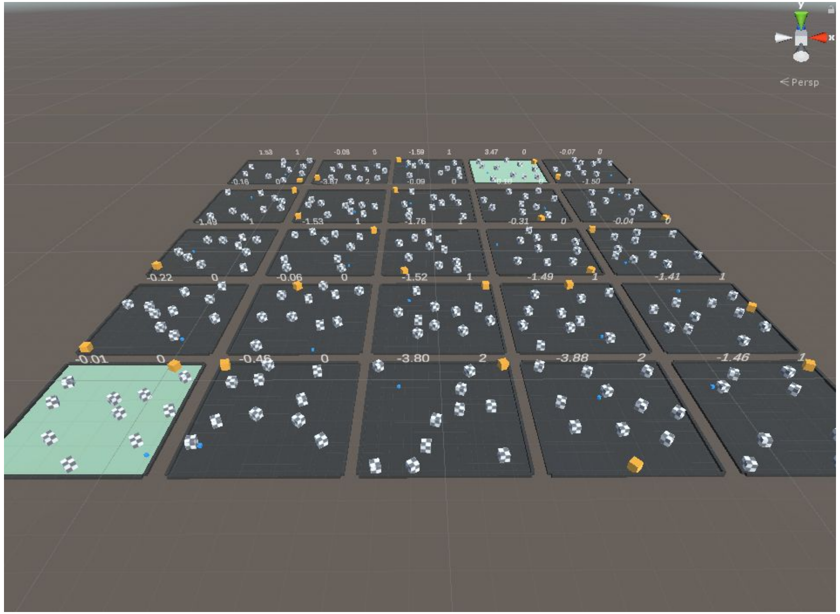
Curriculum Learning

Dynamically change the parameters **during training**, making the task **progressively** harder.

In some cases, allows for **faster policy convergence.**

Curriculum parameters
Reward threshold
Number of obstacles
Minimum object spawn distance
Target-Agent distance
Penalty offset

Parallel Learning



Increases the experience **throughput** of the Agent through **parallel instances** sharing the Brain

Experiments

Performance analysis of different scenarios:

- Baseline
- Curriculum learning
- Harder Penalty function
- Camera sensors
- 3D maneuvering environment

Experiments - Evaluation

Evaluation made through two different types of measurements:

Traditional RL performance metrics

- Cumulative reward
- Policy loss
- ...Others

Environmental performance metrics

- Collisions per minute (CPM)
- Targets per minute (TPM)
- Collisions per target (CPT)

Experiments - Baseline

Fixed parameters:

Number of obstacles	10
Min spawn distance	2
Target distance	45

Penalty function:

$$p = collisions * 0.1 + time * 0.001$$

Observation sensors: LIDAR set

2D maneuvering environment

Experiments - Curriculum

Curriculum parameters:

Reward thresholds	1	2	2.5	2.8	3	3.5	4
Number of obstacles	8	10	13	15	17	18	20
Min spawn distance	6	6	4	4	3	3	2
Target distance	25	28	30	33	35	37	40

Penalty function:

$$p = collisions * 0.1 + time * 0.001$$

Observation sensors: LIDAR set

2D maneuvering environment

Experiments - Harder Penalty

Curriculum parameters:

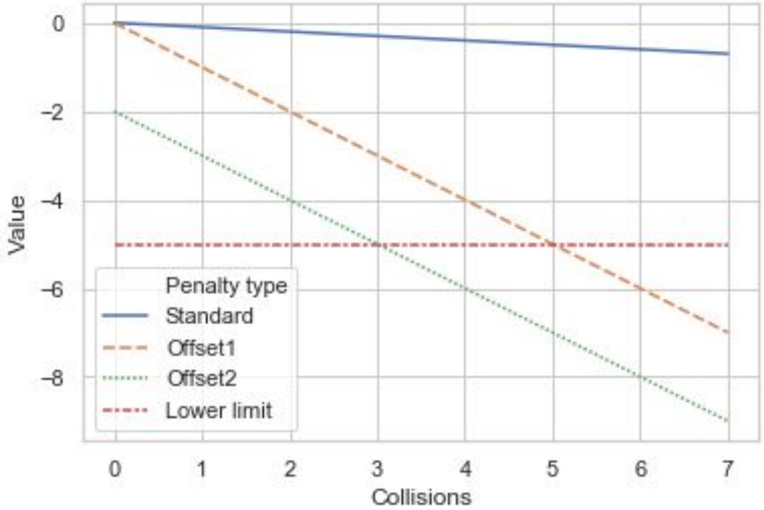
Reward thresholds	1	2	2.5	2.8	3	3.5	4
Number of obstacles	8	10	13	15	17	18	20
Min spawn distance	6	6	4	4	3	3	2
Target distance	25	28	30	33	35	37	40
Penalty offset	0.5	1.5	2	2.5	2.5	2.5	2.5

Penalty function:

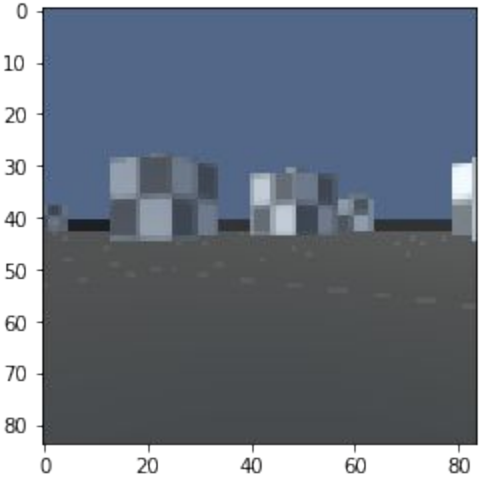
$$p = p_{offset} + collisions + time * 0.001$$

Observation sensors: LIDAR set

2D maneuvering environment



Experiments - Camera Sensors



Curriculum parameters:

Reward thresholds	1	2	2.5	2.8	3	3.5	4
Number of obstacles	8	10	13	15	17	18	20
Min spawn distance	6	6	4	4	3	3	2
Target distance	25	28	30	33	35	37	40

Penalty function:

$$p = collisions * 0.1 + time * 0.001$$

Observation sensor: Camera, 84x84 RGB

2D maneuvering environment

Experiments - 3D maneuvering env.

Curriculum parameters:

Reward thresholds	1	2	2.5	2.8	3	3.5	4
Number of obstacles	8	10	13	15	17	18	20
Min spawn distance	6	6	4	4	3	3	2
Target distance	25	28	30	33	35	37	40

Penalty function:

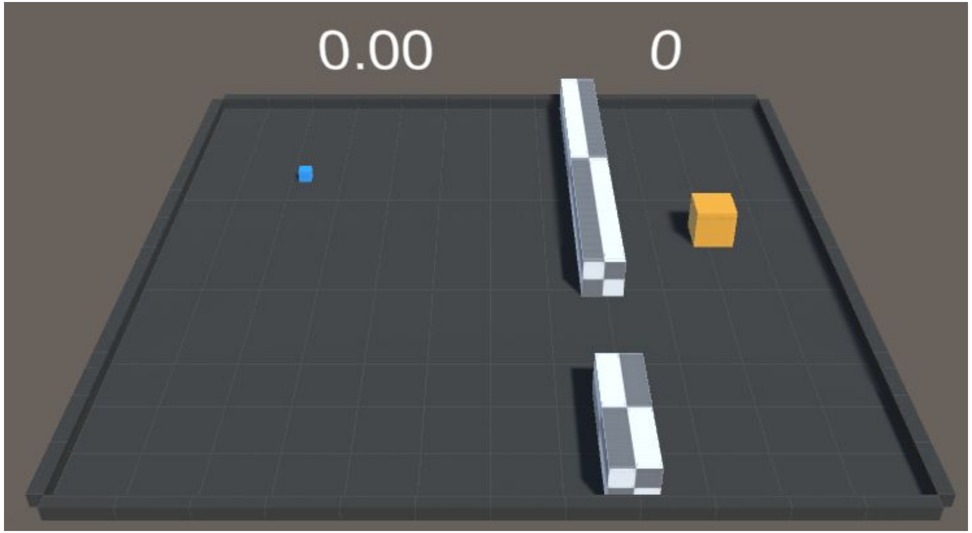
$$p = collisions * 0.1 + time * 0.001$$

Observation sensors: (Extended) LIDAR set

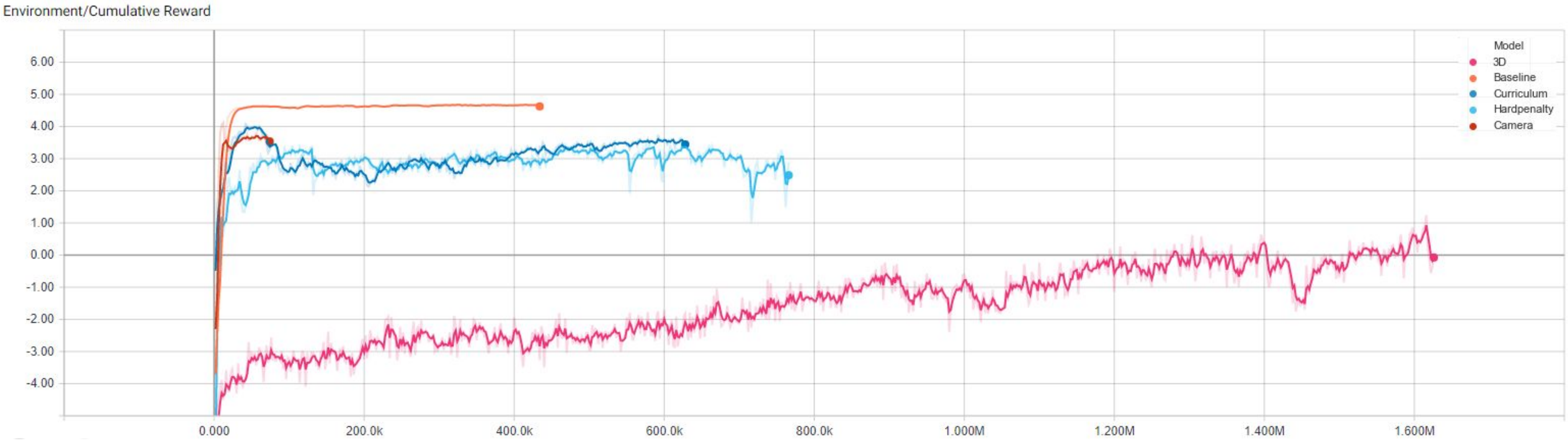
3D maneuvering environment

Experiments - Structured environment transferability

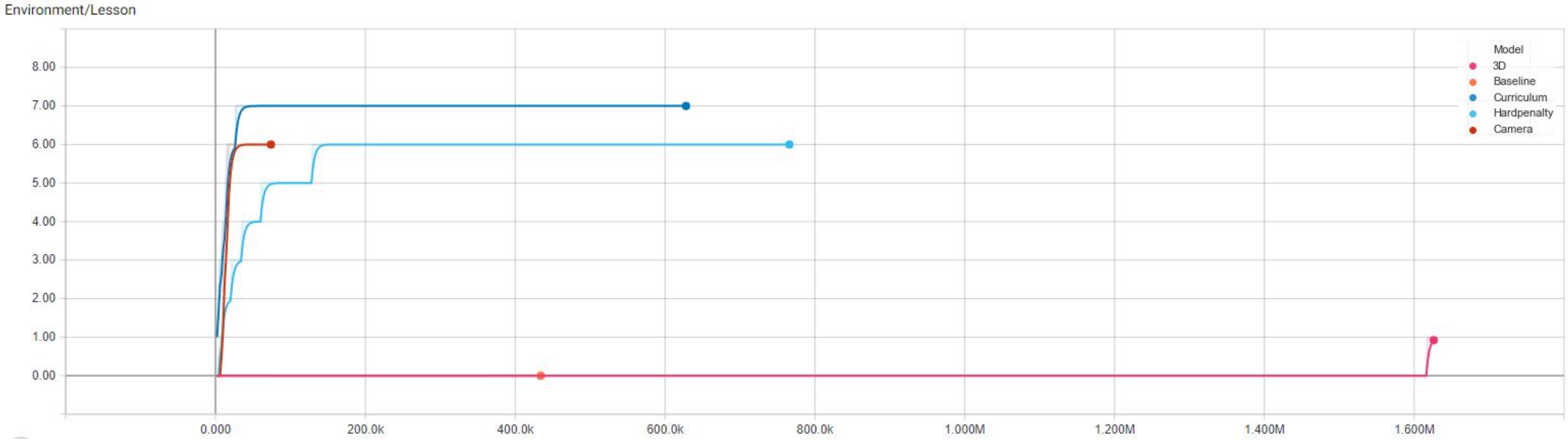
Performance evaluation of the **best** between the aforementioned 2D models in a **structured** environment, performing the same task



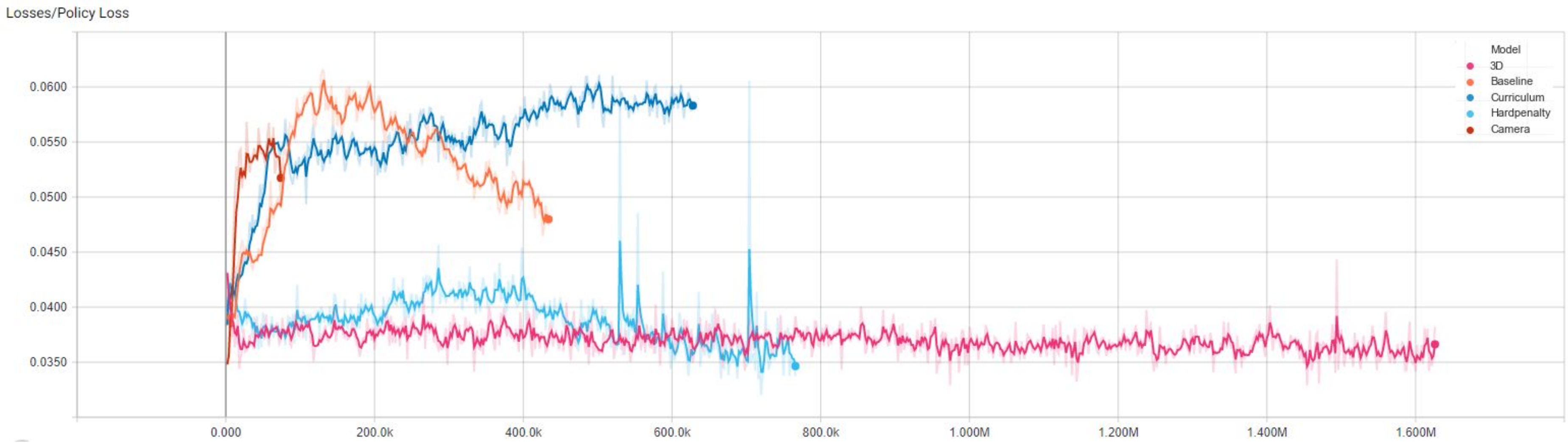
Results - Performance measures



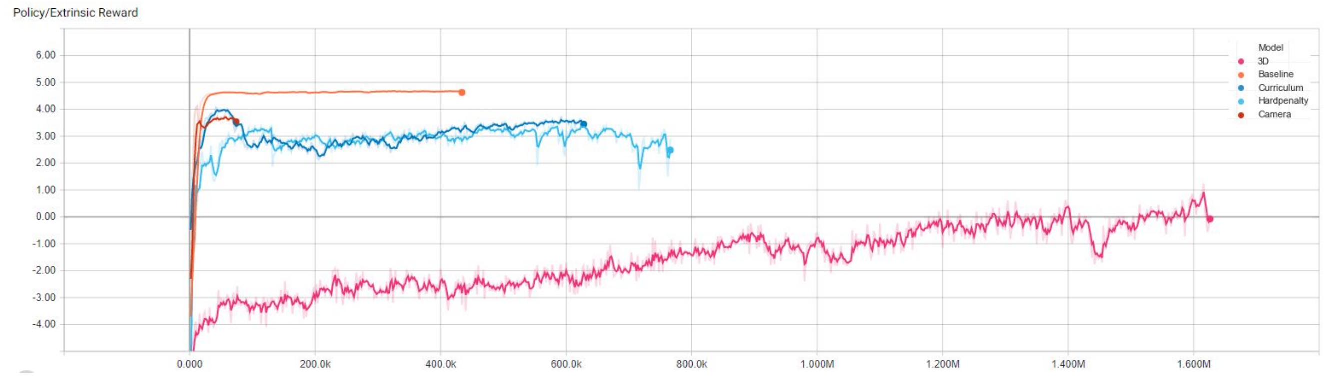
Results - Performance measures

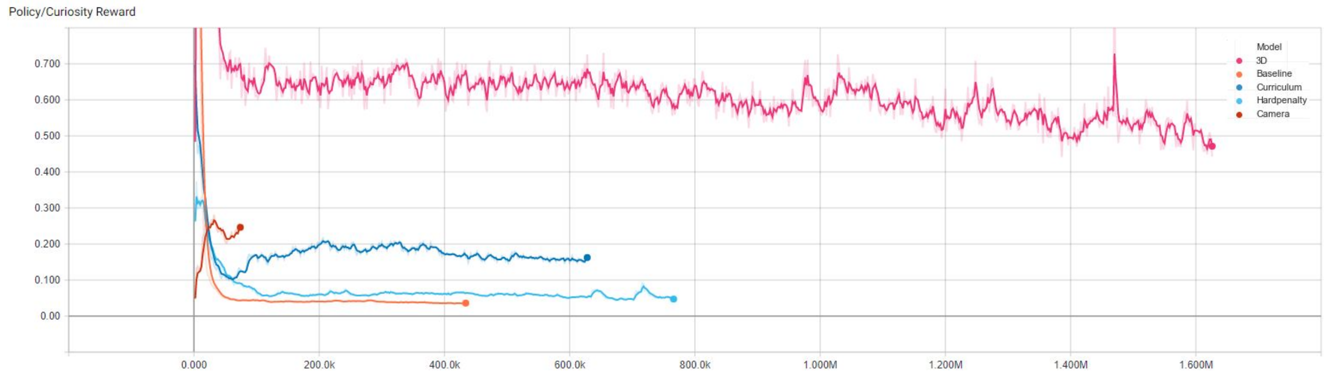


Results - Performance measures



Results - Performance measures





Results - Performance measures

Training time is not equal between experiments due to training time constraints.

The performance measures show **different levels of convergence** of the models and **different advancements in the lessons** of the curriculum that seem to not correlate directly with the training time.

Direct evaluation on these measures is **hard**:

- Which is the best model? The one with a better reward convergence? The one that advanced to the last lesson? The one with a better policy loss convergence? A mix of all them?
- Which measures are significative of the model performance in the experimental scenario?

Results - Environmental performance

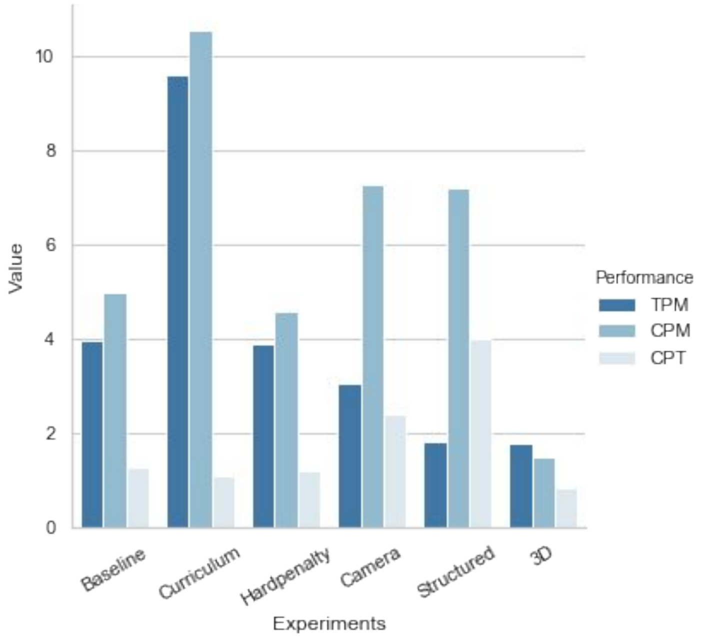
Environmental performance measures allow us to measure the **empirical performance** of the models for the particular task they've been trained for.

Environmental performance is hence a **domain-coupled measure**.

The measures used for this project are TPM, CPM and CPT.

Each model has been evaluated on the **same environment**, having the **same parameters**, to make the comparison fair.

Results - Environmental performance



Best overall: *curriculum learning model.*

Almost every model managed to stay **below 2 CPT.**

The *harder penalty model* **did not improve** the performance.

The *camera model* **did not perform** as well as the LIDAR model, but also **did not train as much.**

The *3D maneuvering model* reached almost 2 TPM and CPM, but did not complete the whole curriculum.

Conclusions

For robotic locomotion and target localization tasks **reinforcement learning** and **curriculum learning** perform effectively.

The results show a different outcome based on the **evaluation method** chosen, underlining the difficulty to evaluate correctly reinforcement learning scenarios.

The models proposed seem to converge on a policy of **random search**, a behaviour shared between every experiment model. Most probably due to the **lack of memory** of the NN and **procedural generation** of the environment.

The models are able to generalize **obstacle avoidance**.

The models, seen the converged policy, manage to perform **target localization** discreetly.

Conclusions - Future works

- **Add memory to the system**: adding an RNN module (akin to the curiosity module) might allow the model to form an experience buffer of sorts and adopt smarter search policies.
- **Rework the reward and penalty functions**: implement more complex/effective functions, (eg soft-collisions)
- **Compare different RL algorithms**
- **Extend the types of scenarios**