



FAKULTÄT FÜR
INFORMATIK

NI controller
Team Emergence,
Julian Blank, Frederick Sander

Overview

- Evolutionary Algorithm
- Sliding Window
- Pessimistic Iteration
- Adaptive Pathlength
- Heuristic and Reward
- Heuristic Switch
- Gamedetection

Evolutionary Algorithm

- Initial pool with a list of just random actions → path
- Crossover: For 50% action of the first and for 50% the action of the second
- Mutation: Crossover with a random path
- Example:

```
--> [score:6.0 , portal:0.5897445701565585, npc:0.8717954414118179, length:5] ACTION_USE,ACTION_UP  
[score:6.0 , portal:0.589744262302434, npc:0.8461544870307561, length:5] ACTION_RIGHT,ACTION_USE,A  
[score:6.0 , portal:0.5897444385748493, npc:0.8717950862006526, length:5] ACTION_RIGHT,ACTION_RIGH  
[score:4.0 , portal:0.5384615976997722, npc:0.8461545467433345, length:5] ACTION_DOWN,ACTION_UP,AC  
[score:4.0 , portal:0.5641025990757296, npc:0.8717950119977794, length:5] ACTION_DOWN,ACTION_UP,AC  
[score:0.0 , portal:0.5384619104396772, npc:0.846153904799866, length:5] ACTION_DOWN,ACTION_RIGHT,
```

Sliding Window

- Do not throw away the population from the last game tick.
- Delete the first action and add one to the end.
- Reset the generation counter and remove the score of all pathes.

Pessimistic Iteration

- Check always the next action n times
- If the agent dies → search for another path by selecting the next best of the pool with another first action → do it again until the agent stays alive or the time is over

Adaptive Pathlength

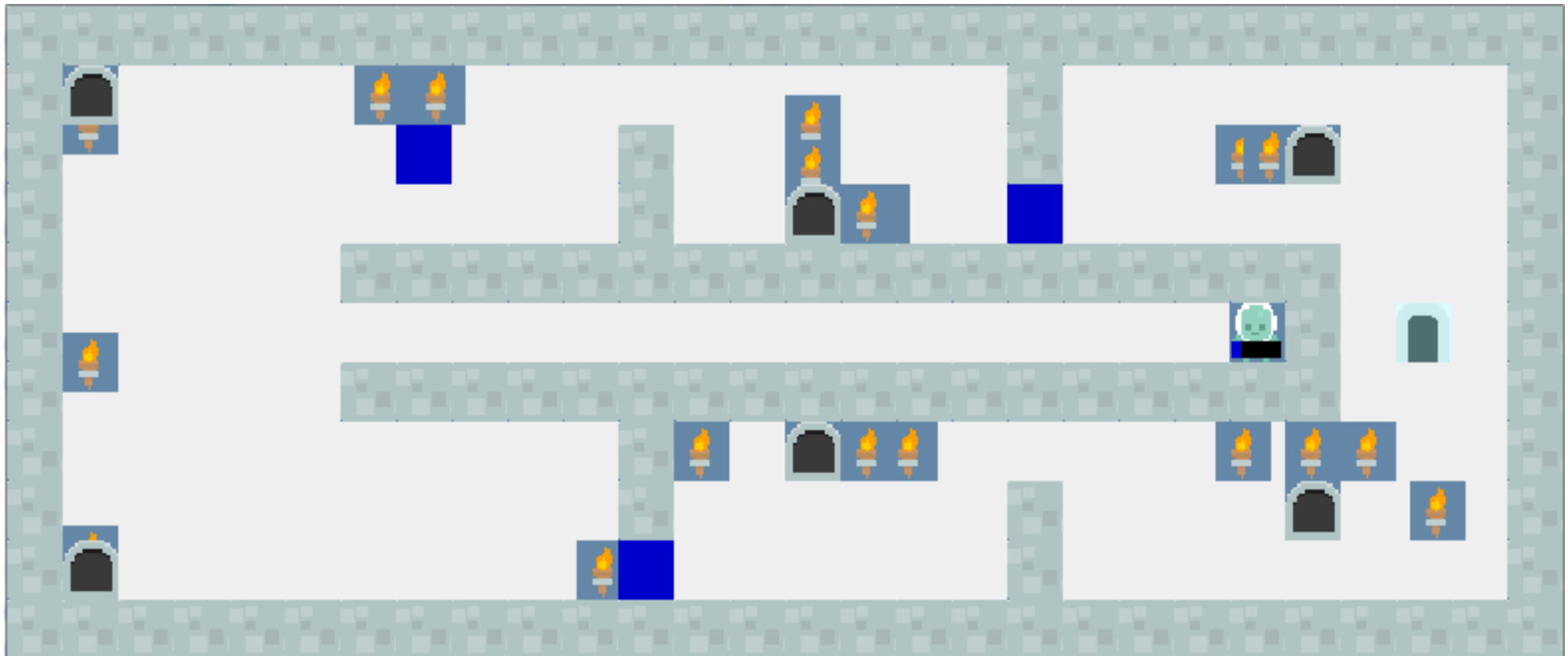
- Problem: Advance method has no static time duration
- An Evolutionary Algorithm makes no sense if we stay always in the first generation
- Adapt the path length of each entry of the pool to complete always the fourth generation → this is combined with the sliding window approach.

Heuristic and Reward

- Different ways to evaluate a path:
 - 1. Reward
 - Depends on gamescore and win/lose
 - 2. Heuristic Value
 - Is generated for every state in the path
 - Different Targets:
 - NPC
 - 2x Portal
 - ect...
- Compare two paths:
 - When $\text{Reward} == 0 \rightarrow$ use the heuristic value

Heuristic Switch

- Sometimes one game needs different heuristics
- approaches:
 - switch heuristic after a defined number of timesteps
 - switch heuristic randomly



Gamedetection

- Detect the game which is played:
 - generate String of Objects in the game (npc, portal,...) → store hash value
 - Constructor: put known hash values in hash set
 - at the beginning of every game → check which game is played
 - set settings depending on the game (pathlength, heuristic, ...)
- Improved performance in the 20 known games
- Did not decrease performance in the test set
 - no game is detected → standardsettings

References

- First
- Second
-