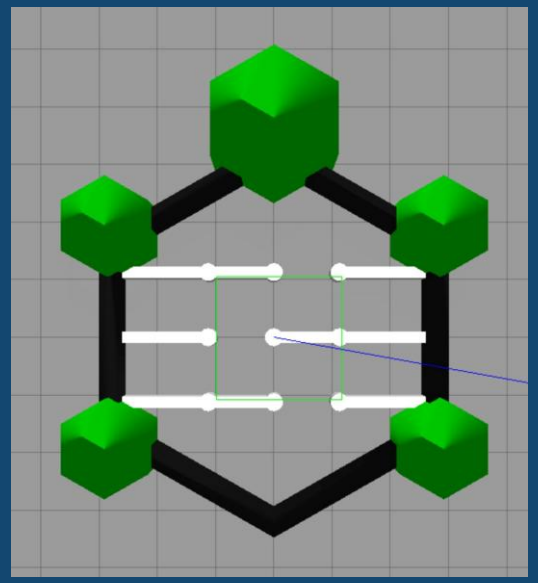
PROJECT PROPOSAL GROUP K

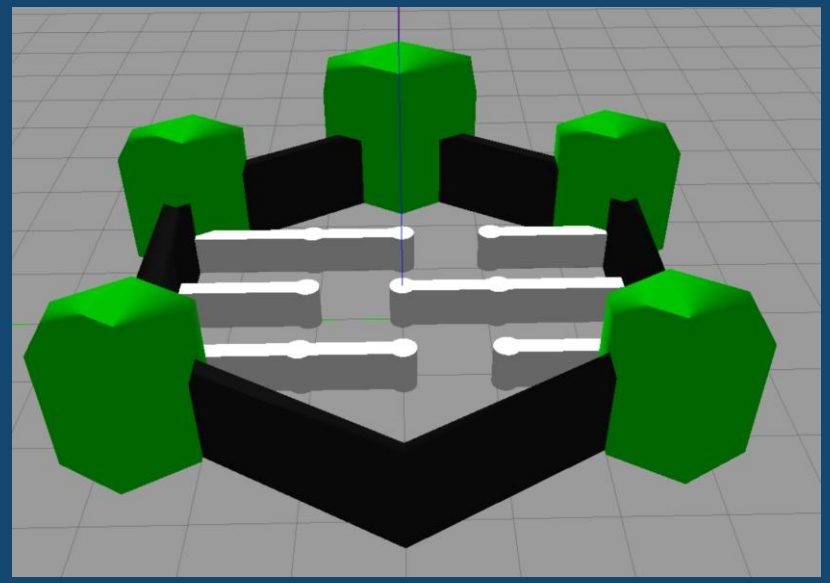
PILUTTI FILIPPO, SIMONITTI SAMUELE, FIORITTO ALESSANDRO

ENVIRONMENT

We decided to use the normal turtlebot 3 arena with some changes: we added 3 pairs of walls to form a simple maze that the robot should navigate without hitting said walls or any other solid object

Optional: If we find the robot too efficient in navigating this maze we plan to make it harder (by adding more obstacles or a longer route to the objective)





TASK

Primary task:

The chosen task is for the robot to navigate the maze without hitting the walls, the columns or the perimeter of the arena and after that, at the end of the arena, there is a specific position the robot should go towards until it reaches it.

Optional task:

After reaching the first given position, the turtlebot should navigate the maze backwards, back to the starting position.

INPUT AND SENSOR

For the visual input of the robot we want to exclusively use the laser sensor: this is because we figured that the laser would be less burdensome than image processing.

We also changed the number of scannings, from 5 scans (one every 72°) to 24 scans (one every 15°) to hopefully solve some problems with the original model, mainly the tendency to crash on the columns.

DRL ALGORITHM

We intend to use DQN algorithm as its fast training time could allows us to try different combination of weights and parameter values in a relatively short amount of time.

If we are not able to obtain good results, we plan to try Policy Optimization Algorithms (Reinforce or Actor Critic) which would require much more training time.

REWARD STRUCTURE AND CURRICULUM

We aim to use a more sparse reward scheme, specifically want to use goal sampling curriculum, meaning that we start from a simple objective (e.g. pass the first line of walls) and build up to a more complex one (reach the coordinates on the other side of the arena) and balance this with an exploration bonus; to optimize the maze navigation time and smoothness we also plan to implement factorized rewards (collision avoidance, time to target).

We want the robot to perform the maze navigation precisely and, with limited outside information, the exploration bonus could provides adaptability to different types of mazes.

TRAINING STRATEGY

In the past months, as we made some trainings on the different models provided we understood that a good choice of parameters and reward weights is crucial if we want to have good results: we want to train different models with less episodes to infer on what's the best combination of parameters for these specific objectives (mainly the decay rate, the batch size, learning rate and the reward function).

Also, choosing a good network architecture is pivital in ensuring the best outcome for the training so we plan to explore different types (e.g. Fully connected, Residual, RNN) to find the best suited one for our goals.

EVALUATION METRICS

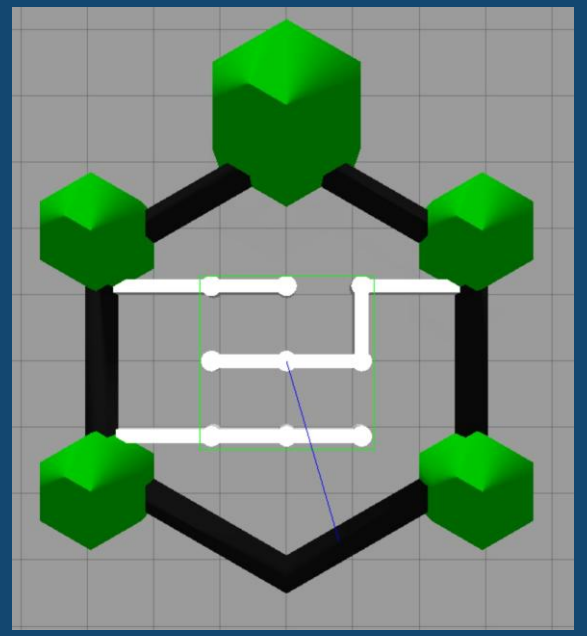
Having a number of clear objectives makes it easy to evaluate the success of the overall training process and, in addition to avoiding obstacles and reaching the final coordinates, we want to use the number of steps, meaning the time needed, to reach the final objective as another evalution metric.

This has also the purpose of helping us troubleshooting, as a slower time helps us to evaluate if we need to change our training strategy.

ACTUAL IMPLEMENTATION AND ACHIEVED GOALS

ENVIRONMENT

As proposed, we used the original maze for all the training phases we did. As instructed by the professor we also created two more mazes with the aim of testing the generalization of the training: one specular to the original one and a more complex one, but sadly we didn’t get to use them as the training was never that successful.



TASK

Unfortunately we didn’t fulfill the original proposed goal: the robot, in all the tested configurations, had serious issues with the navigation, especially towards the medium goal provided (which shouldn’t have been, in our opinion, too difficult to reach).

This was the primary concern that didn’t allow us to continue with the actual testing phase.

INPUT AND SENSOR

We tried modifying two things from the original proposition: the number of readings, decreased from 24 to 18 (one every 20°) and the linear and angular movement velocity; with the former we didn’t notice significant changes in the results but the latter helped the robot to do more precise movements and apparently marginally lowered the collision rate in some of the tests.

DRL ALGORITHM

Since using the normal DQN provided very poor results we opted to switch to another version of it called DuelingDQN: this version works by separating the state value $V(s)$ from the action advantage $A(s, a)$ and making the learning more stable when many actions have similar effects, as in our case.

The improvement was noticeable as the success rate climbed significantly in the final batches of the training episodes but this didn't solve the main problem of the navigation towards the medium goal.

REWARD STRUCTURE AND CURRICULUM

As proposed originally, we used a Goal Sampling Curriculum with 3 types of goal, named Easy, Medium and Hard: the easy goal was directly in front of the starting position of the robot and the medium goal was slightly to the left of the former. The robot refused to navigate with stability towards the medium goal in any of the tests we have done even and especially when forced to choose this goal in the training.

The usual reward structure used to train the robot was a factorized reward composed of collision avoidance, time to target, goal proximity and yaw but we sometimes tried to cut some of them for troubleshooting purposes.

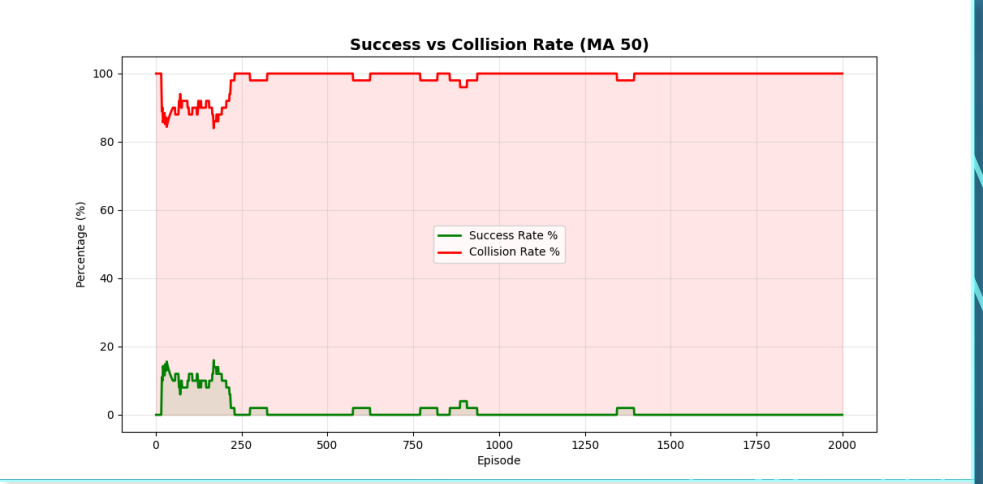
TRAINING STRATEGY

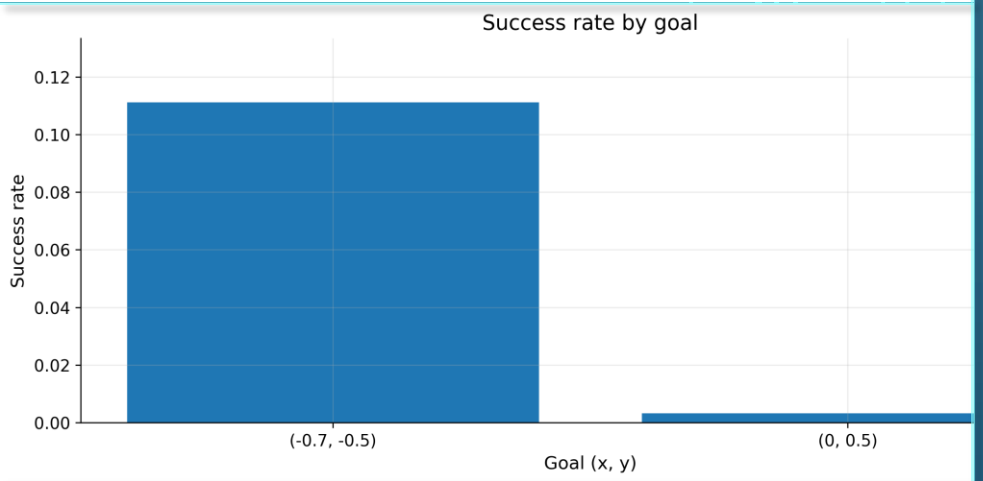
As already discussed we tried various configurations with different function and training parameters, different reward weights and different functions to calculate those rewards. Once we noticed the aforementioned problems we divided the group to search more efficiently for a solution: some tried various configurations for small training periods (2000 episodes) and others tried a multi-phase training (starting from a maze without obstacles and increasing the difficulty gradually).

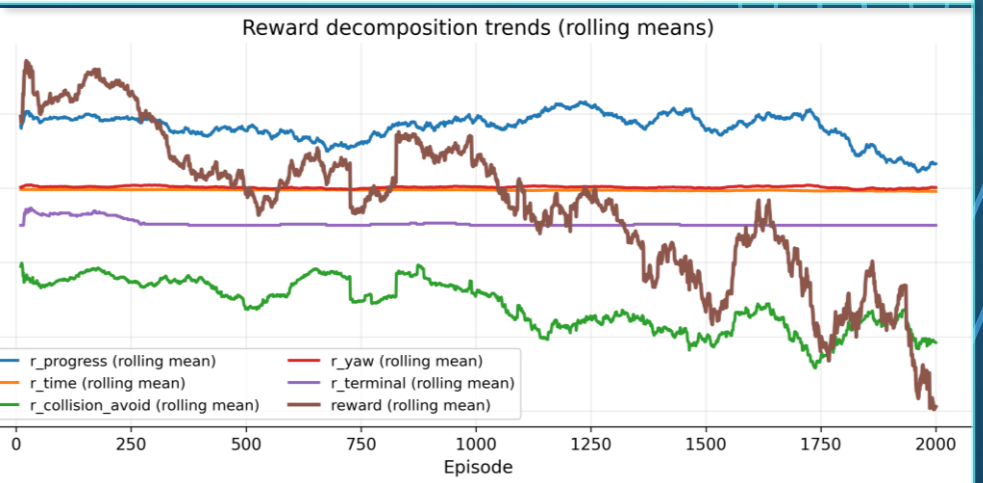
EVALUATION METRICS

Using the Goal Sampling Curriculum and the other metrics proposed originally as well as a simple script to plot the results created after all the trainings in metrics.csv was fundamental to test different configurations and realize which one worked better.

We also analyzed a single goal success rate to present the results achieved by showing that the robot had difficulty reaching goals other then the first if not by pure chance.

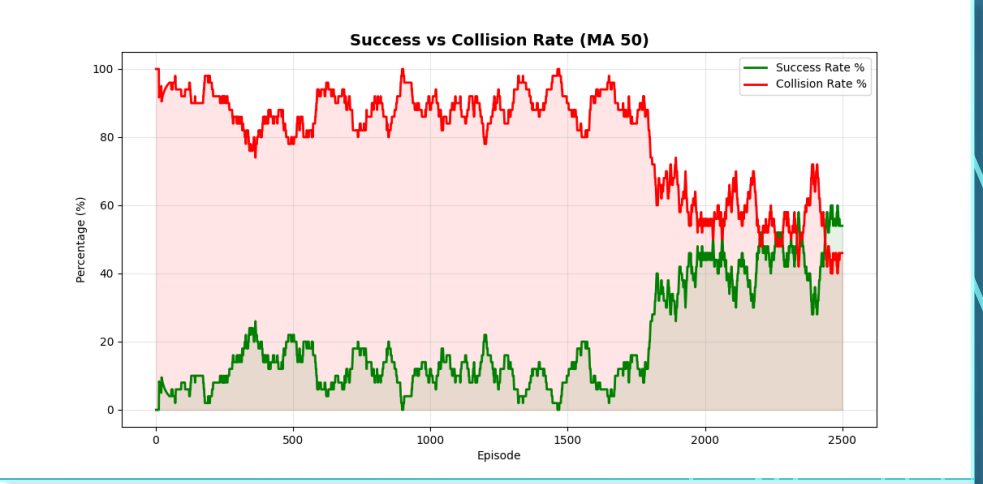


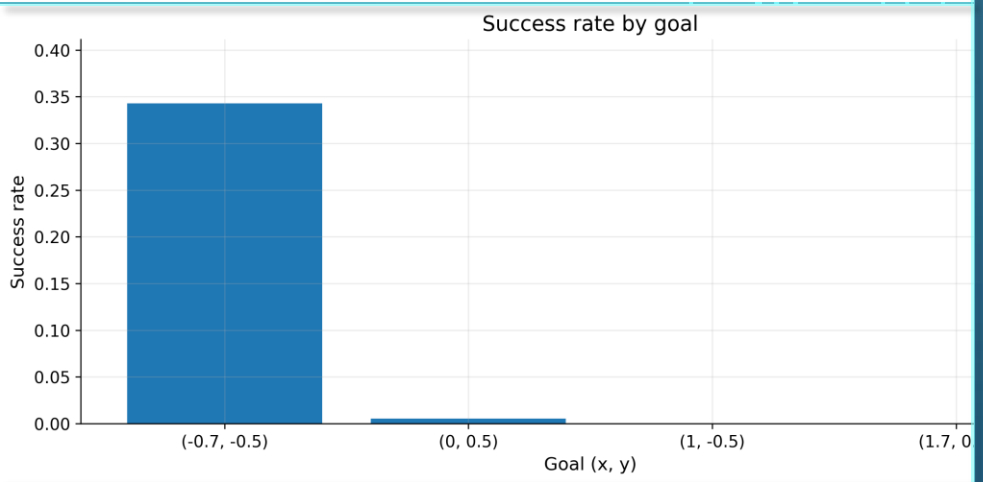


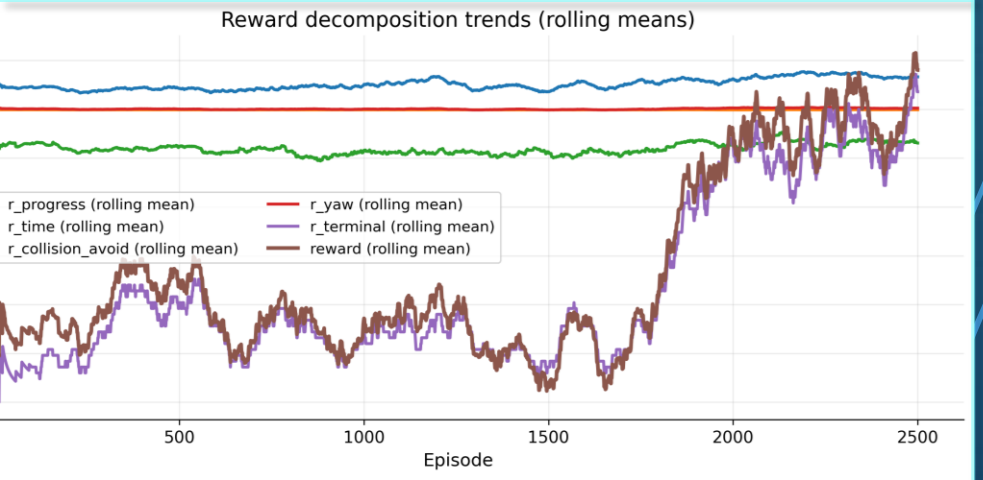


RESULTS A

Test number 6: after noticing that the robot failed to reach the medium goal we tried to force the choice of the goal but this resulted in the plummeting of the success rate

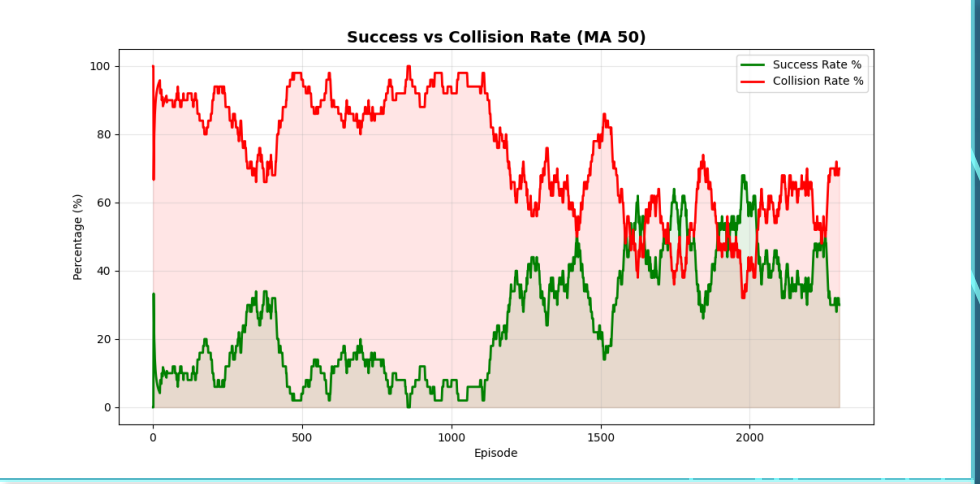


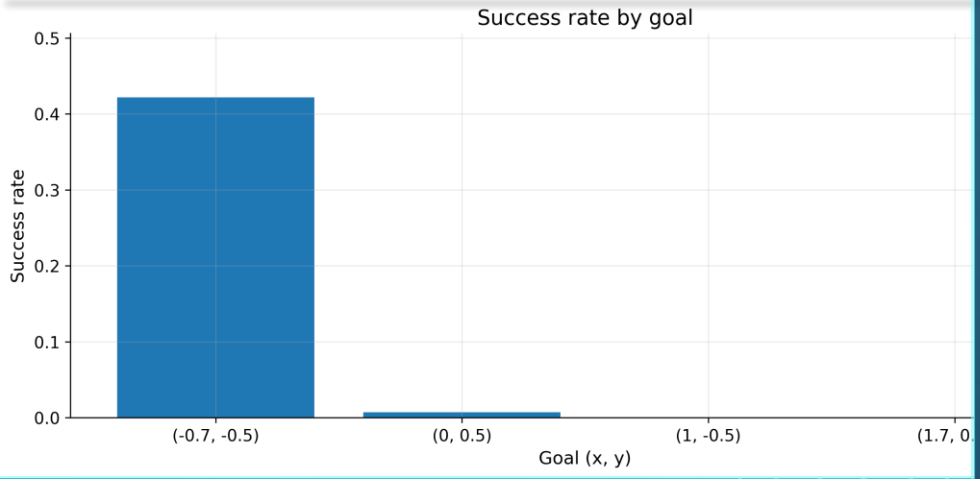


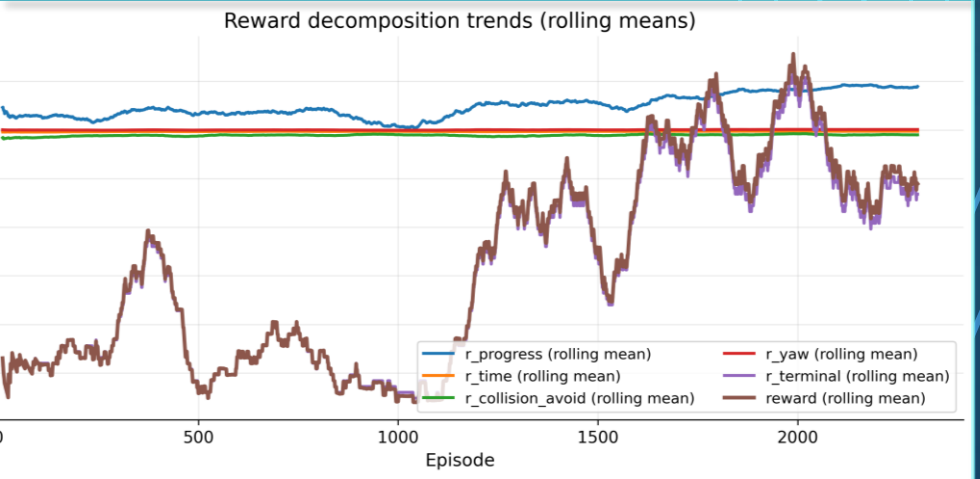


RESULTS A

Test 14: we introduced the DuelingDQN network and this resulted in a significant improvement in the success rate although it was boosted almost entirely by the navigation to the easy goal.







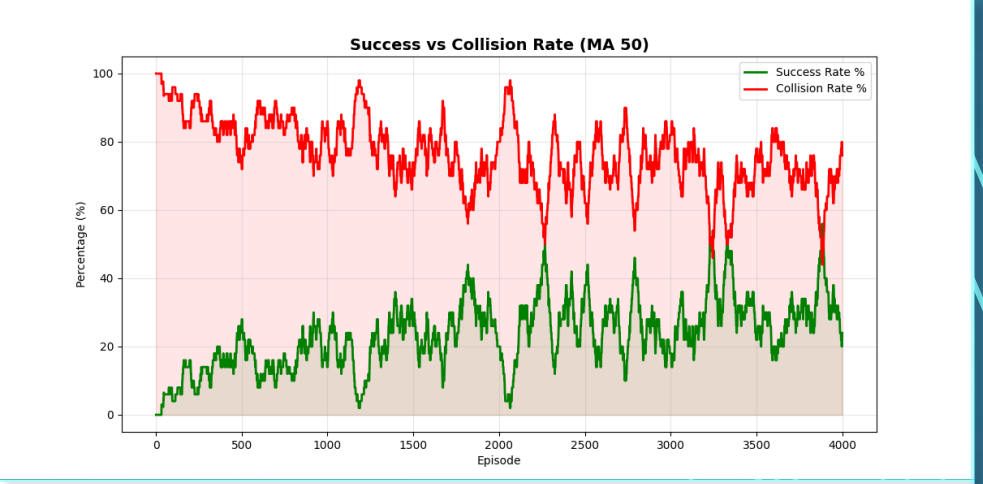
RESULTS A

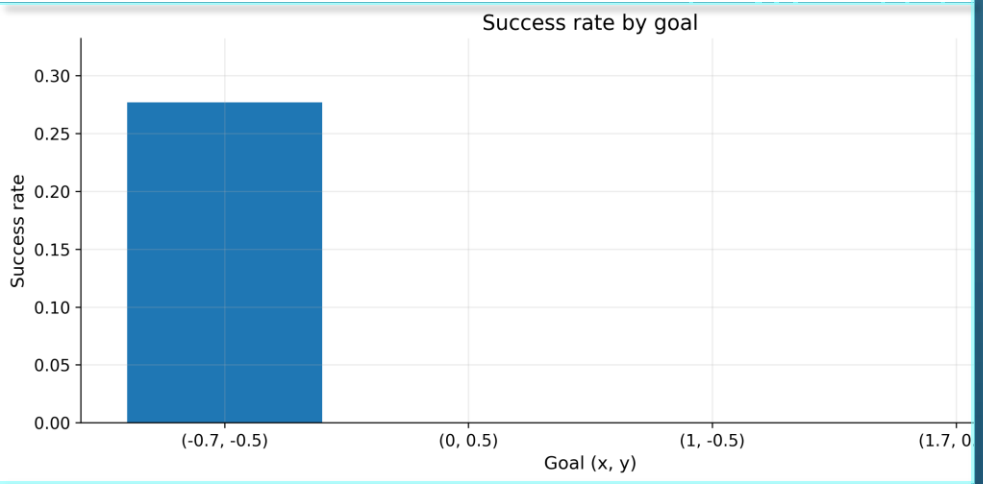
Test number 24: we implemented all the positive changes we found in this last test with some decent results (but with the same main problem as before) as well removing the direct reward for the movement towards the objective: this seemingly made the robot slower in the navigation but more precise as it was mainly guided by the yaw and collision avoidance rewards.

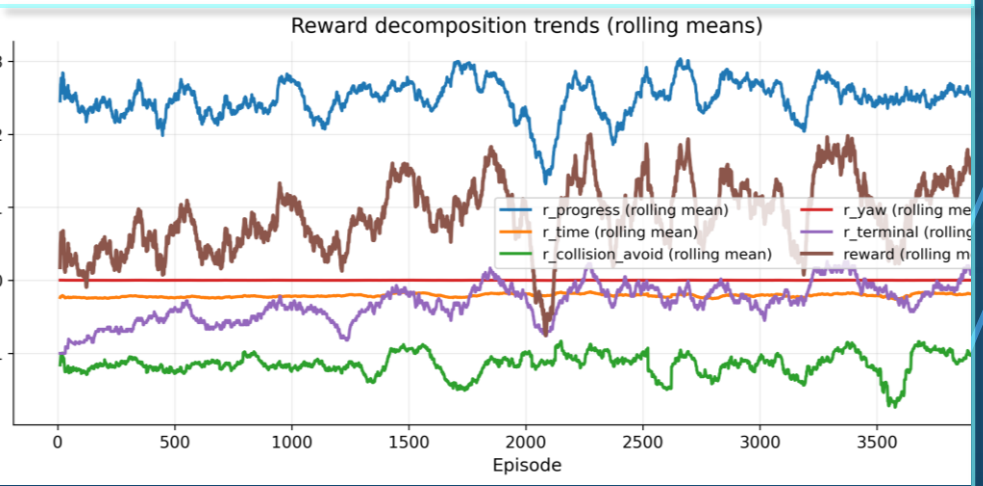
RESULTS B

In this results we adopted a structured four-stage training strategy aimed at progressively increasing the complexity of the environment and the task difficulty:

- Phase 1: empty arena with no obstacles, the agent was required to reach a randomly generated goal within the map.
- Phase 2: arena containing only columns as obstacles, the agent was trained to reach a fixed goal with hard difficulty.
- Phase 3: arena with a maze layout, the agent was required to reach medium and hard difficulty goals.
- Phase 4: arena with a maze configuration, introducing a goal sampling strategy to enhance generalization and robustness.



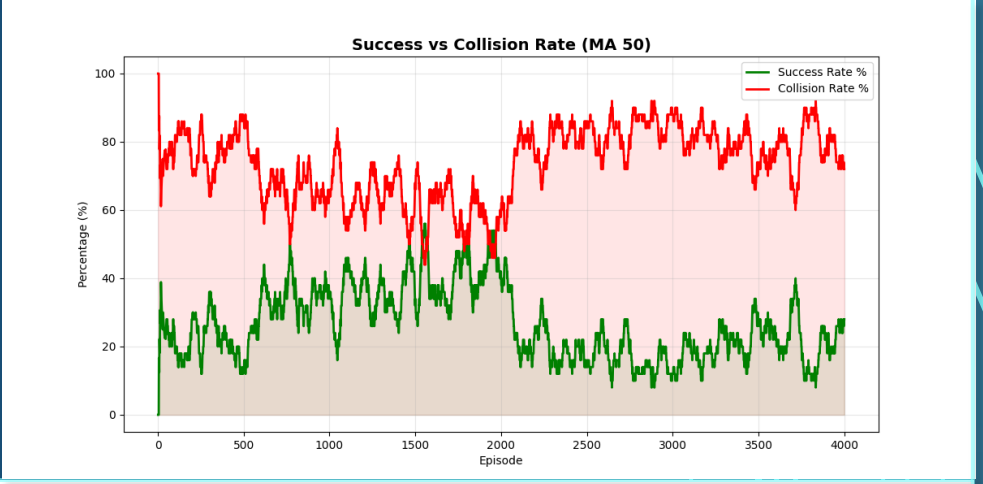


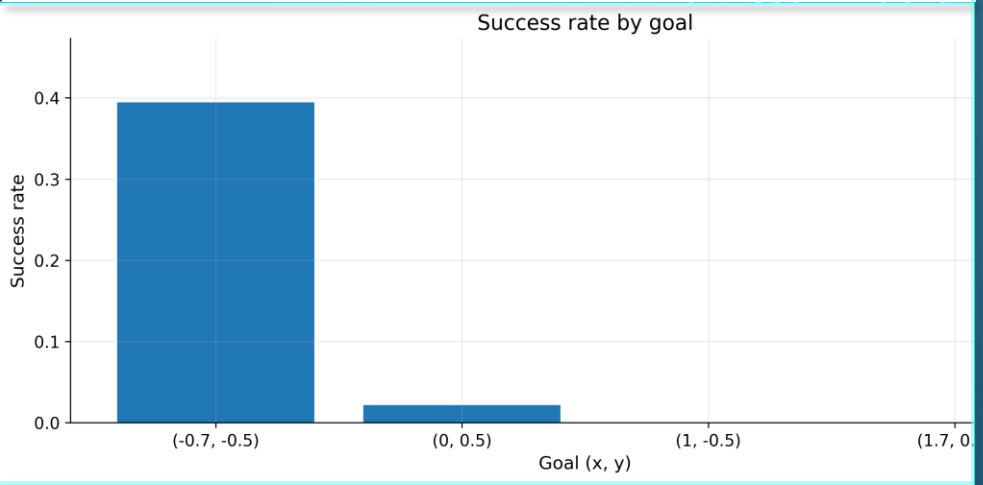


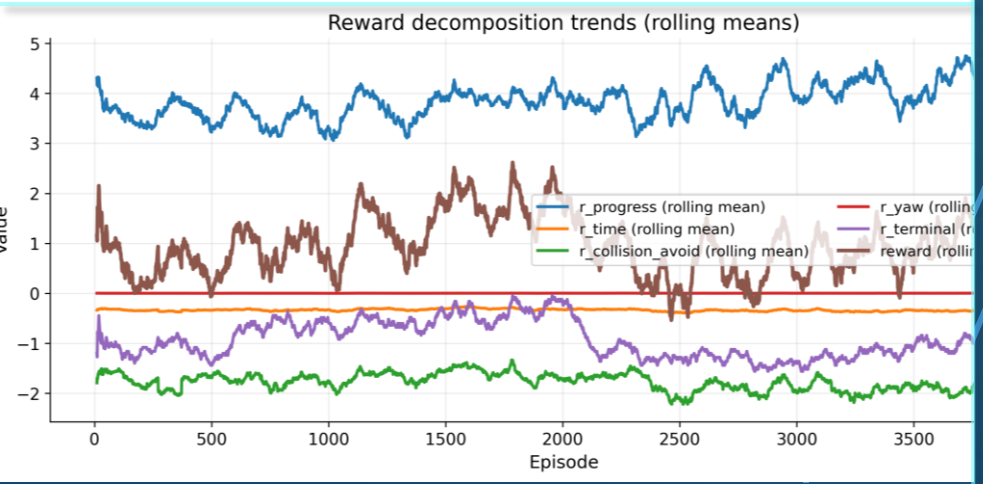
RESULTS B

Test number 8: the results were obtained during the fourth phase using a standard Deep Q-Network (DQN) algorithm.

The Collision vs. Success Rate trend appears overall promising, showing a gradual improvement in performance and a relatively stable behavior over time. However, when analyzing the success rate per goal difficulty, a critical limitation becomes evident: the agent consistently reaches only the easiest goal, while medium and hard goals remain unsolved.







RESULTS B

Test number 12: these results were obtained using a Dueling DQN architecture. Unlike the standard DQN, Dueling DQN separately estimates the state value and the advantage of each action, which can improve learning stability and action evaluation.

From the Collision vs. Success Rate graph, the overall trend appears slightly worse compared to the previous experiment with the standard DQN, with lower stability and a higher collision rate.

However, the success rate per goal difficulty shows a small but meaningful improvement: the agent has started, albeit marginally, to reach the medium difficulty goal. While overall performance is still limited, this indicates a first step toward handling more complex tasks.

PROPOSED FUTURE WORK

As the training wasn’t successfull the proposed objective for the future is to fix the problems, mainly the goal navigation one: this can be made by totally chaning the factorize reward structure (as we did in some of the final tests where it proved to be somewhat useful) or modifying the DRL alghoritm by using, for example, the Actor Critic method: this was originally proposed as an alternative but due to time constraint we weren’t able to implement and test it.

While researching online we also found some papers that proposed the use of more movement options, namely the Turn «strongly» left/right: they support the idea that by adding these options the robot could navigate the environment with more precision