Solving Sparse-reward Problems in Partially Observable 3D Environments using Reinforcement Learning

Cobus Louw

Supervisor: Prof HA Engelbrecht

Co-supervisor: Mr JC Schoeman

Mobile Robots

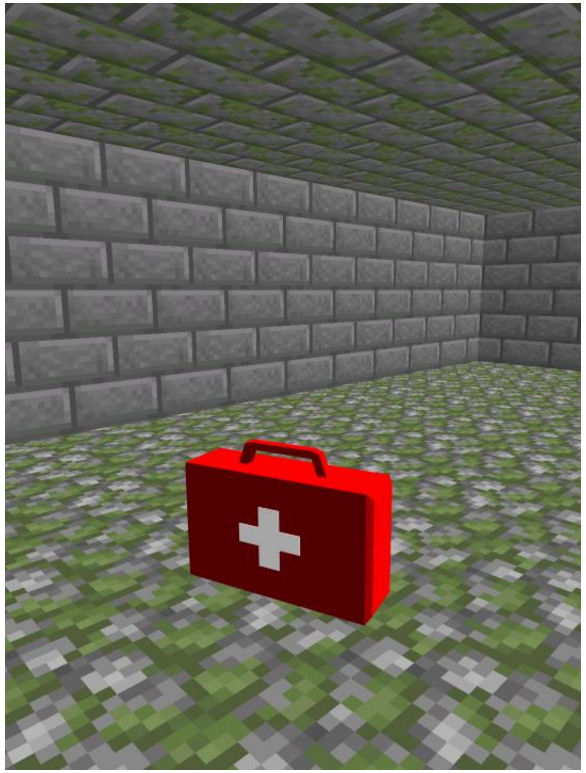
- Mobile robots are widely used
  - In areas that are difficult to access
  - In dangerous situations
- Usually controlled by a human operator
- What happens when RF signals cannot reach the robot?
- For example: a collapsed mine where assistance is needed
- The robot must be capable of making the decisions of the human controller

Problem Statement

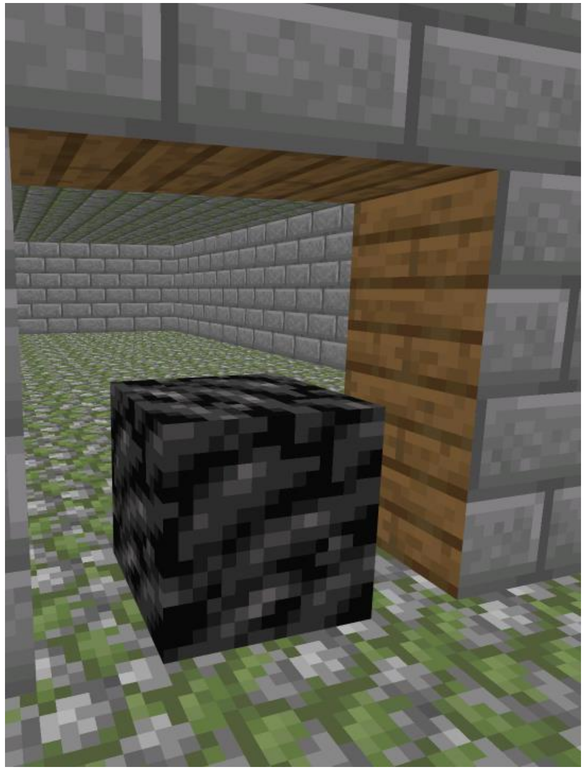
- Collapsed mine
- Injured miner needs resources/assistance
- Autonomous robot has to deliver supplies
- First-person RGB observation (76x44x3)
- MiniWorld – 3D simulation environment in Python
- Environment action space

Action space	Unit
Move Forward	0,5 units
Move Back	0,5 units
Turn left	-9°
Turn right	9°
Pick up / drop	-
No operation	-

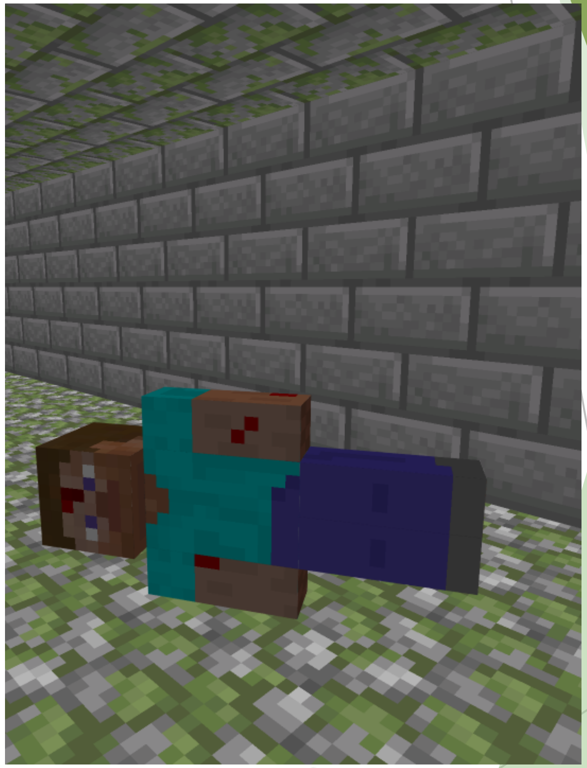
Simulation Environment



First aid kit



Obstacle



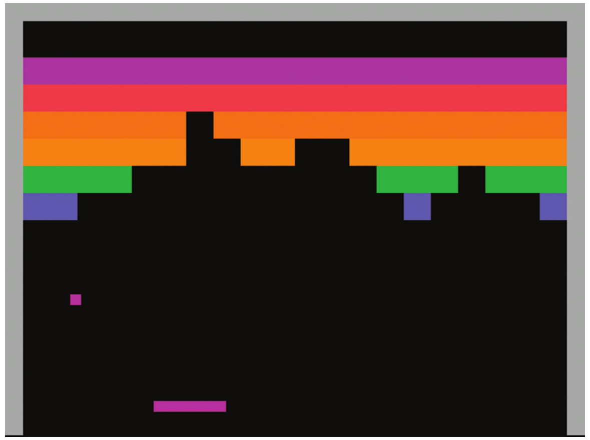
Injured miner

- Find and deliver first aid kit to miner

Possible Solutions

- Behavioural Cloning
  - Clone behaviour of expert
  - For example using supervised learning
  - Dependent on quality of data-set
  - Rarely becomes as good as expert
- Reinforcement Learning
  - No data-set required
  - Learn by trial and error
  - Can become better than expert demonstrator

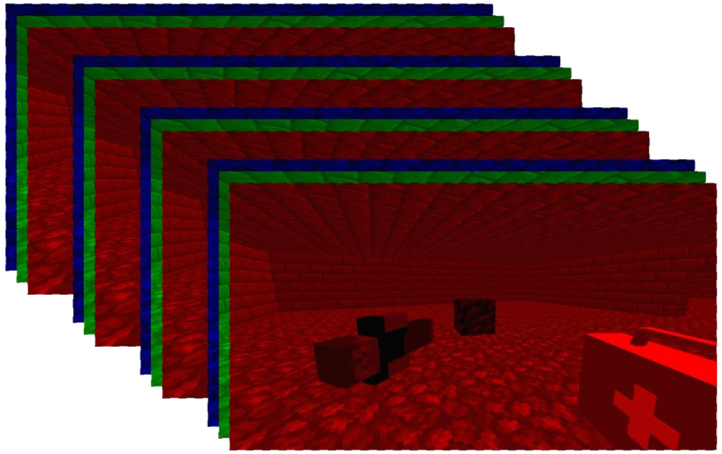
Deep Q-learning (by DeepMind)



- Combines Q-learning with deep neural networks
- Difficult to combine reinforcement learning with deep learning
  - Correlated data
  - Moving targets
- Two important modifications to stabilise training
  - Experience Replay
  - Frozen Q-targets

Partially Observability

- First-person camera
- Partially observable observations
- Function approximation helps according to Sutton and Barto
- Augment observation
  - Frame-stacking
  - Action memory

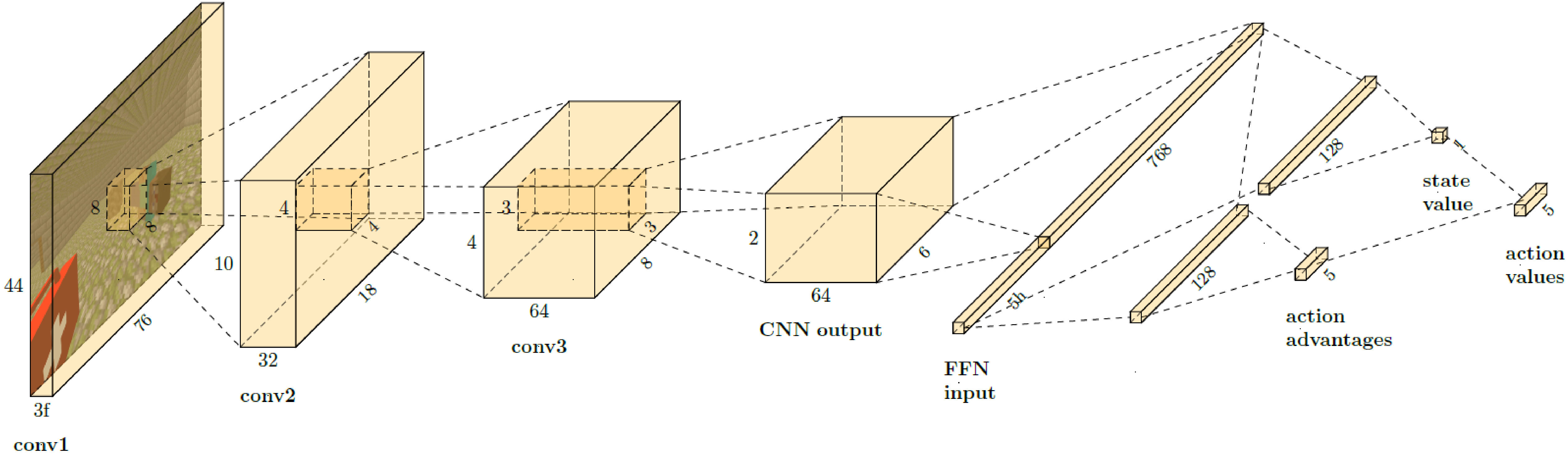


Frame-stacking

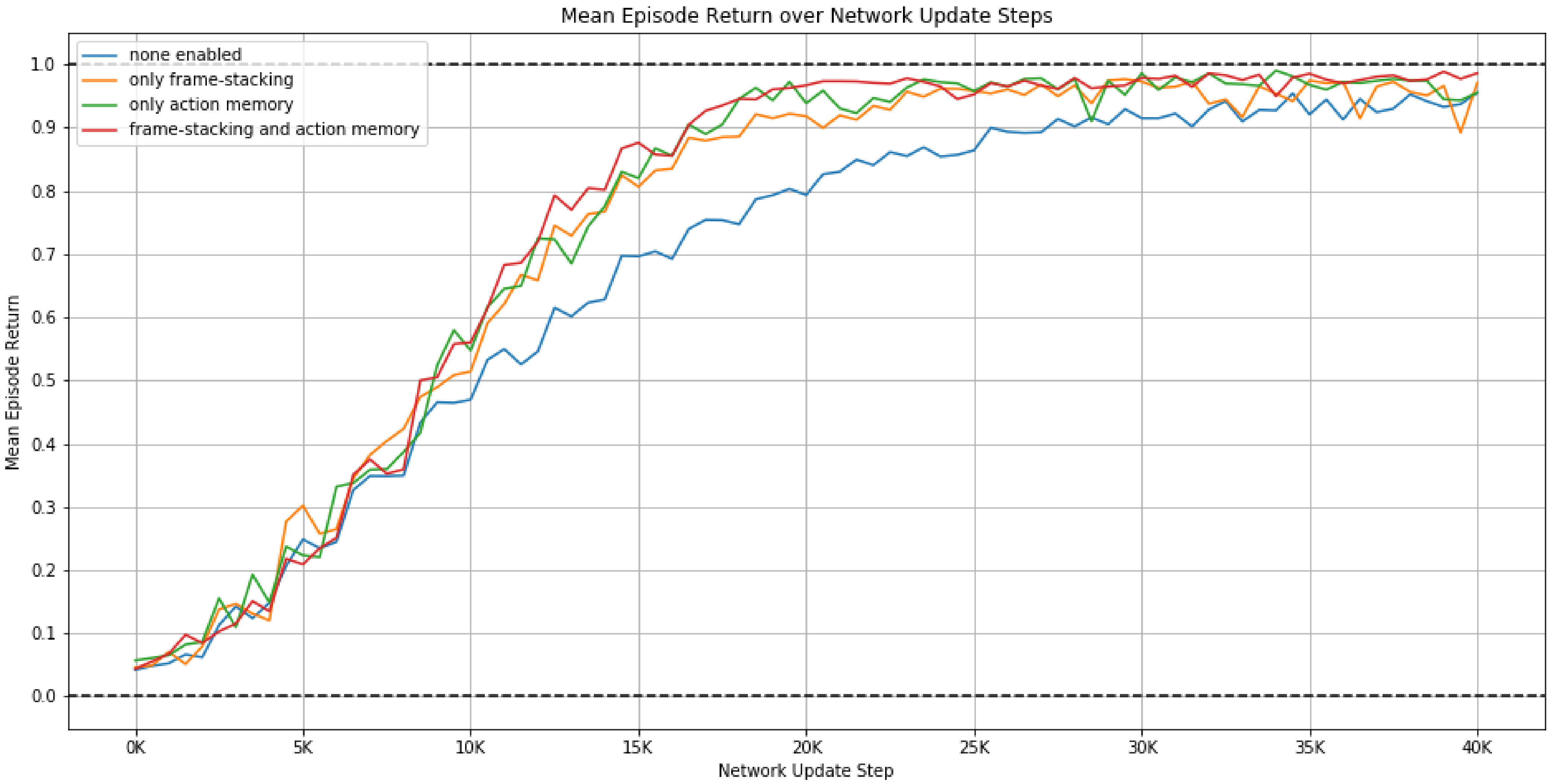
A_{t-1}	A_{t-2}	A_{t-3}	…	A_{t-h}

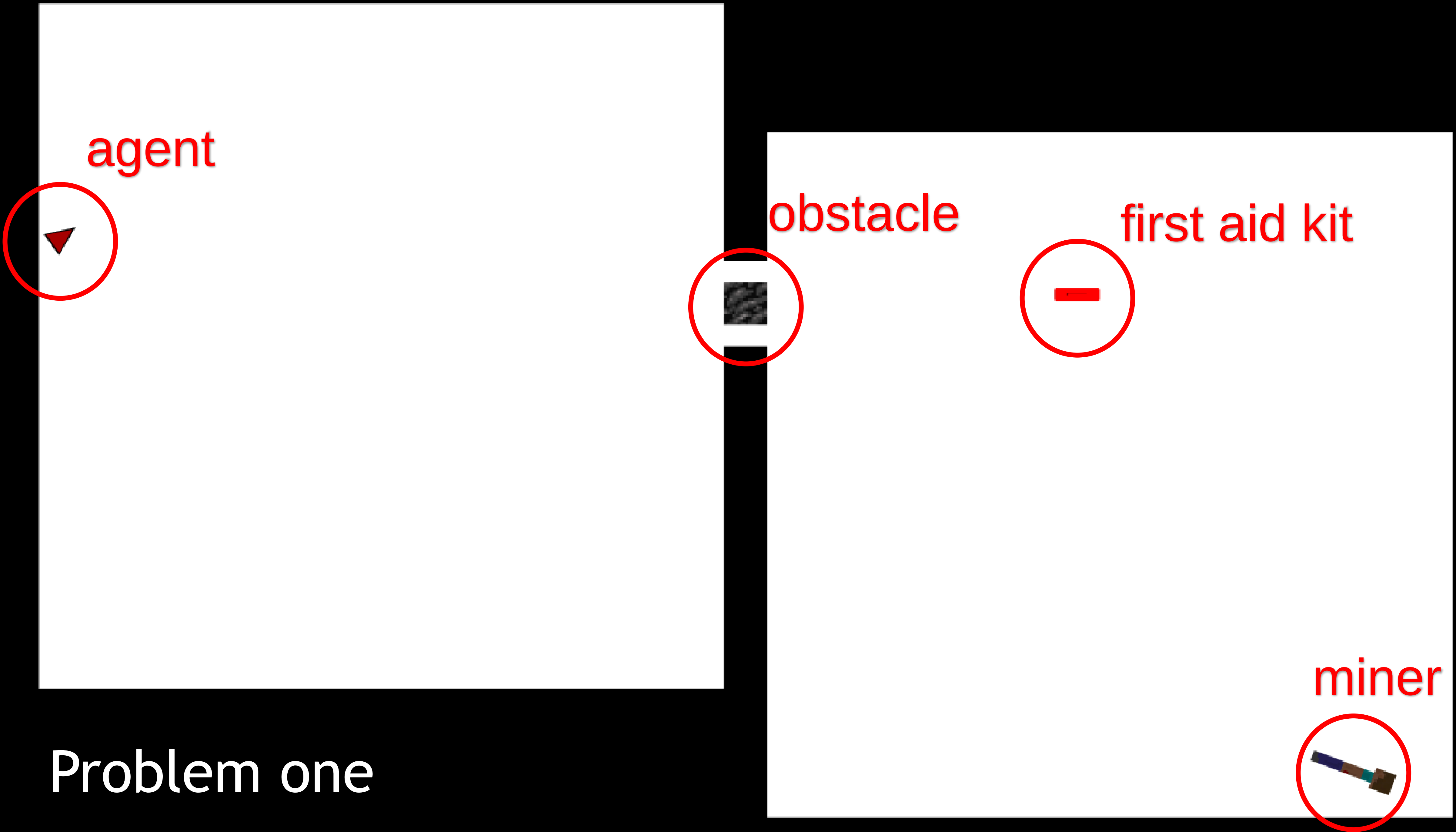
Action memory

Deep Neural Network Architecture (Dueling)

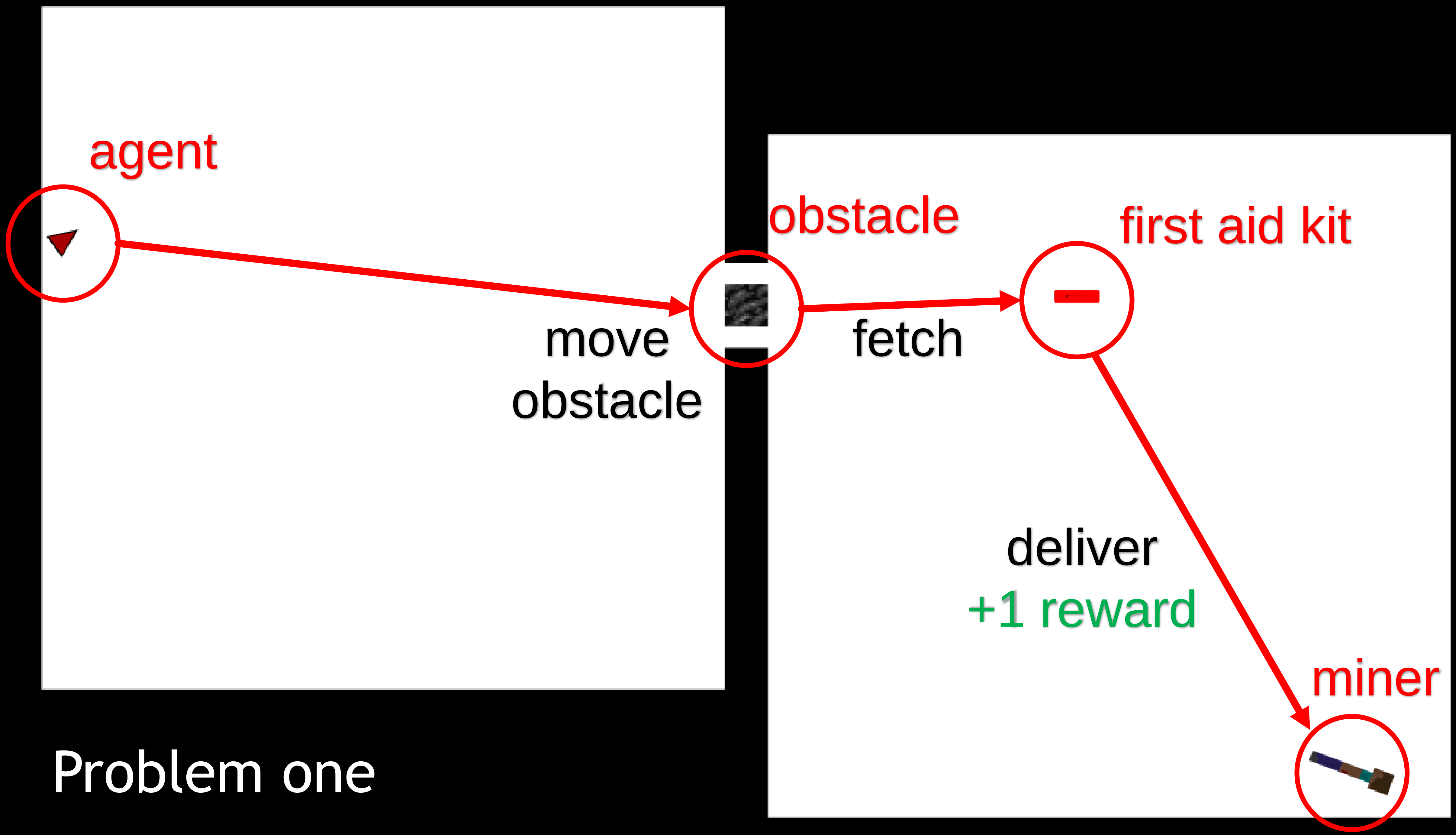


Result
Partially Observability





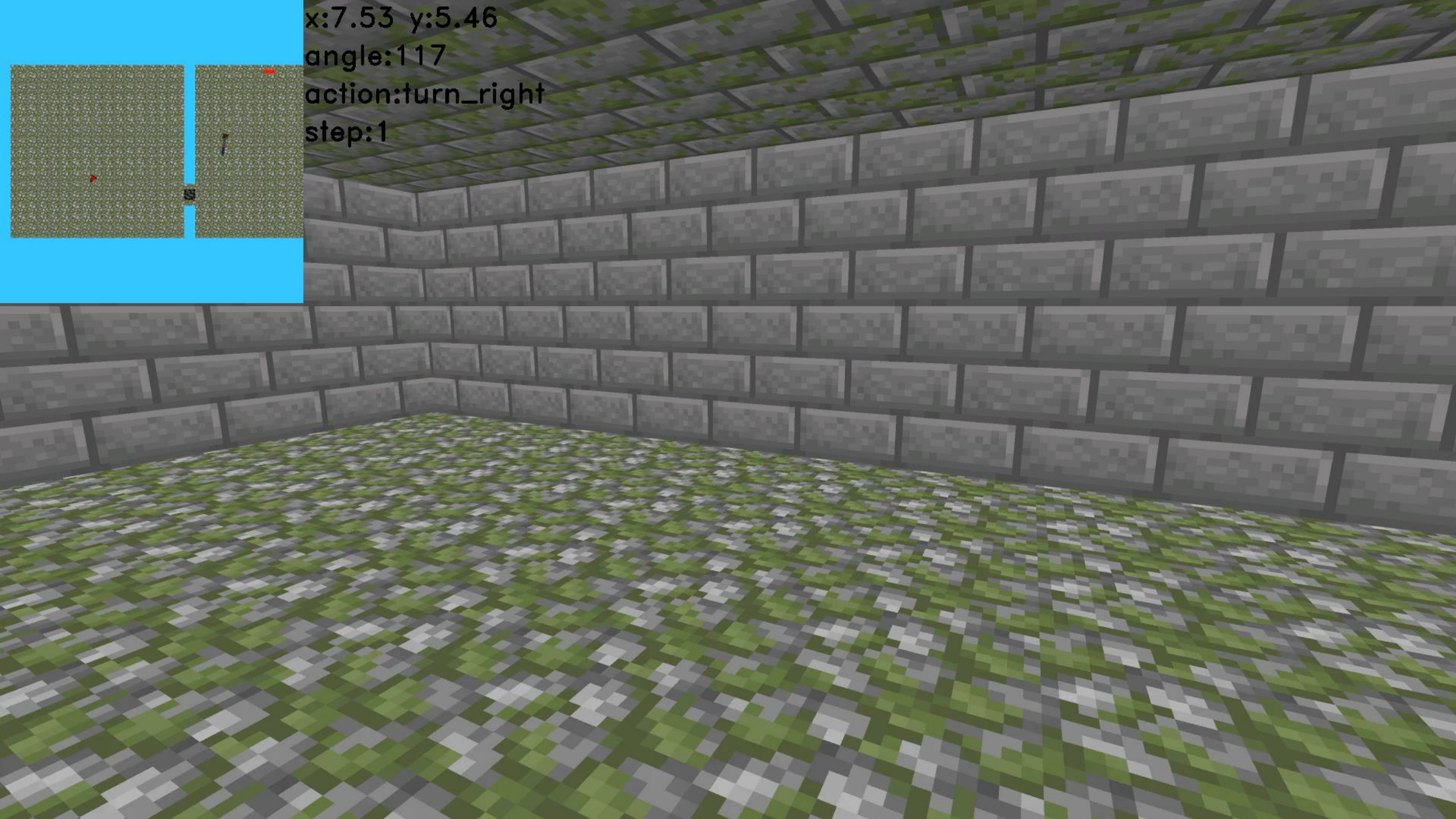
Problem one


agent
obstacle
first aid kit
move obstacle
fetch
deliver
+1 reward
miner

Problem one

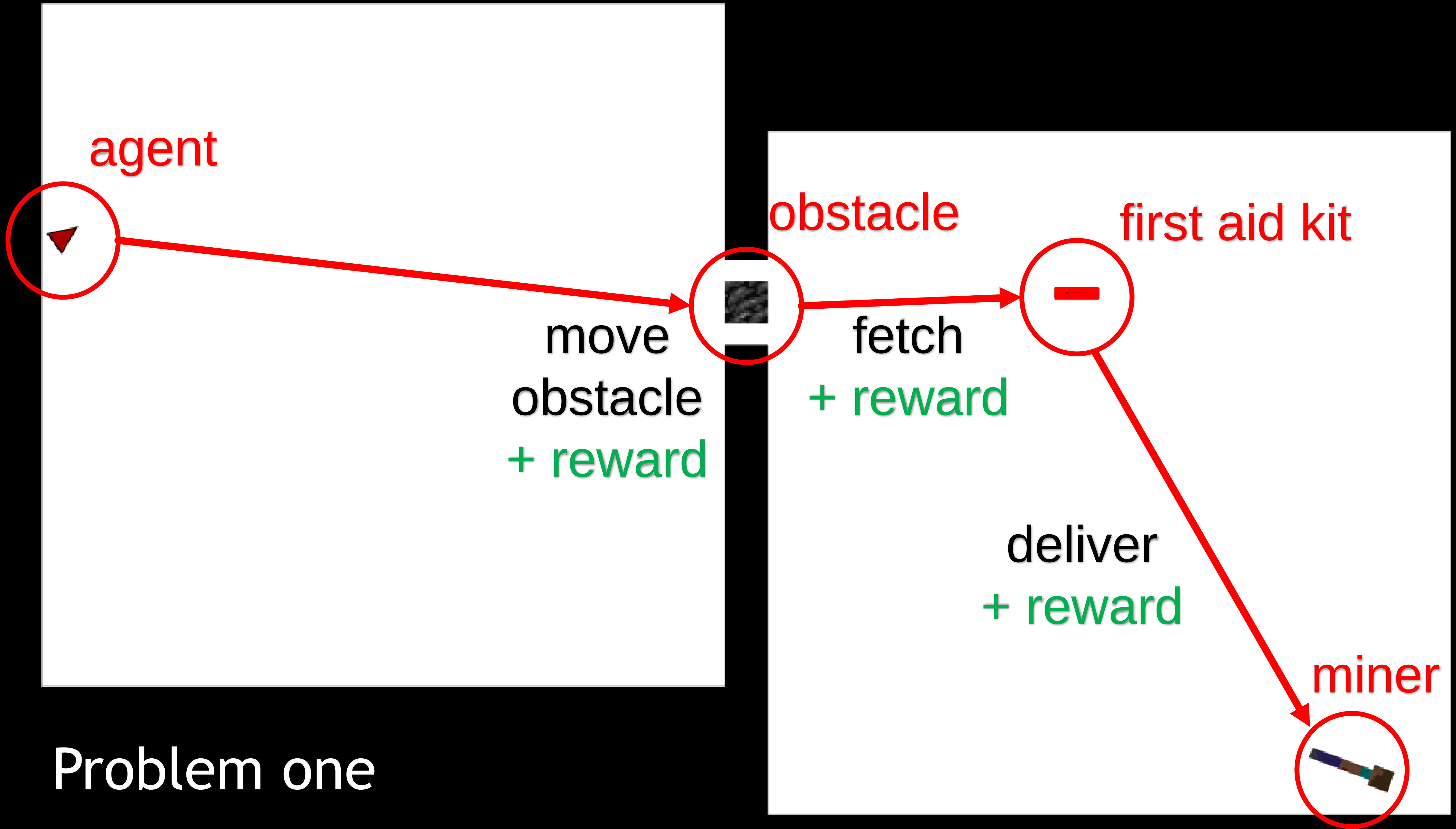
Sparse Reward Problem

- Agent only receives reward when full task is completed
- Problem
  - A newly initialised agent is clueless
  - Epsilon-greedy exploration / random exploration


x:7.53 y:5.46
angle:117
action:turn_right
step:1

Sparse Reward Problem

- Took roughly a million interactions to receive single reward
- Possible solutions:
  - Reward the agent more frequently

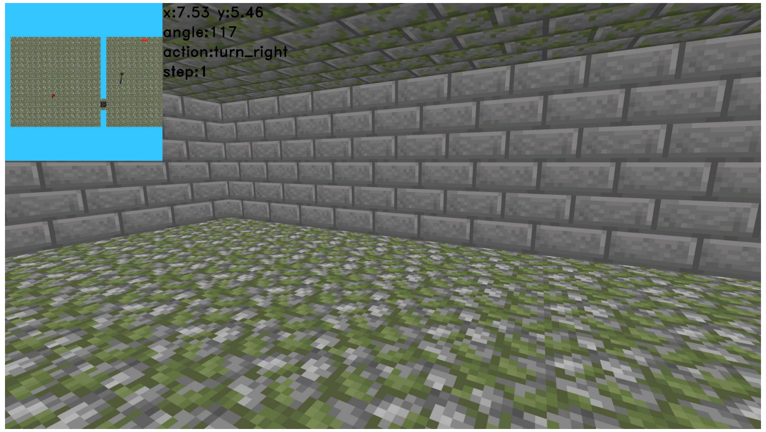

agent
obstacle
first aid kit
move
obstacle
+ reward
fetch
+ reward
deliver
+ reward
miner

Problem one

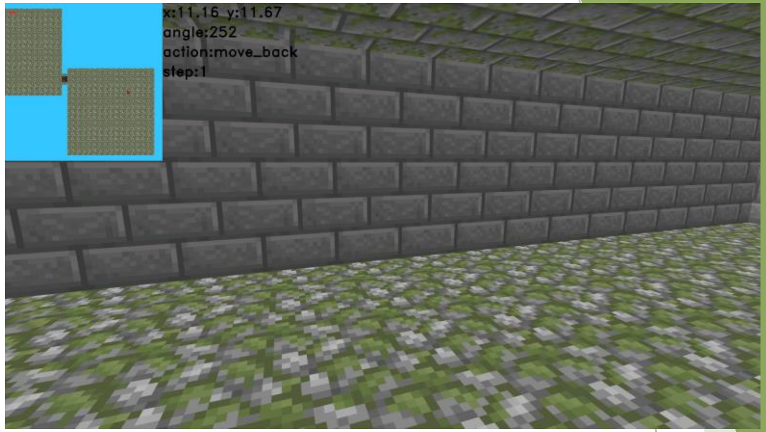
Sparse Reward Problem

- Took roughly a million interactions to receive single reward
- Possible solutions:
  - Reward the agent more frequently
  - Use demonstration data to pretrain agent
  - Generate more data

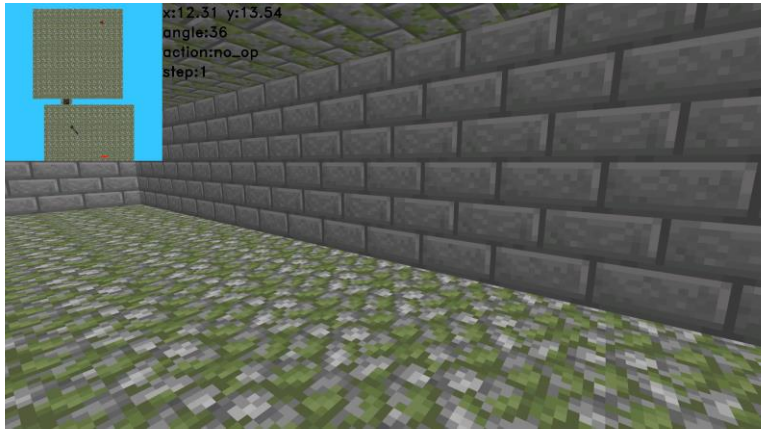
Actor 1
Thread 1



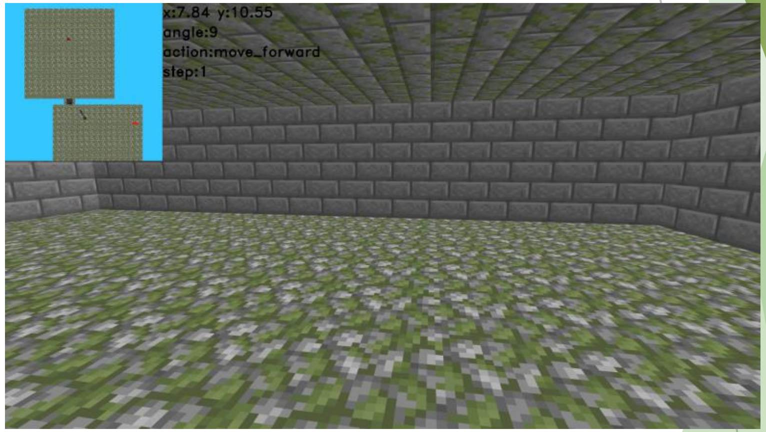
Actor 2
Thread 2



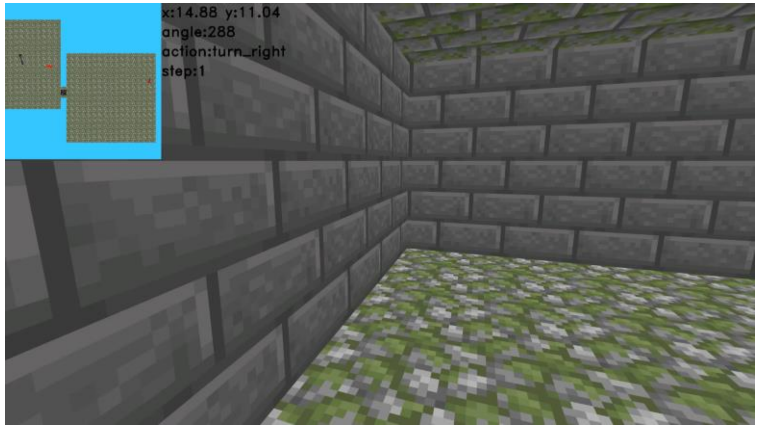
Actor 3
Thread 3



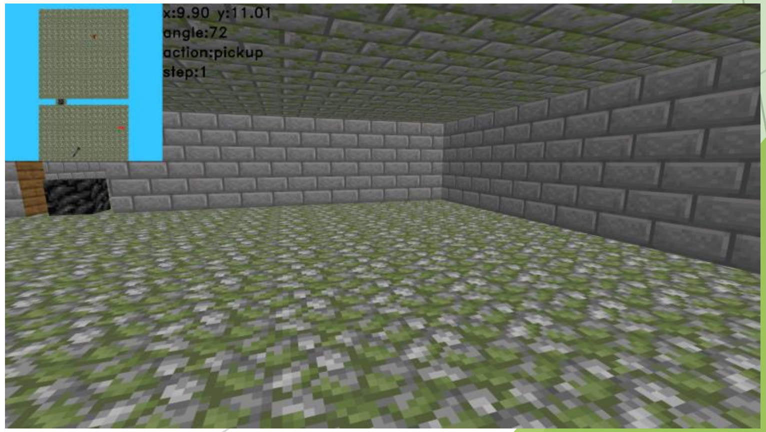
Actor 4
Thread 4



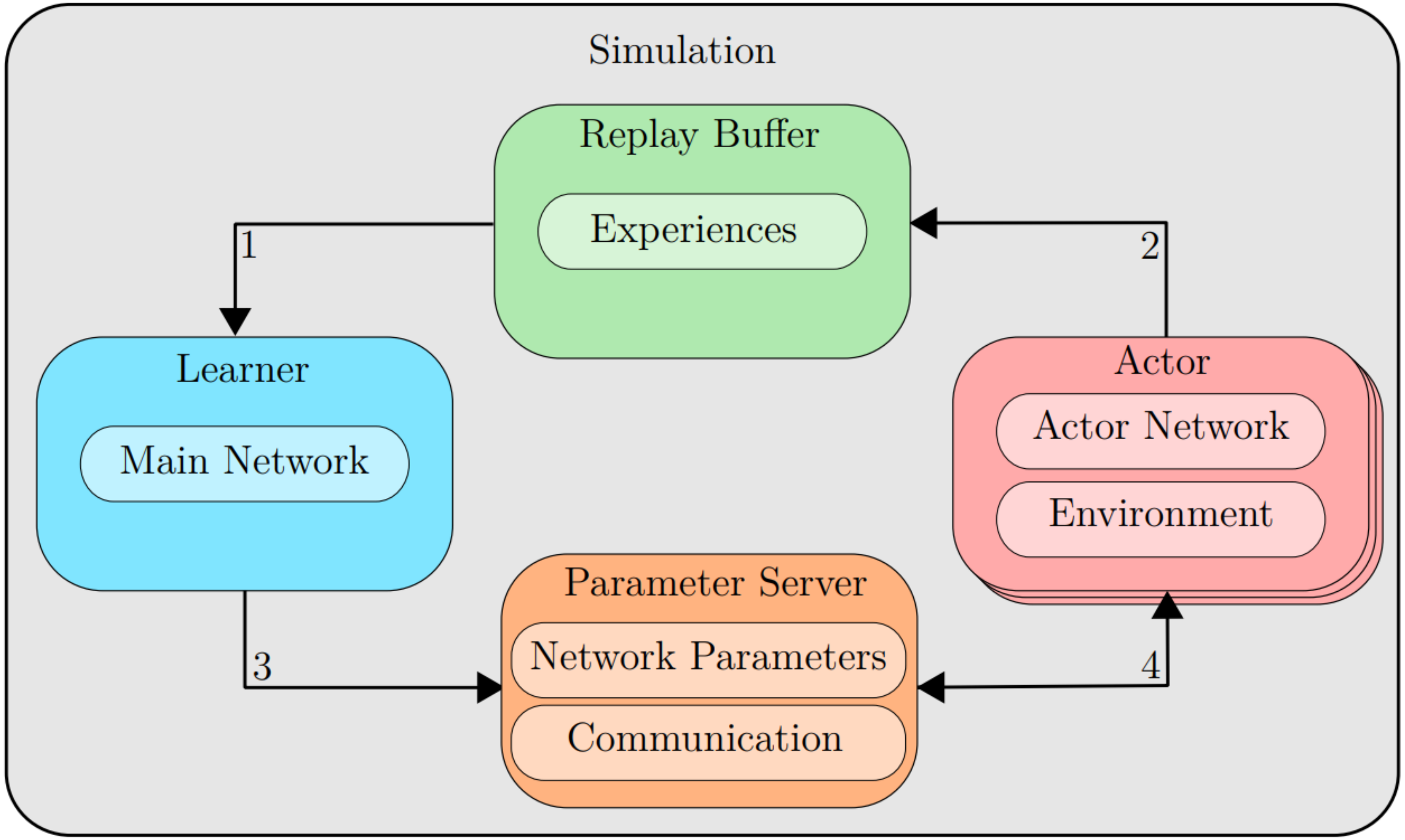
Actor 5
Thread 5



Actor 6
Thread 6



A-peX

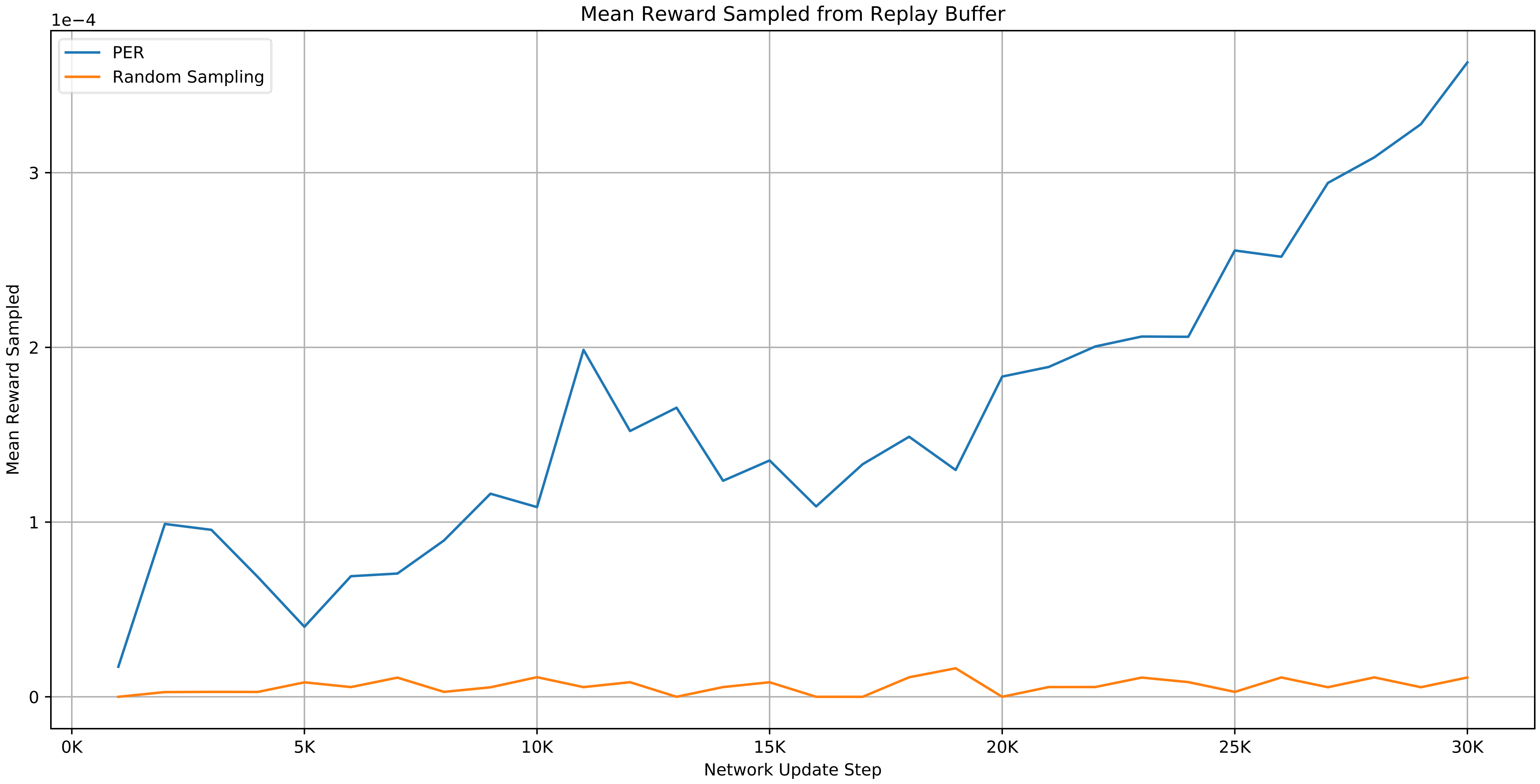


Simulation
Replay Buffer
Experiences
1
2
Learner
Main Network
Actor
Actor Network
Environment
Parameter Server
Network Parameters
Communication
3
4

Prioritised Experience Replay

- Important transitions are still in the minority
- Sampling at random will rarely sample them
- Need to prioritise important transitions
- Prioritised experience replay
- Priorities are based on prediction errors

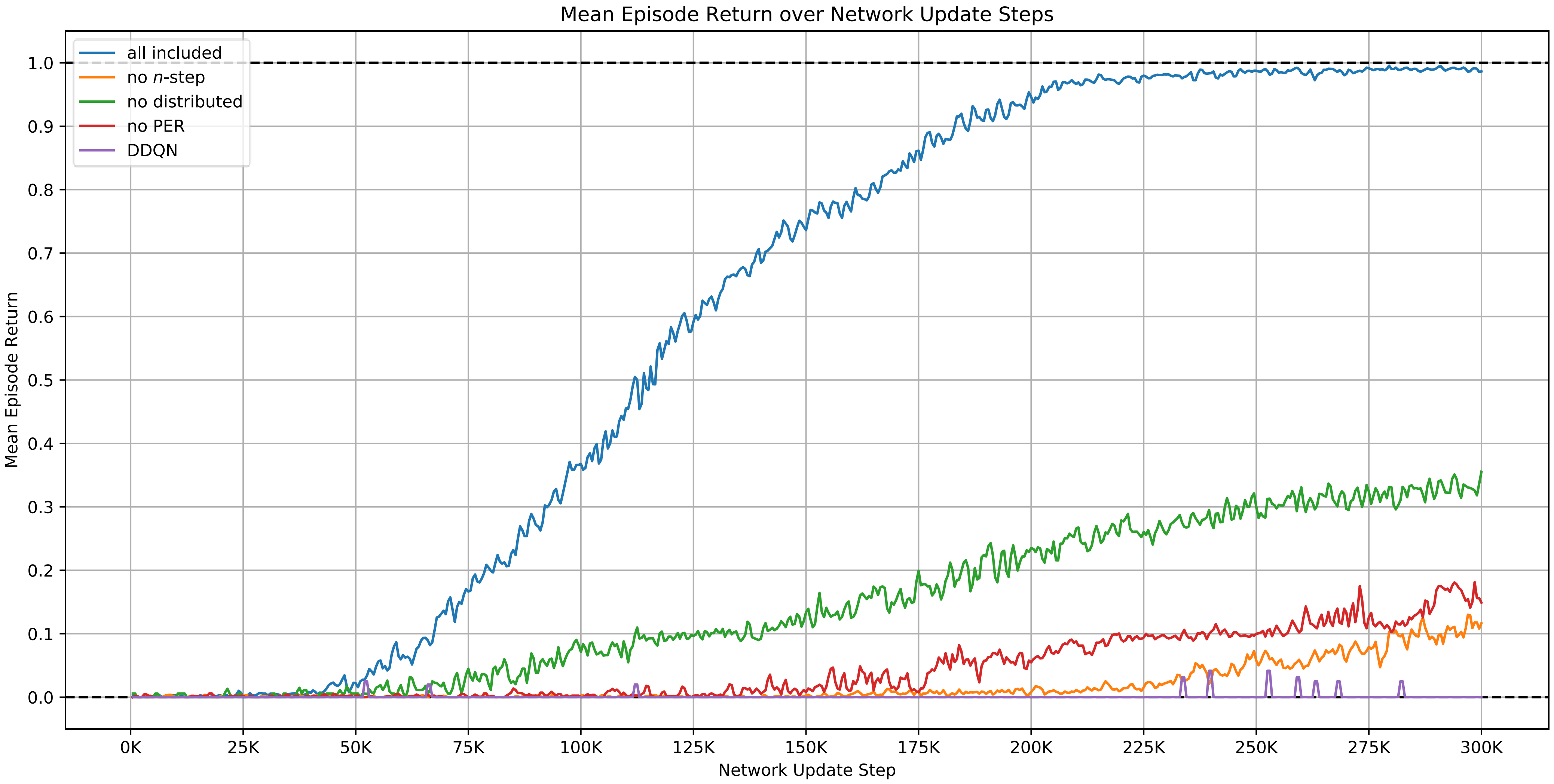
Prioritised Sampling

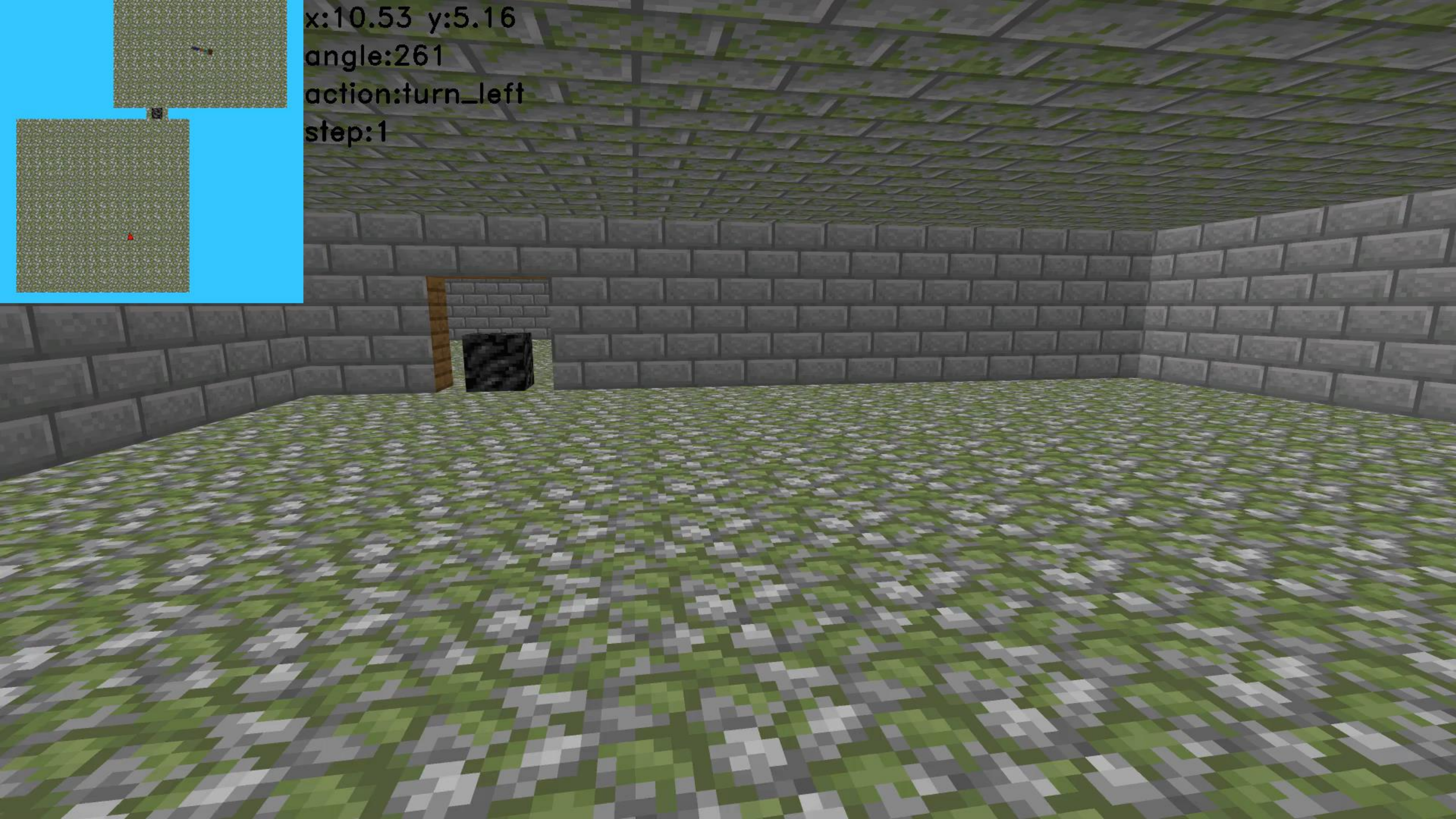


Delayed Rewards and Credit assignment

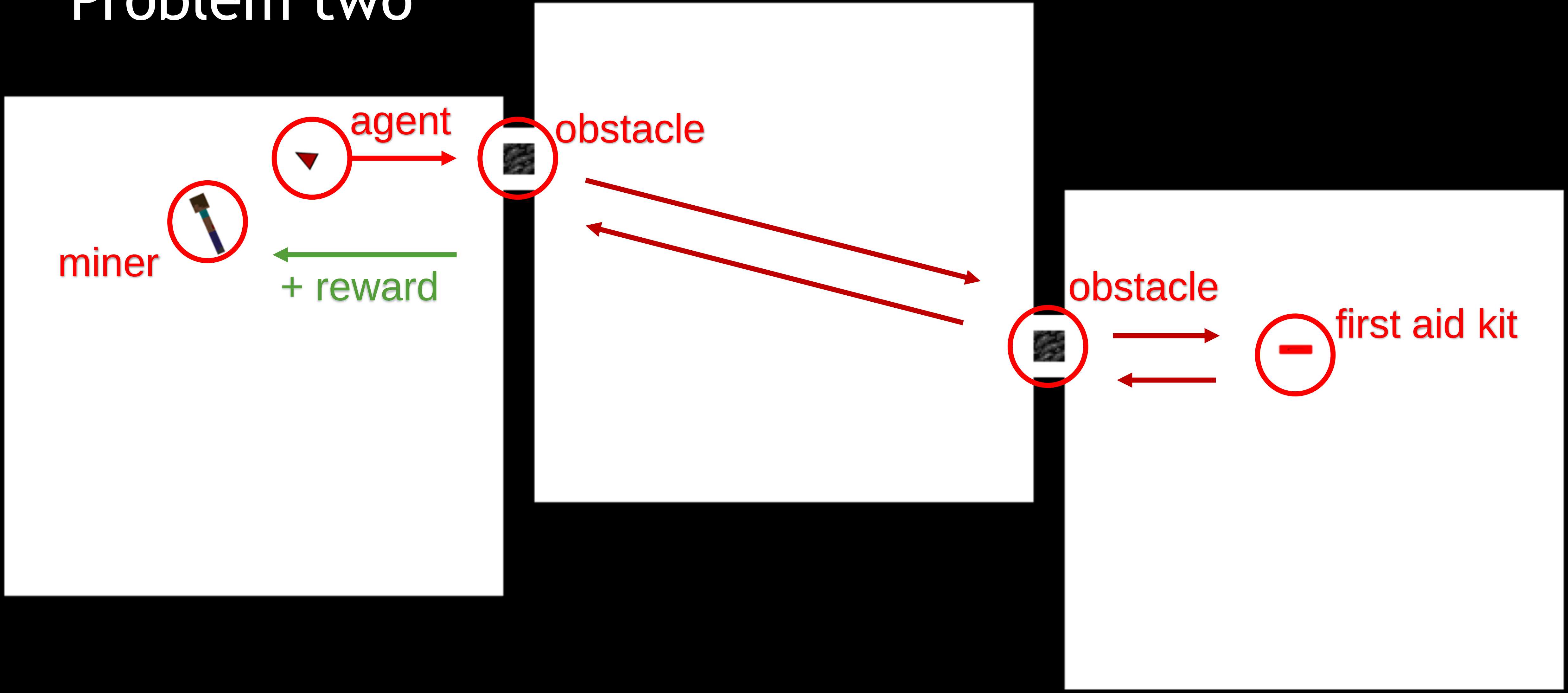
- Multiple sub-tasks without credit
- Eligibility traces
  - Cannot be combined with deep Q-learning
- N-step update

Problem one
Ablation Result




x:10.53 y:5.16
angle:261
action:turn_left
step:1

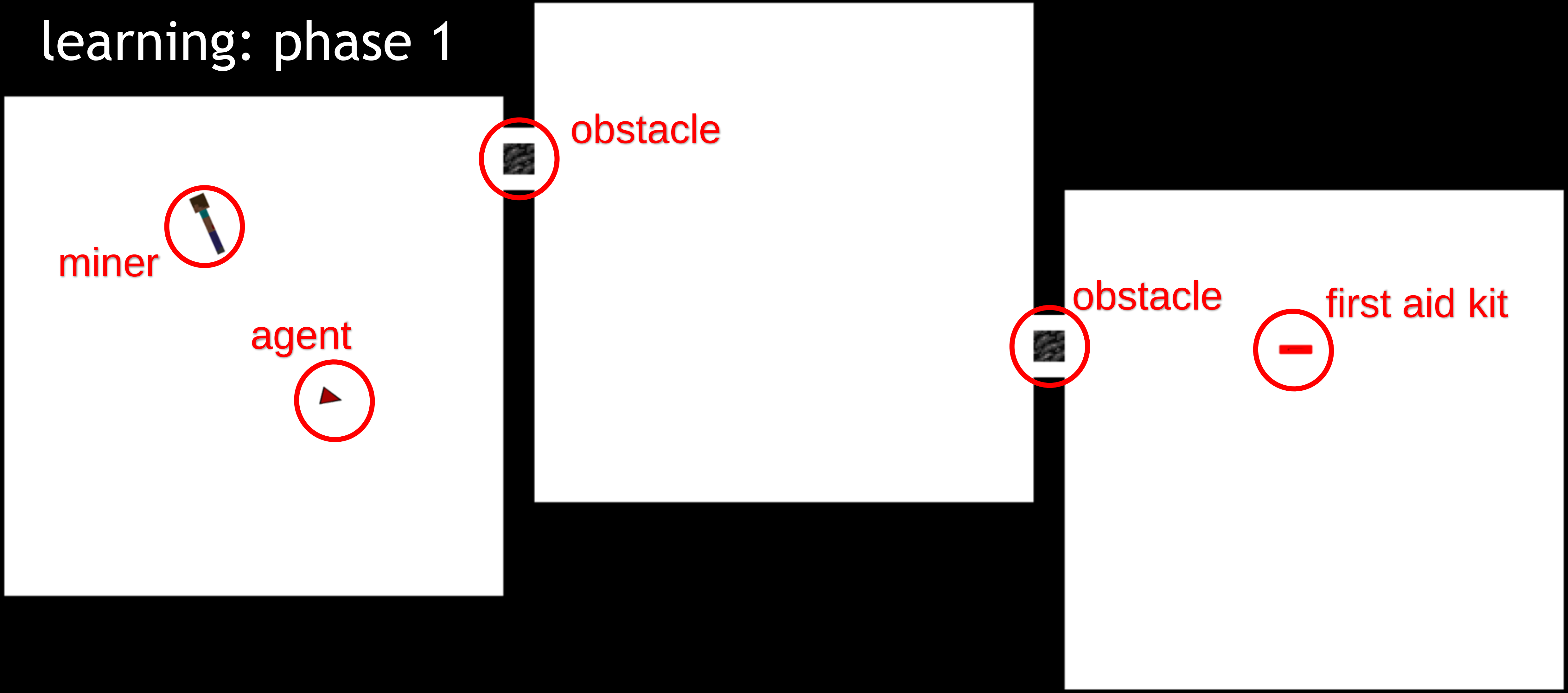
Problem two



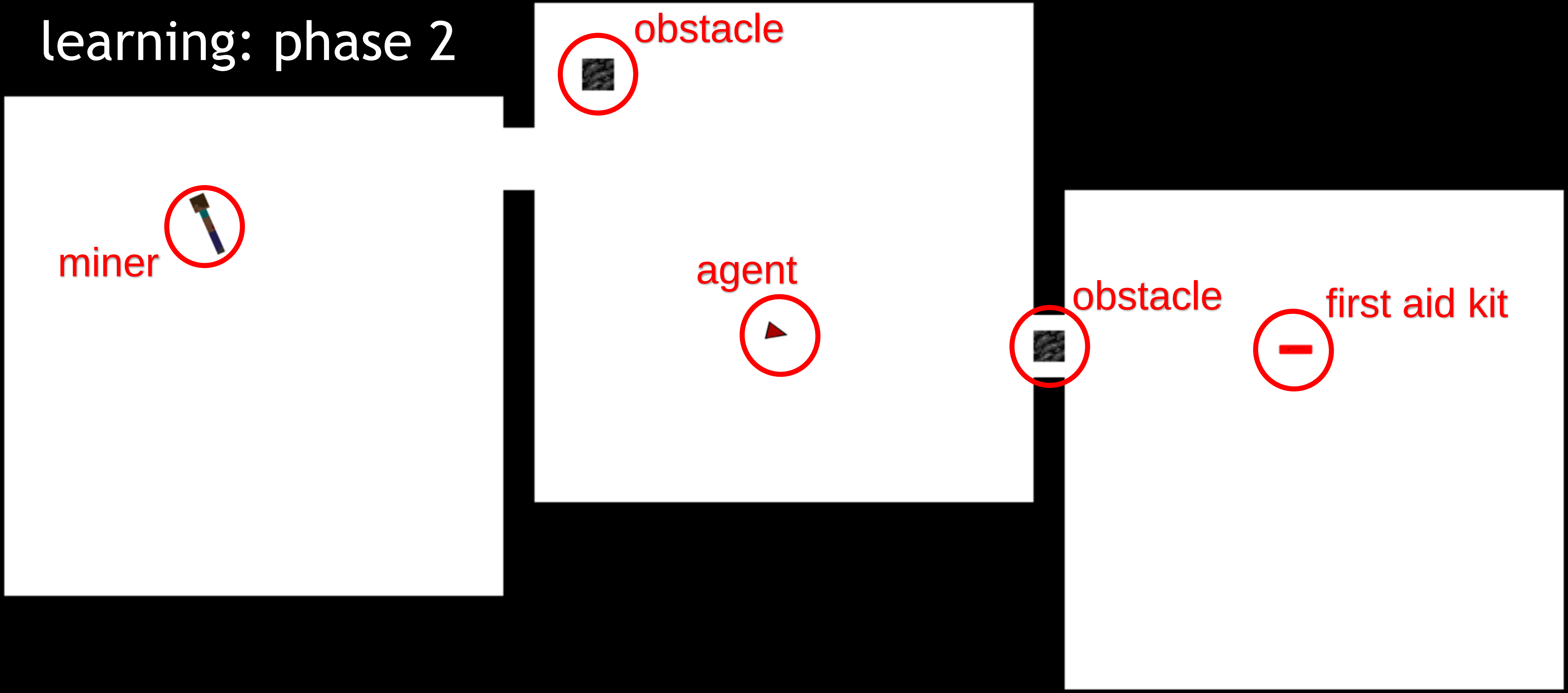
Possible Solutions

- Better exploration required
- Curriculum learning

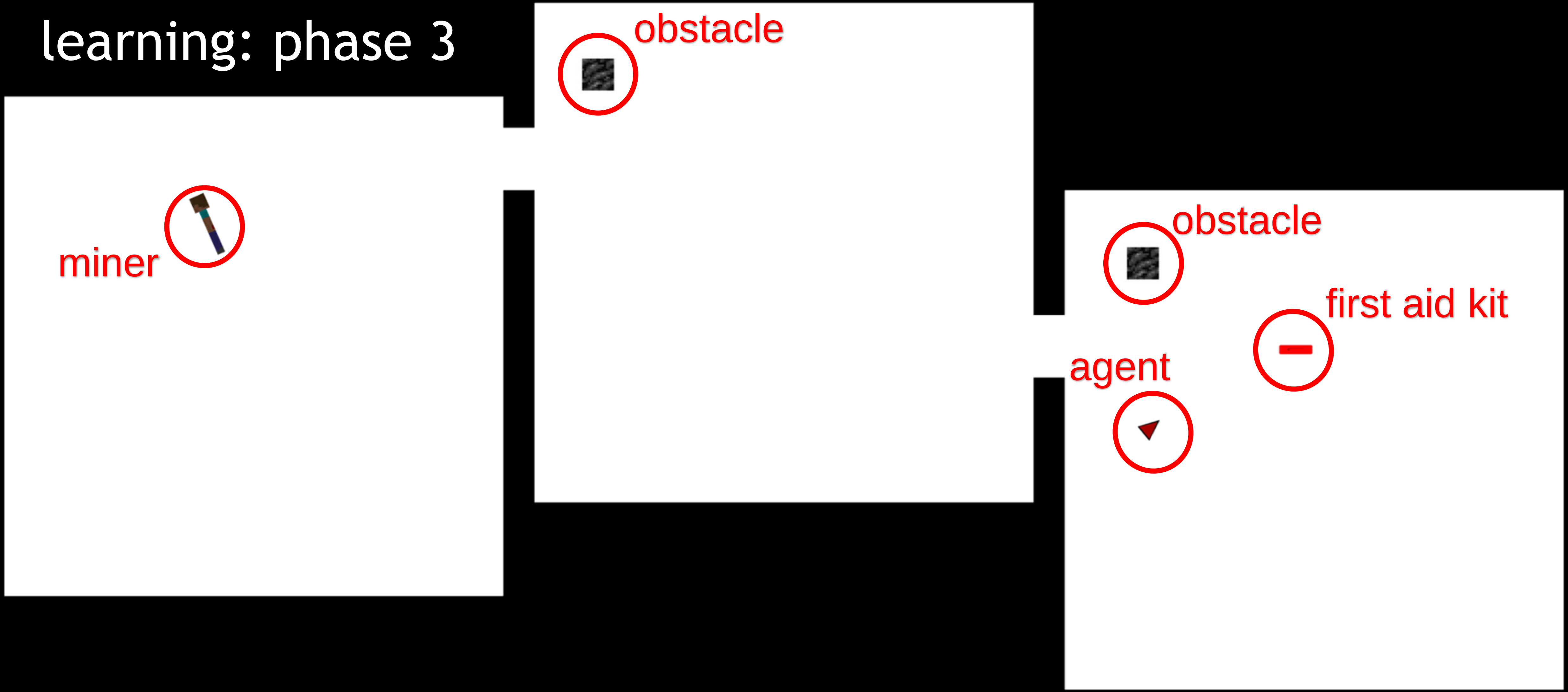
Curriculum learning: phase 1



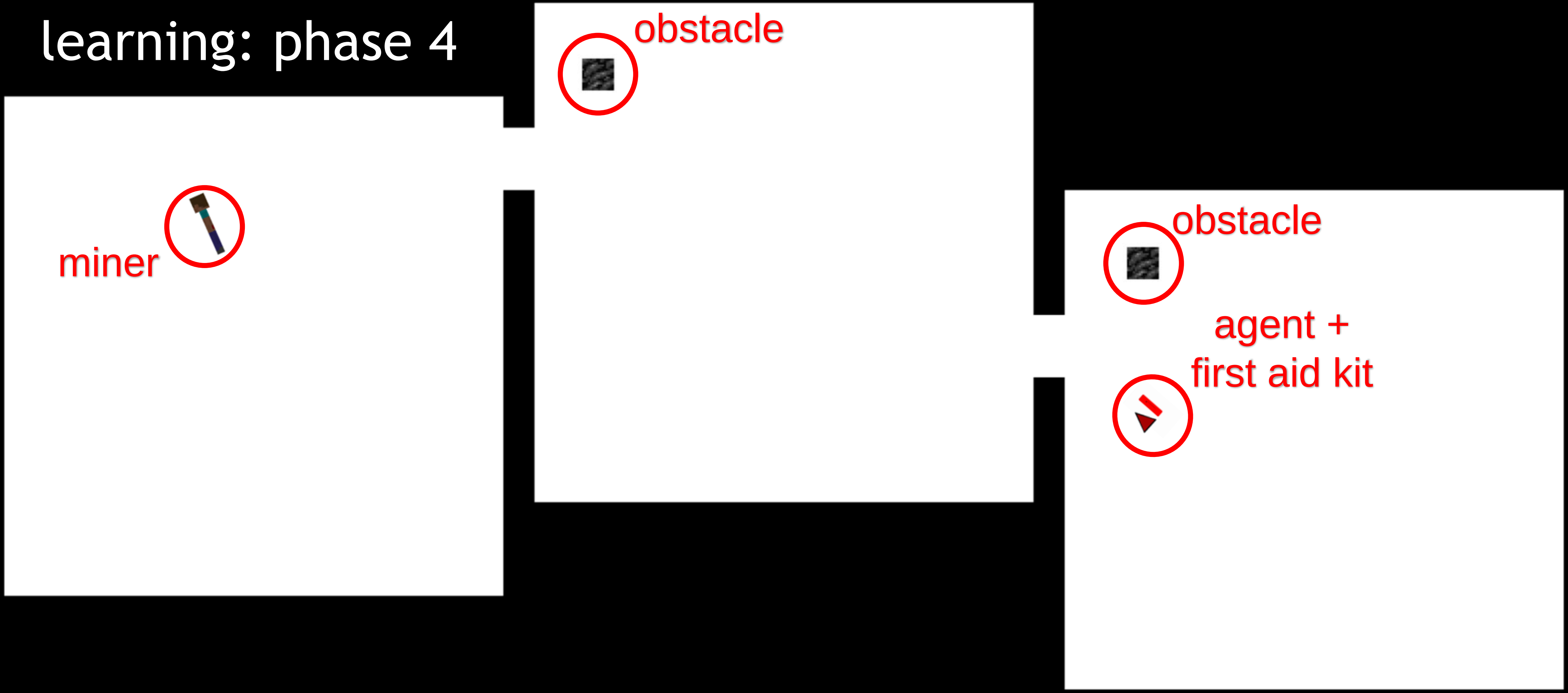
Curriculum learning: phase 2



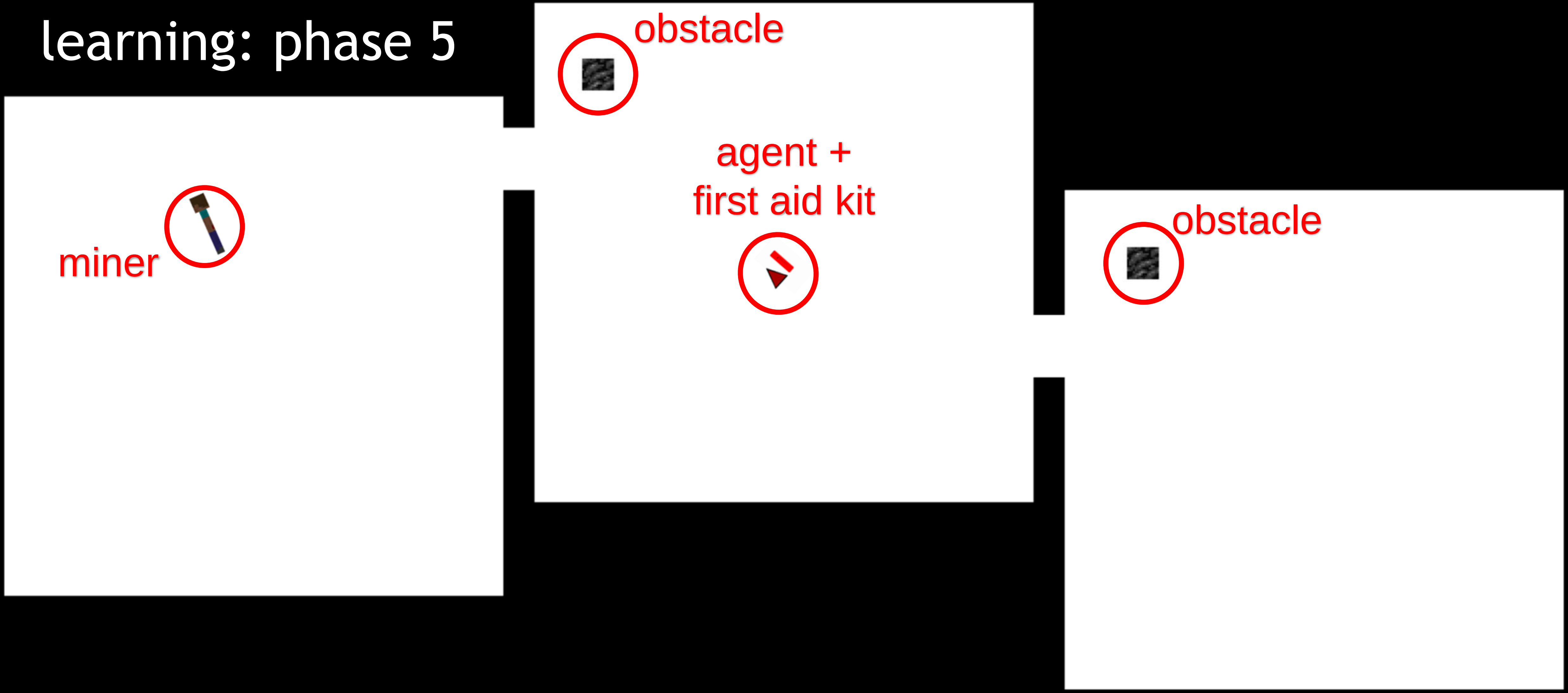
Curriculum learning: phase 3



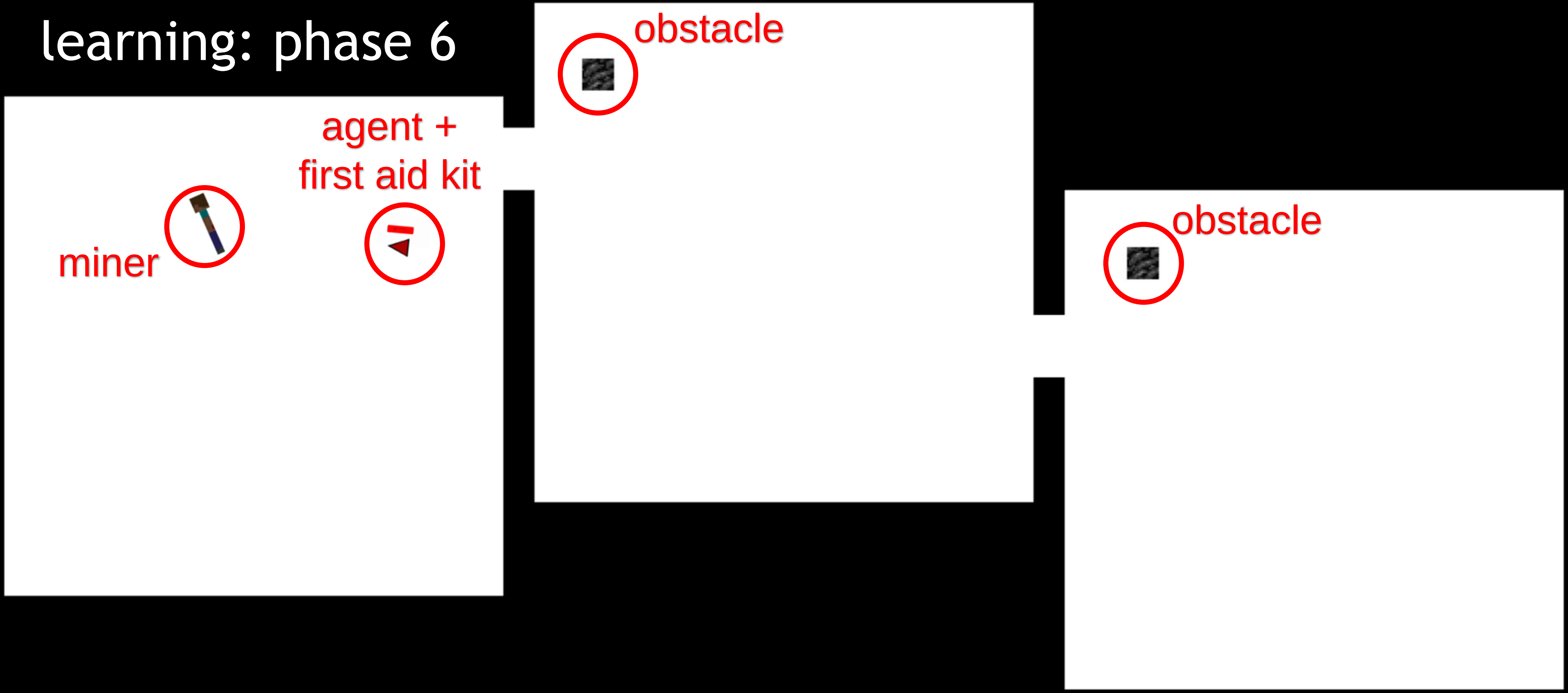
Curriculum learning: phase 4



Curriculum learning: phase 5



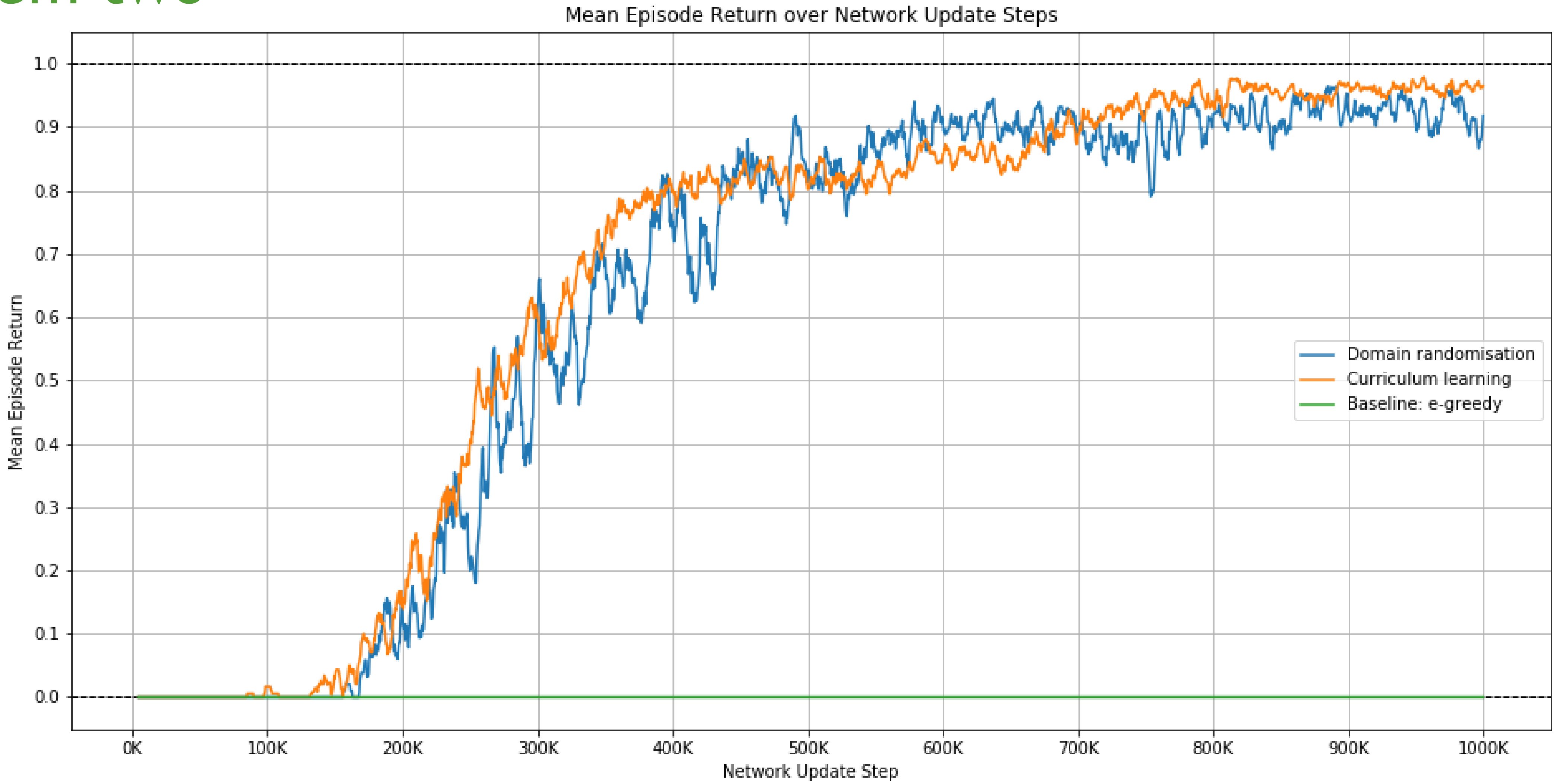
Curriculum learning: phase 6



Possible Solutions

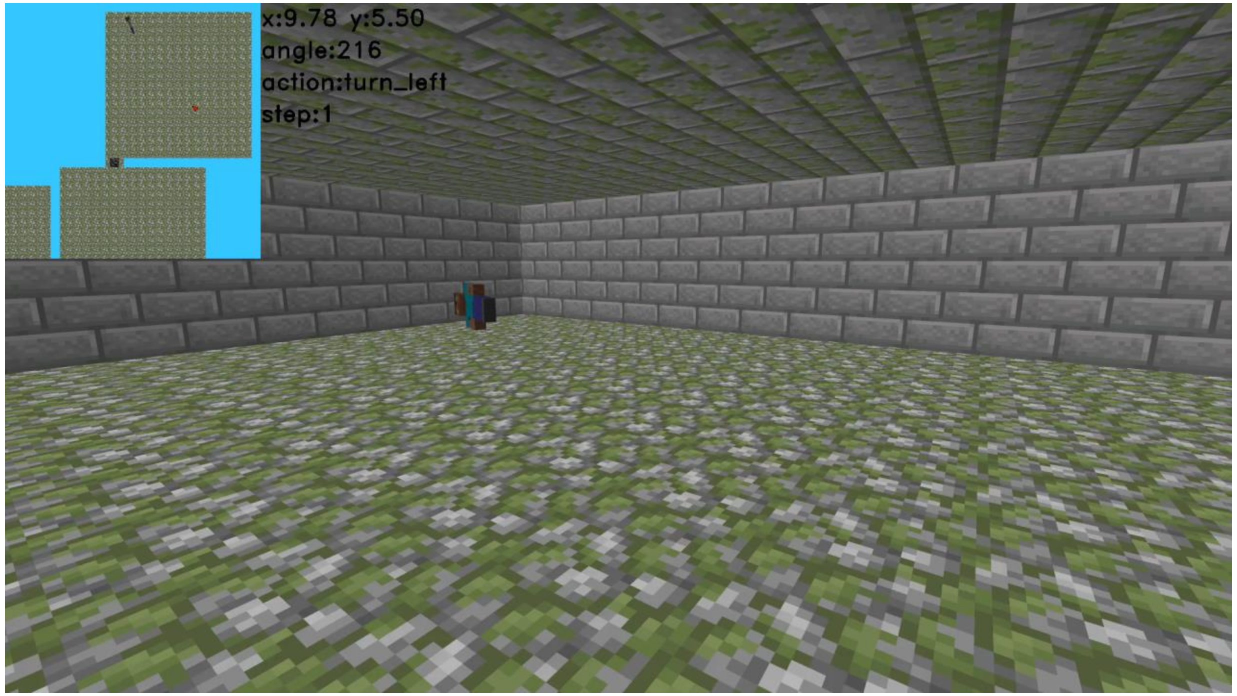
- Better exploration required
- Curriculum learning
  - Tedious to implement
- Environment initialisation
  - Randomise the rooms entities are placed (Domain randomisation)
  - Allows agent to learn simpler versions of the problem
  - Hopefully learns also to solve the more complex problem

Result
Problem two



Result
Problem two

Curriculum learning

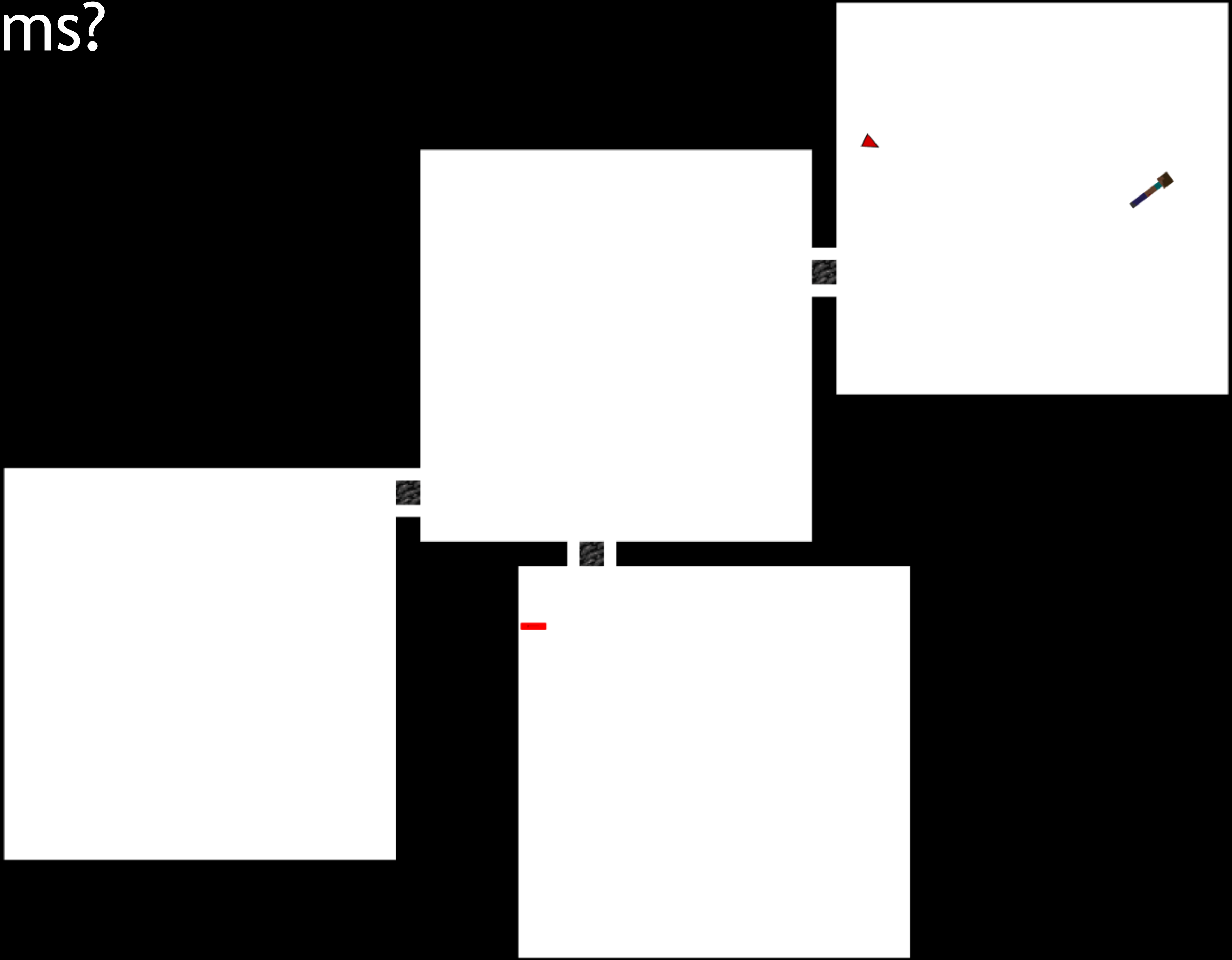


Domain randomisation

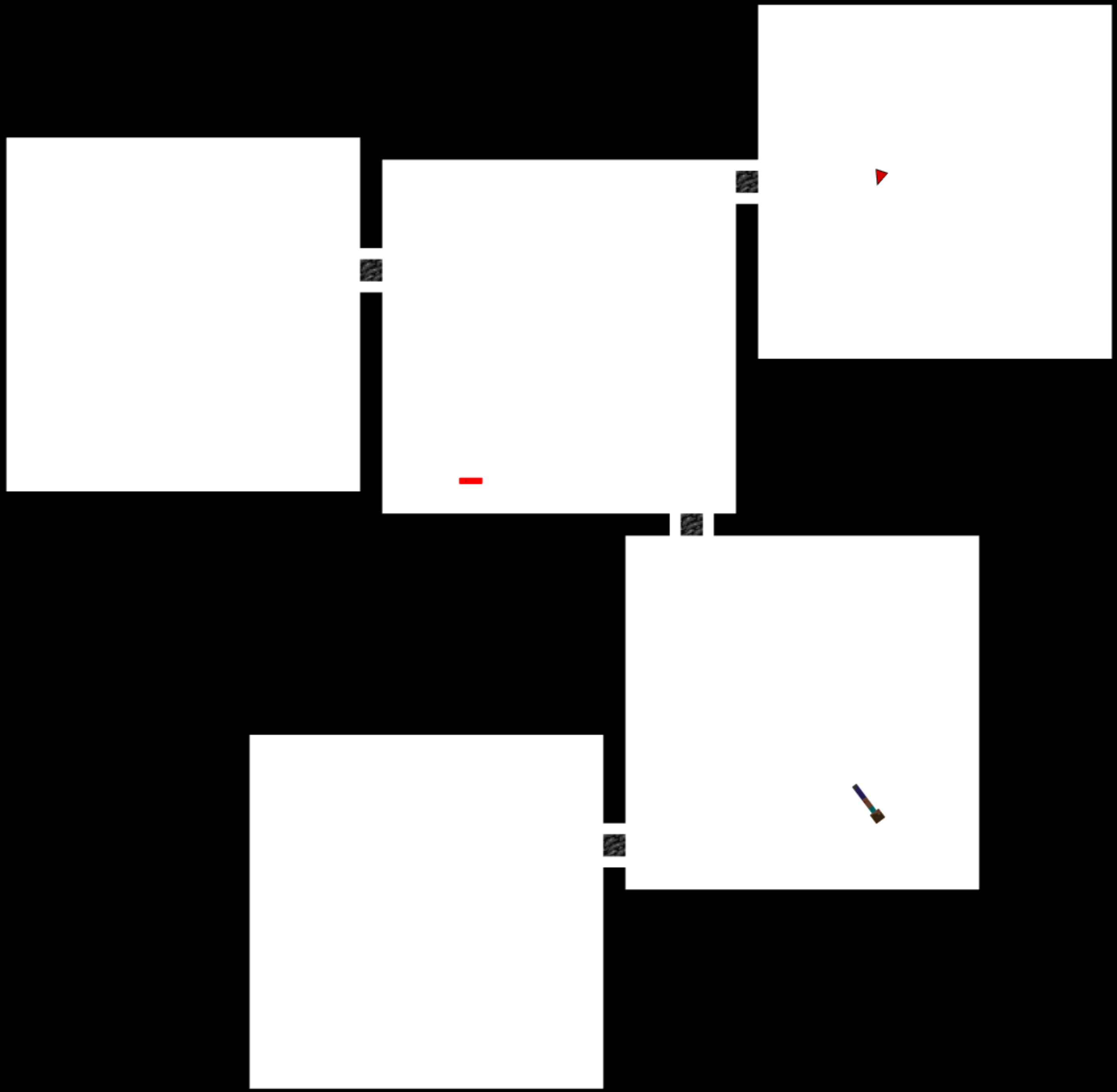
Scalable to Larger Environments?

- How well does the algorithm scale to larger environments?
- Domain randomisation
- Curriculum learning combined with domain randomisation

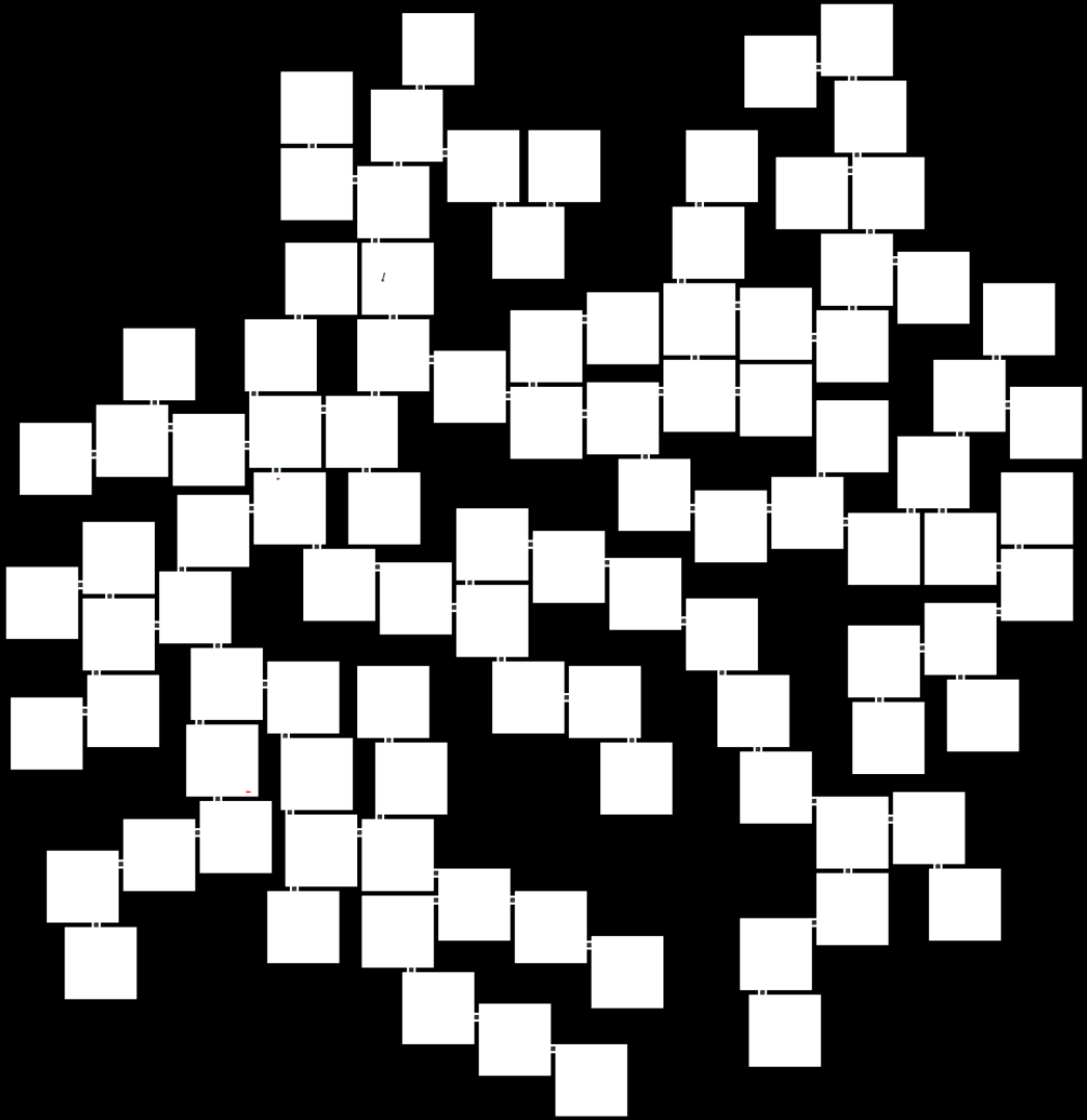
Four Rooms?



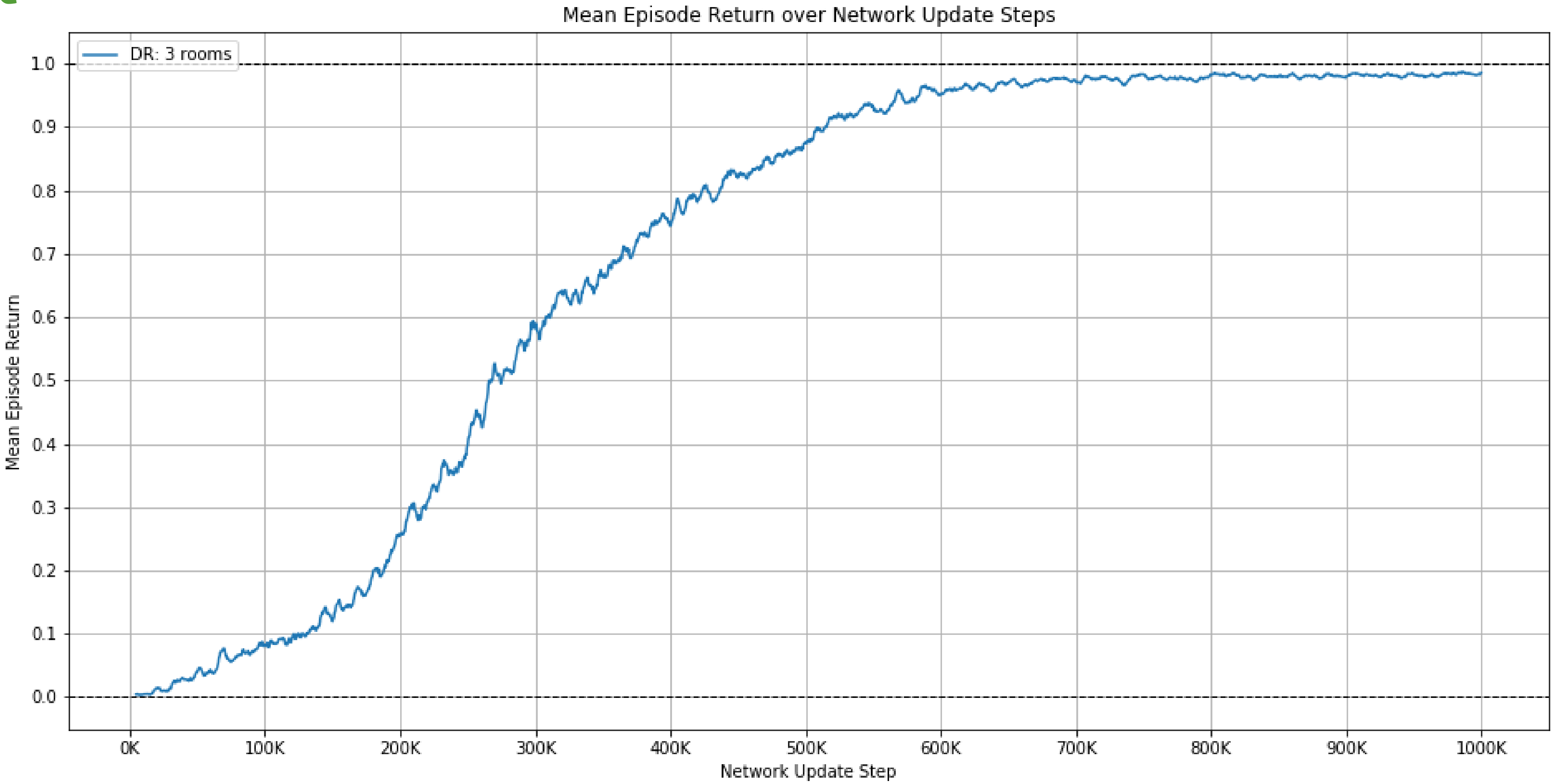
Five Rooms?



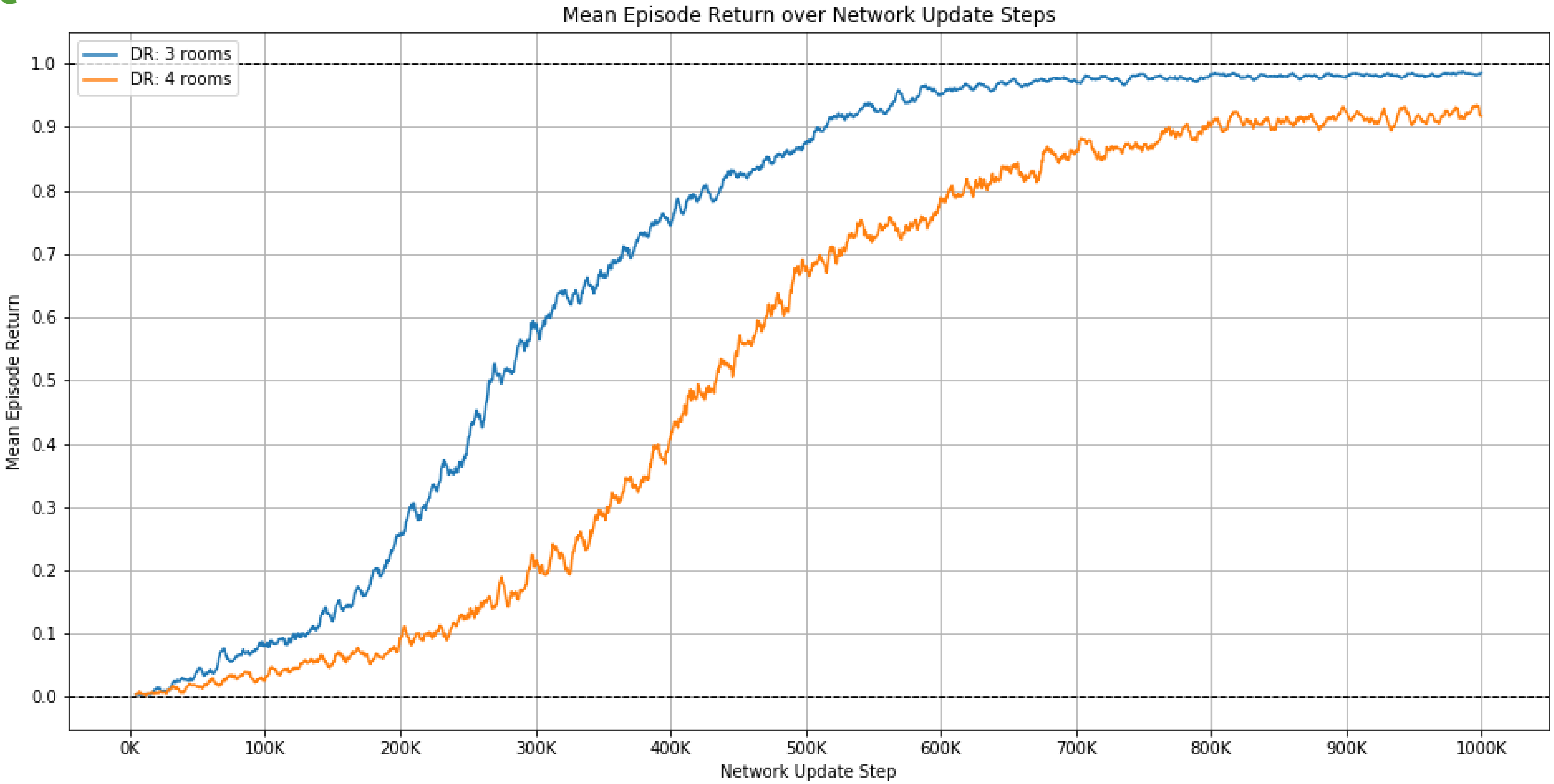
100 Rooms?



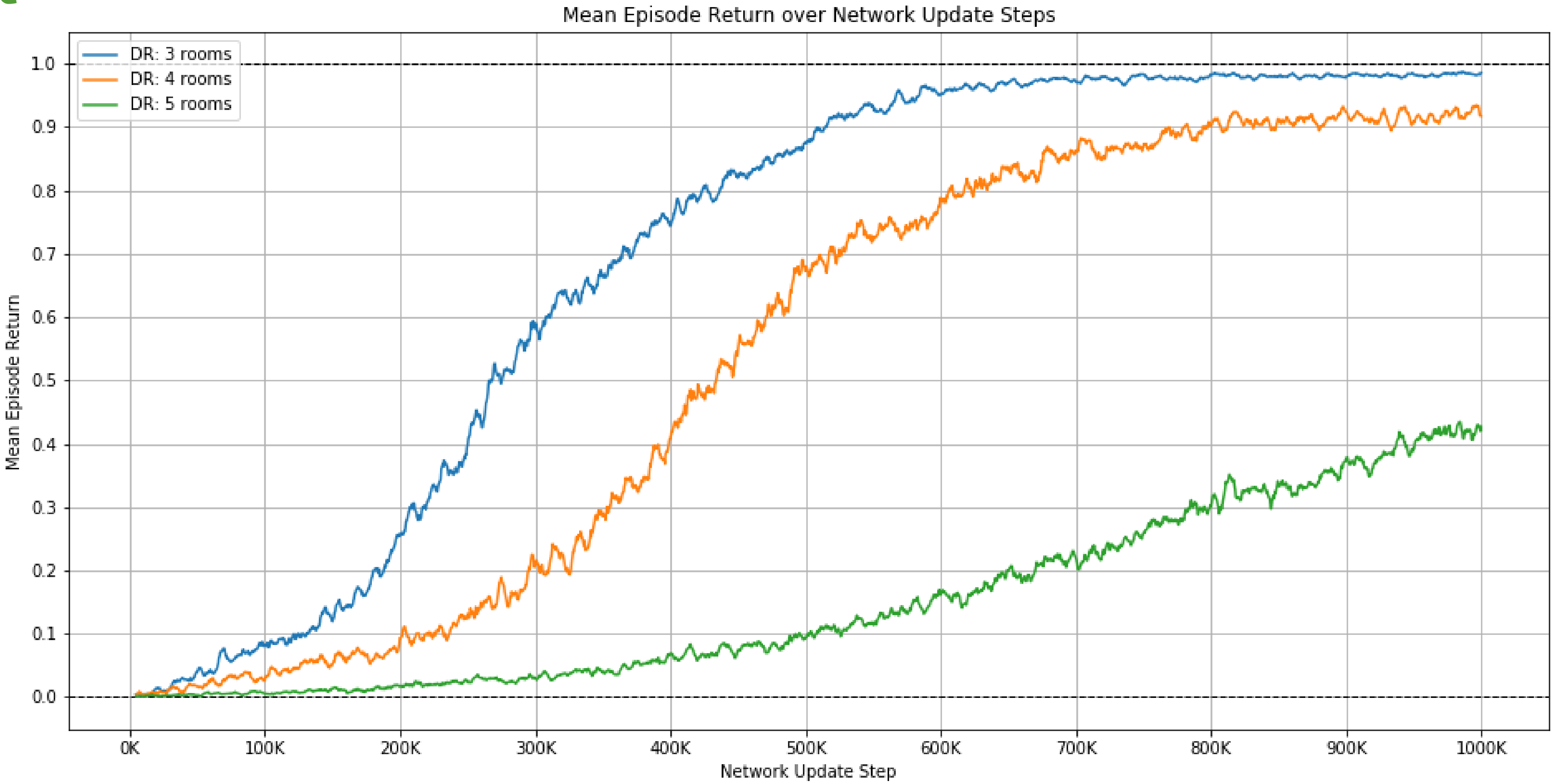
More Rooms Result



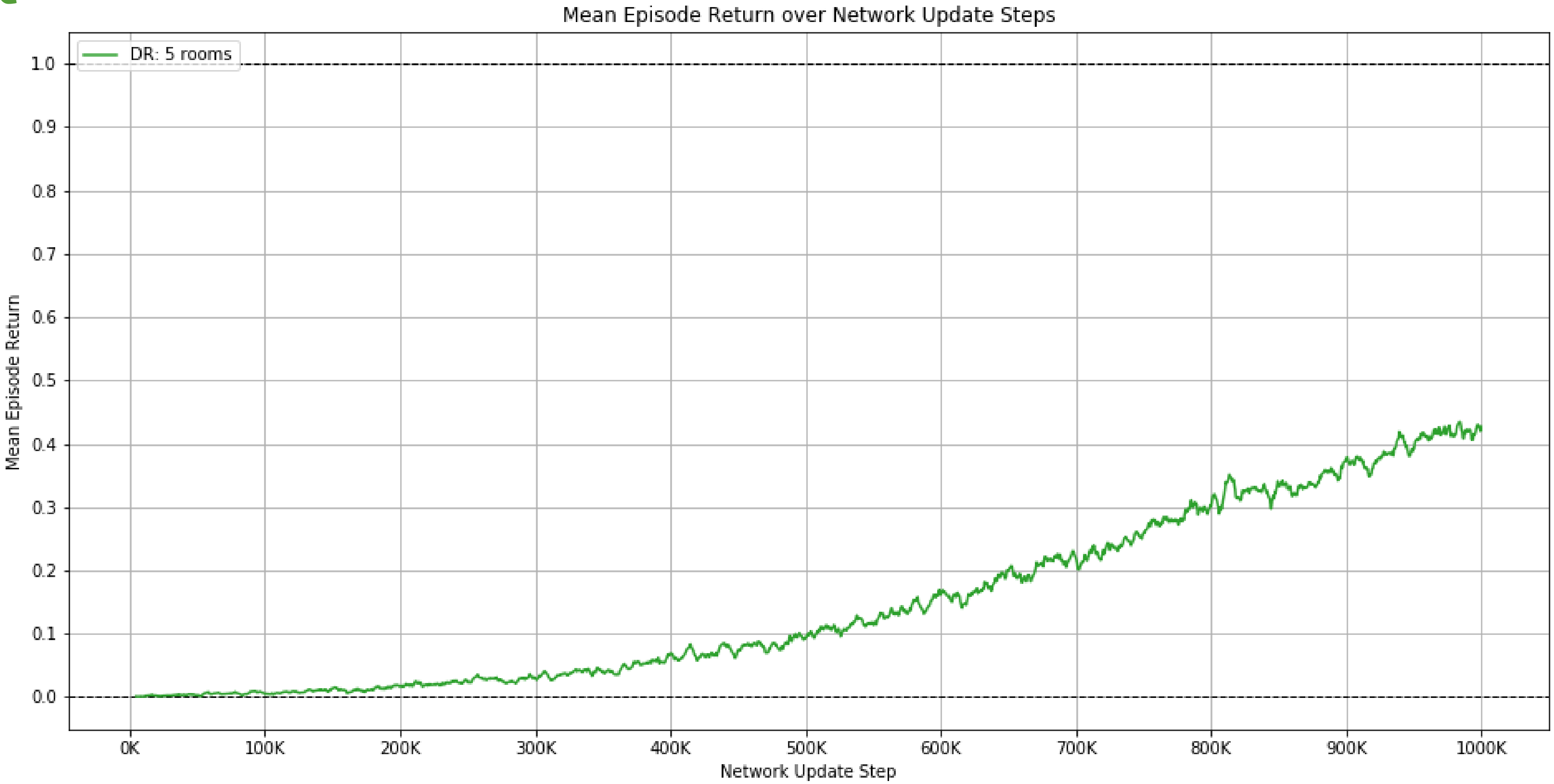
More Rooms Result



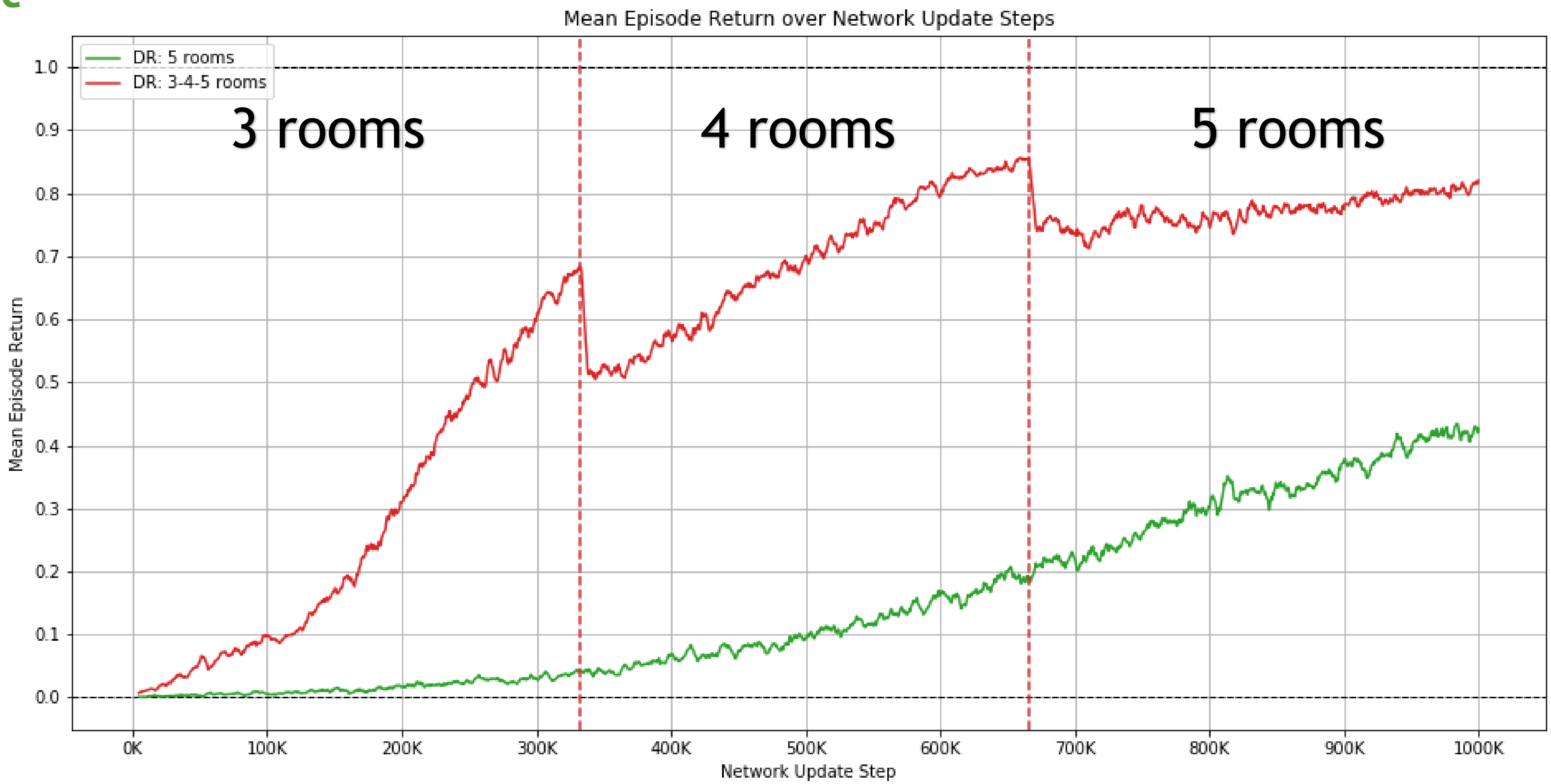
More Rooms
Result



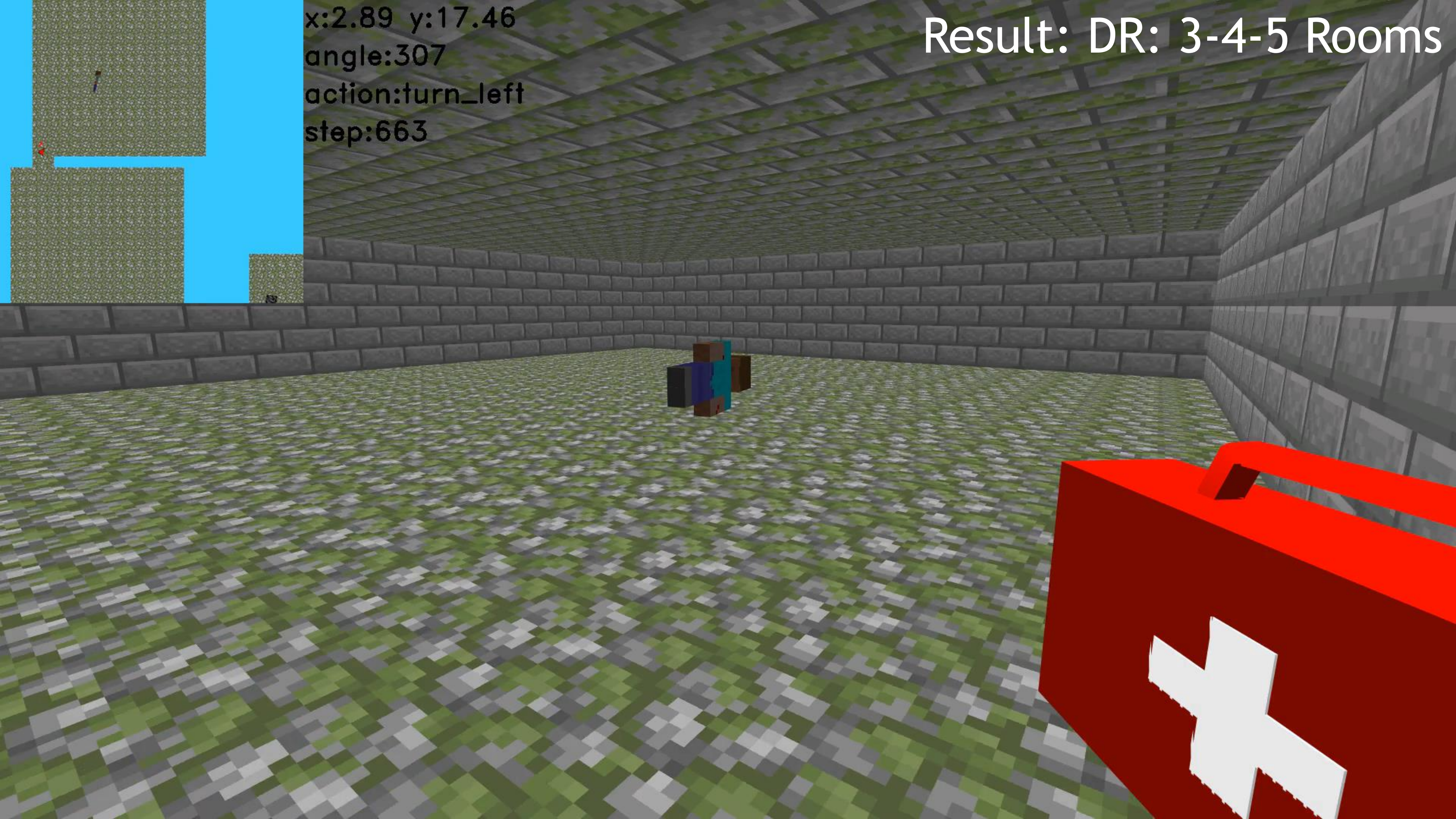
More Rooms Result



More Rooms
Result



Result: DR: 3-4-5 Rooms



Conclusion

- Important modifications to include for sparse reward problems
  - Distributed data generation
  - Prioritised experience replay
  - N-step update
- Exploration improvements
  - Curriculum learning
  - Domain randomisation
- Partially observability
  - Frame-stacking
  - Action memory
- Future work
  - Better exploration strategies (other than e-greedy)
  - Longer memory (LSTMs)

Applications

- General algorithm
- Only requires image observation and a reward signal

snake