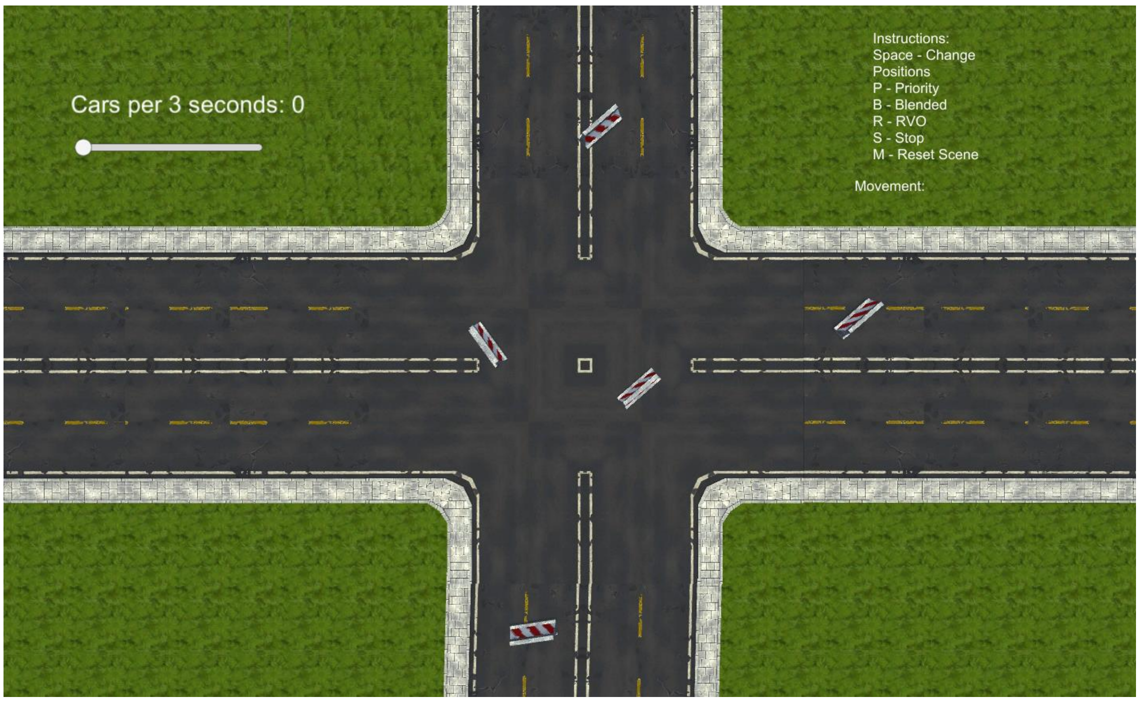
TÉCNICO LISBOA

Inteligência Artificial para Jogos

2020-2021

# First Project of Artificial Intelligence for Games



In the first project of IAJ, the goal is to implement and test several collision avoidance algorithms, including the AvoidCharacter, AvoidObstacle and Reciprocal Velocity Obstacle methods lectured. The project is divided into tasks, which are grouped in levels. In order to facilitate the development of the projects, we recommend you to start working on a new level only after the previous one is finished.

The project must be submitted electronically in the form of a zip file until 23:59 of Tuesday October 13th 2020. Any project submitted after the deadline will receive 0.5 values of penalization for each hour of delay. We also recommend trying to finish the work corresponding to the two initial laboratory classes before starting to implement this project.

## Level 1 – AvoidCharacter

- Implement the AvoidCharacter DynamicMovement, and apply it to both the Priority and Blending Movement.

## Level 2 – AvoidObstacle

- Implement the AvoidObstacle Movement in order to avoid collisions with static objects (the road limits and obstacles from lab2). Use Unity's collision detection mechanism. Select the ray configuration that provides the best results. Integrate the movement in both the Priority and Blending Movement. Select appropriate weights so that resulting behavior appears reasonable.

## Level 3 – Reciprocal Velocity Obstacle

- Implement the RVO Movement. In this stage you should ignore collisions with obstacles, and focus only in collisions with other moving characters. Adjust the movement's parameters in order to make the resulting behavior realistic.

## Level 4 – RVO with Static Obstacle Avoidance

- Extend the RVO movement algorithm so that it also considers static obstacles. You cannot assume that all static obstacles will correspond to spheres. How to deal with linear obstacles?

## Level 5 – Efficiency

- Use Unity's profiling mechanism to analyze the performance of your code.
- Taking into account the feedback from Unity's profiling, identify an aspect of the source code worth improving. Annex to the submitted code a picture with a print screen of the profiling analysis to justify your decision.
- Improve the efficiency of your code, by improving the aspect identified in the previous task. Run the profiling mechanism again and compare the new results with the previous results.
- Finally, compare the performance of the different collision avoidance algorithms (RVO vs the other avoidance algorithms).

## Bonus Level – Actuators

- After the movement algorithms implement the output filtering algorithm or Heuristic-based rules for the cars' movement.