The state is represented as the individual grids making up the map/state space for where the robot can travel to. The action is the direction of movement of the robot. The state transition probability is the probability that the robot will move from the current grid square to its desired square and has a probability of p, or one grid to either side of the desired direction with probability q=0.5\*(1-p).

A picture containing diagram

Description automatically generatedA picture containing chart

Description automatically generated

The reward function will be represented as follows:

**+C** for successfully visiting a cleaning site, this value is quite big as the robot should be encouraged to perform this task as often as possible

**+R** for visiting a recharge station. This value is relatively big since we do not want the robot to fully discharge on accident, but also don’t want it to spend majority of its time recharging.

**-COL** is the cost for collisions with objects, and luggage reclaim areas

**-S** is the cost for each step which should be minimized as higher step cost will not only make the movement of the robot inefficient, but also

**-T** is the cost for every period of time (second) passed

**-D** for discharged battery, and is a significantly larger value than other rewards

Finally, the discount factor will be a high value, close to 1 because the later into the job, monitoring the state of the battery becomes more critical.

Policy iteration will be applied to this problem in two steps, policy evaluation and policy improvement. This is because policy iteration’s computational costs are typically cheaper and has an execution time faster than value iteration, which is important as we want the robot to start cleaning as quickly and efficiently as possible while operating online.

Initially, the robot will have a policy of π0, a random sequence of movements. Then the robot will estimate how optimal this policy is using policy evaluation which is used to evaluate the state value function and is given by the equation: . A terminal state with value 0 () is therefore introduced for key locations such as the charging location, cleaning site, objects, luggage reclaim and discharged state so the robot knows when it has reached the goal location and can terminate its movement. However, because these states all have large rewards and the terminal state value must be 0, a virtual terminal state which exists outside the map is used and has a value of 0. To get to the virtual state, we have virtual terminal actions which are only available in these locations. The only action that can be performed from these locations is a virtual step to the terminal state and the rewards on these actions tell how good the final state is. Finally, we will also introduce a terminal condition θ which is very small (≥1\*10-6) so that once a close enough estimate for has been met, we can terminate the episode and the robot does not waste time and energy iterating further.

Following on from this, the robot then performs policy improvement to introduce a new policy π1. The formulation of a new policy is done by visiting every possible grid in turn and picking the action (the direction of movement of each grid) that maximises the state action value and is given by the equation: . If after propagating through the state space, no values in the policy changes, then the robot can assume it has the optimal solution, if not, the entire process repeats evaluating new policies with the key being the reward returned from the new policy is always greater than or equal to the one generated previously: and over a certain amount of iterations, the robot will converge on an optimal policy π\* with optimal value function vπ\*, travelling energy and time efficiently through the airport.