In this section the detailed of our implementation is introduced and ordered in a way for convenient reproduction and verification. The disc robot has two types of sensors mounted – two distance sensors and one contact switch. The input from two distance sensors are mapped as state for Q-table, and the contact switch is used to approximate collisions with obstacles in the environment.

1. Packages and parameters

The platform chosen for conducting this experiment was MATLAB 2020b[[1]](#endnote-1). Robotic playground and its dependencies[[2]](#endnote-2) were used for the modeling of the disc robot and obstacle environment. The default disc robot from the package was used, and its detailed parameters were as follows: The distance between wheels of the disc robot was 0.5 meters, with wheels of radius 0.1m. The two mounted distance sensors had a position offset of [0.2 0.15] with orientation offset 45 and a position offset of [0.2 -0.15]. Both sensors had maximum range of 20 meters and minimum range of 0 meter, with a resolution of 0.01 meter and a sample time of 0.1 second. The input value will be processed to be accurate to 0.2m and with a range of 0 to 1.8 meters inclusive. The contact switch had a position offset of [0 0], with orientation offset of 0 degree. The distance for it to be triggered was set at 0.02 meter. It was much less than the radius of the disc robot itself, but this setting gave the best estimation of collision. The arena was of size 5 x 8 and the starting position of the disc robot was [-1.4 0]. In the simple setting, a block of 3 x 5 was placed in the position [0 0] of the arena. The medium setting had two blocks of size 2 x 2, placed at [0 2] and [0 -2].

1. Implementation of Q-learning

Broadly speaking, the Q-learning algorithm requires two types of input and produces one kind of output. A reward signal is needed for punishing or rewarding the robot. The current (state, action) as well as the future (state’, action’) after action was implemented was also needed for the update of Q-table entry correspond to (state, action).

The update function we used is as follows:[[3]](#endnote-3)



When given a state, Q-learning will search the list of actions for the highest value and output the corresponding action.

1. (@manual{MATLAB,

   address = {Natick, Massachusetts},

   organization = {The Mathworks, Inc.},

   title = {{MATLAB 9.9.0.1592791 (R2020b) Update 5}},

   year = {2020}

   }) [↑](#endnote-ref-1)
2. (MathWorks Student Competitions Team (2021). Robotics Playground (https://github.com/mathworks-robotics/robotics-playground/releases/tag/20.1.4), GitHub. Retrieved March 4, 2021.) [↑](#endnote-ref-2)
3. <http://incompleteideas.net/sutton/book/ebook/the-book.html>, 6.5 Q-Learning: Off-Policy TD Control [↑](#endnote-ref-3)