A Deterministic Improved Q-Learning for Path Planning of a Mobile Robot

AER1516 Paper presentation

Authors: Amit Konar, Senior Member, IEEE, Indrani Goswami Chakraborty, Sapam Jitu Singh, Lakhmi C. Jain, and Atulya K. Nagar

Presented by: Zihan Huang, Alejandro Solares, Swapnil Patel

March 8th, 2023

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Table of Contents

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Introduction

- Traditional machine learning algorithms emphasize the need for supervised learning to train a robot to determine its next state in a given map using the sensor readings.
  - However, the acquired knowledge is no longer useful if there's a small change in the robot's environment.
- Reinforcement learning might be an alternative solution to this type of motion planning problem due to its inherent power of automatic learning even in the presence of small changes in the world map.

Goals of the paper

To prove that the improved Q-learning algorithm could outperform the existing Classical Q-Learning and Extended Q-learning. To demonstrate its potential to be applied in motion planning of mobile robots, particularly when obstacles are added in real time.

UNIVERSITY OF TORONTO

Related Work

- Classical Q-learning(CQL) [2] [3]

Classical Deterministic Q-Learning
For each S, a initialize $Q(S, a) = 0$;
Observe the current state S;
Repeat
Select $a \in \{a_1, a_2, \ldots, a_m\}$ and execute it;
Receive an immediate reward $r(S, a)$;
Observe the new state $S^{/} \leftarrow \delta(S, a)$;
Update the table entry $Q(S, a)$ by

$$Q(S, a) = r(S, a) + \gamma \operatorname*{Max}_{a^{/}} Q(\delta(S, a), a^{/});$$

$S \leftarrow S^{/}$;
For ever.

- Extended Q-learning(EQL)[1]

The extension offers an additional benefit of avoiding unnecessary computations involved to update the Q-table. A flag variable is used to keep track of the necessary updating in the entries of the Q-table. EQL was developed in their previous paper.

- Improved Q-learning(IQL)

An improvement on top of the EQL.

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Classical Q-learning — State action value function Q

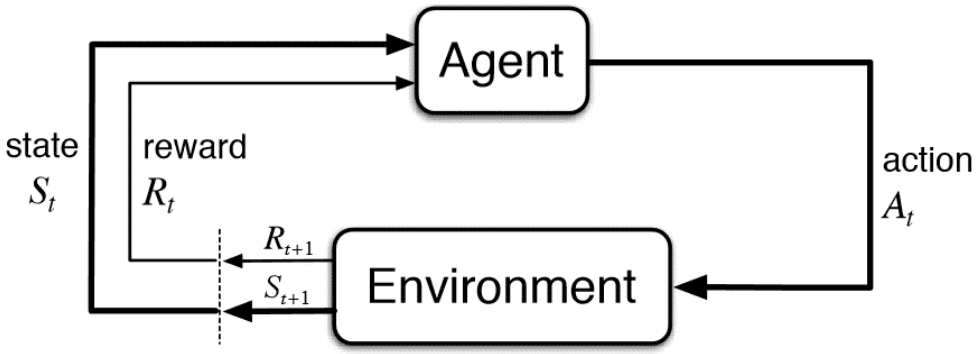


Figure: 4 basic components in Reinforcement Learning; agent, environment, reward and action.[4]

- Given a policy π, Q is the expected return G_t, given that you're at state S_t, and you take action A_t. $Q_\pi(s,a) = E_\pi\left[G_t \mid S_t = s, A_t = a\right]$, where $G_t = R_{t+1} + \gamma R_{t+2} + \nu^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots..$
- By bootstrapping, using dynamic averaging, and using off-policy RL, the Q learning update rule is obtained: $Q\left(S_t, A_t\right) = Q\left(S_t, A_t\right) + \alpha\left[R_{t+1} + \gamma \max_a Q\left(S_{t+1}, a\right) - Q\left(S_t, A_t\right)\right]$, where γ is the discount factor and α is the learning rate.

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Classical Q learning — How do we improve the policy

	action 1	action 2	action 3
state a	0.2	0.1	0.6
state b	0.5	0.8	0.9
state c	0.8	0.7	0.3

Exploration

Exploration usually refers to selecting any action with non-zero probability in every encountered state to learn the environment by the agent.

Exploitation

Exploitation, on the other hand, is targeted at employing the current knowledge of the agent to expect achieving good performance by selecting greedy actions

Classical Q learning — Algorithm explained

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

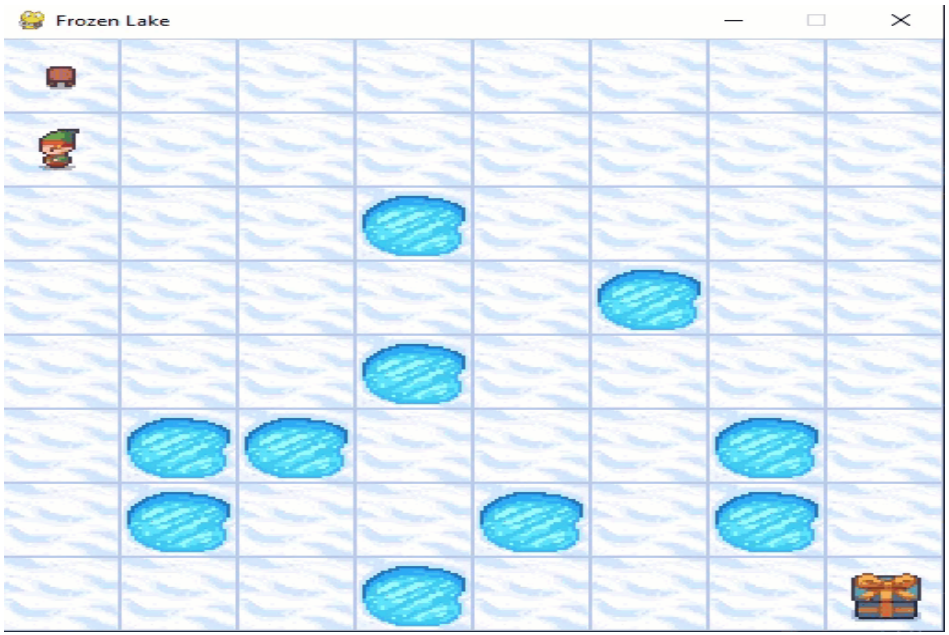
Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

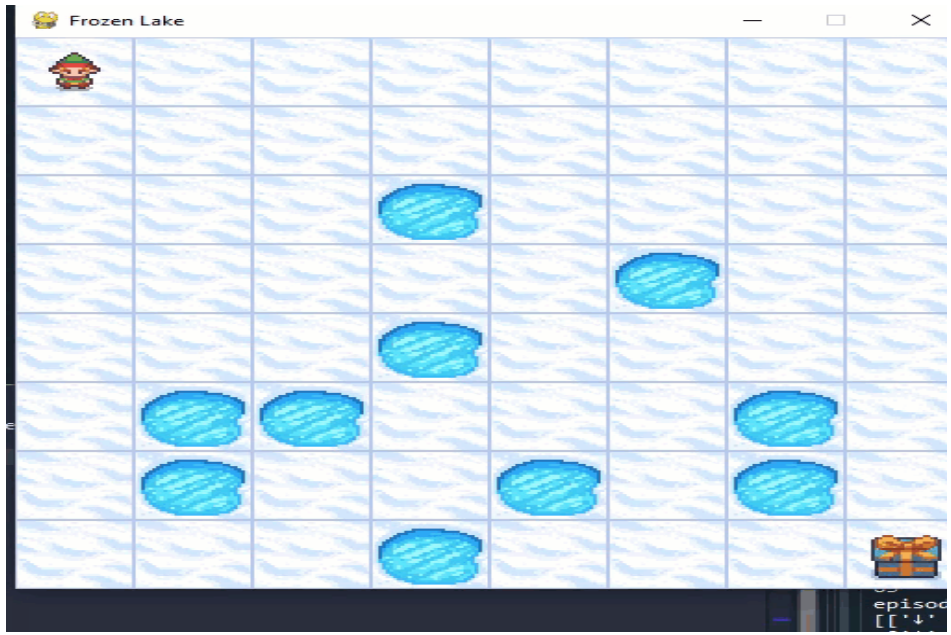
Figure: Q-learning algorithm

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Implementation - Classical Q learning



(a) Q learning Training



(b) Q learning Path planning

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Improved Q-Learning (IQL)

Definition

Let S_p, S_n, and S_G be the present, the next, and the goal states.
Let Q_p and Q_n be the Q-value at the present and next states S_p and S_n.
Let d_{xy} be the city block distance between the states S_x and S_y.

L_x is a boolean variable lock to indicate that the Q_x value of a state is fixed permanently.

We set lock $L_n = 1$ if the Q-value of the state n is fixed and will not change further after L_n is set to 1 . The Lock variable for all states except the goal will be initialized to zero in our proposed Q-learning algorithm.

- Property 1: If $L_n = 1$ and $d_{pG} > d_{nG}$, then $Q_p = \gamma \times Q_n$ and set $L_n = 1$.
- Property 2: If $L_p = 1$ and $d_{nG} < d_{pG}$, then $Q_n = Q_p/\gamma$ and set $L_n = 1$.
- Property 3: If $L_p = 1$ and $d_{nG} > d_{pG}$, then $Q_n = \gamma \times Q_p$ and set $L_n = 1$.
- Property 4: If $L_n = 1$ and $d_{nG} > d_{pG}$, then $Q_p = Q_n/\gamma$ and set $L_p = 1$

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Space and Time Complexity reduction

- Space Complexity:
  - CQL: Q table will have $(m \times n)$ dimension.
  - IQL:Two storages are required for each state, one for storing the Q-value and the other for storing the value of the lock variable of a particular state.
  - Reduction: Saving in Space complexity: $mn - 2n = n(m-2)$, which is of the order of mn.
- Time complexity:
  - CQL:Requires $n(m-1)$ comparisons
  - IQL: n
  - Reduction: $n(m-1) - n = nm - 2n = n(m-2)$, which is of the order of mn.

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Pseudocode for IQL

Pseudocode for IQL

1. Initialization

For all S_i, $i = 1$ to n, except $S_i = S_G$

{set $L_i = 0$; $Q_i = 0$;}

L_G(for goal S_G) $= 1$;

Q_G(for goal S_G) $= 100$;

Assign γ in (0, 1) and initial state $= S_p$;

2. **Repeat**

{

Select a_i from $A = \{a_1, a_2, \ldots, a_m\}$ and execute it;

} **Until** $S_P = S_G$;

3. Update *Q-table*:

Repeat

{

a) Select a_i from $A = \{a_1, a_2, \ldots, a_m\}$;

b) Determine d_{nG} and d_{pG};

If$(d_{nG} < d_{pG})$

Then if$(L_n = 1)$

Then if$(L_p = 0)$

Then $\{Q_p = \gamma \times Q_n; L_p = 1;\}$

Else if$(L_p = 1)$

Then $\{Q_n = Q_p/\gamma; L_n = 1;\}$

Else if$(L_p = 1)$

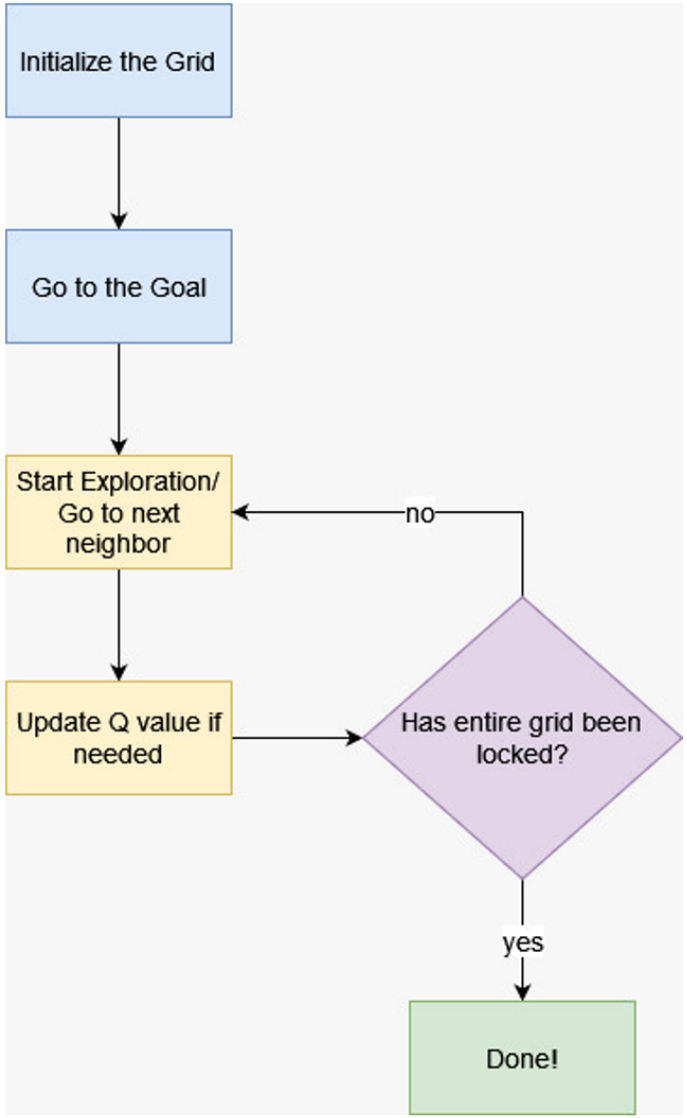
Then if$(L_n = 0)$

Then $\{Q_n = \gamma \times Q_p; L_n = 1;\}$

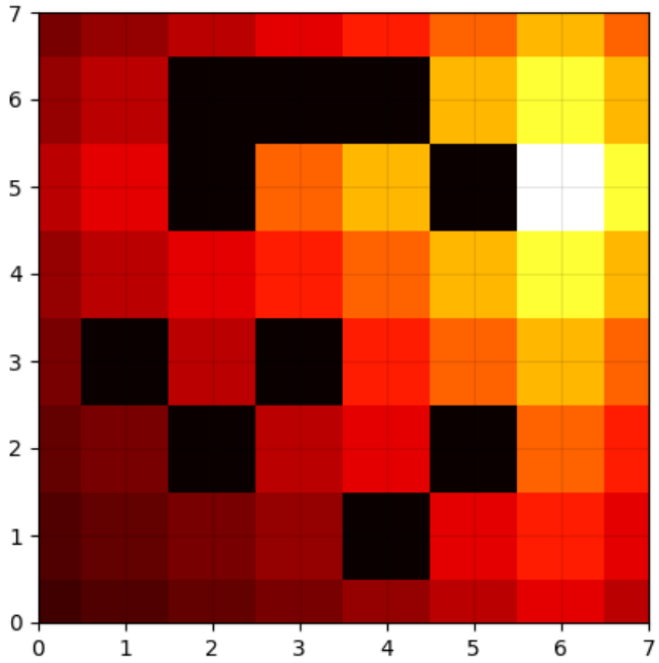
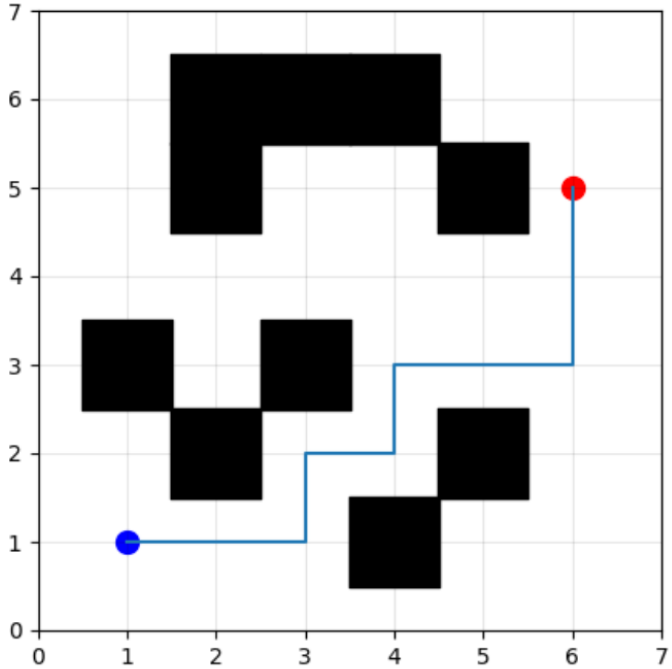
Else if$(L_n = 1)$

Then $\{Q_p = Q_n/\gamma; L_p = 1;\}$

} **Until** $L_i = 1$ for all i without obstacle;



Implementation - Improved Q learning

Q Table	Path Planning
	

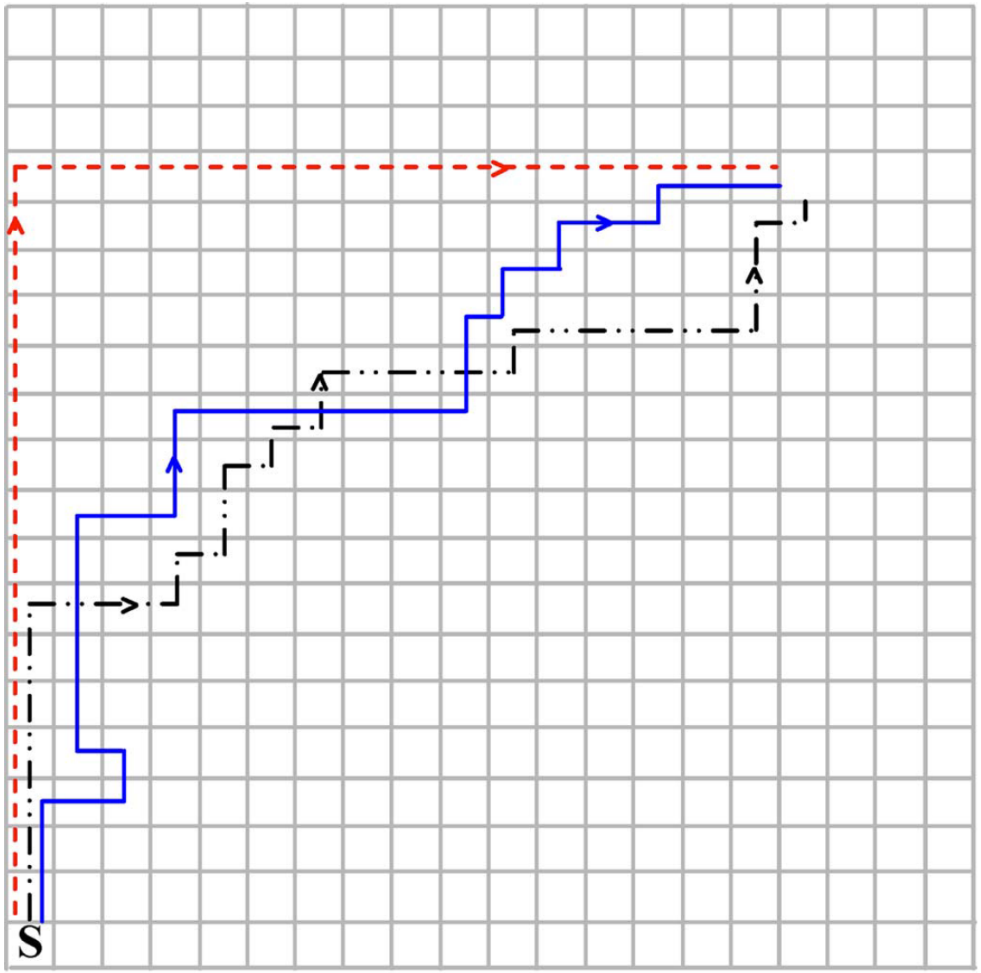
Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Results and Discussion

- Simulation Setup
  - Environment of 20 x 20 grids
  - A given world map is trained by the Q-learning
  - Trained world map with a Known Q table is used to generate a trajecotry
- Experiments
  - Experiment 1: Carry out each Q-learning algorithm on the world map of 20x20 grids to compare the performance
  - Experiment 2: Train in the obstacle-free environment.Then add obstacles to the map and change the starting position.
  - Experiment 3: Execute the algorithm in a world map with obstacles during both the training and planning phases.
  - Experiment 4: Train in the map with 5 obstacles, and plan trajectory with 3 additional obstacles.

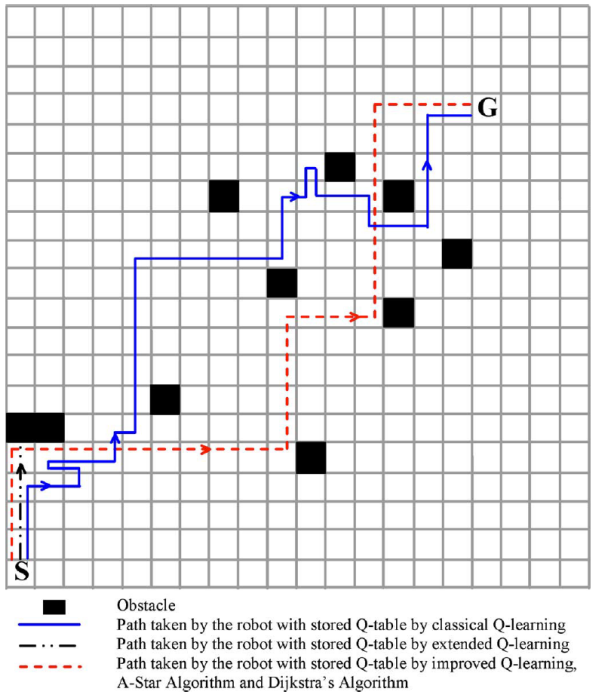
Experiment 1

Carry out each Q-learning algorithm on the world map of 20x20 grids to compare the performance



Experiment 2

Train in the obstacle-free environment.Then add obstacles to the map and change the starting position.



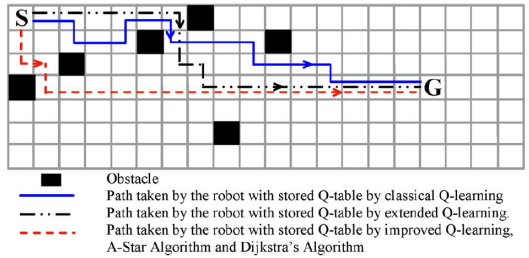
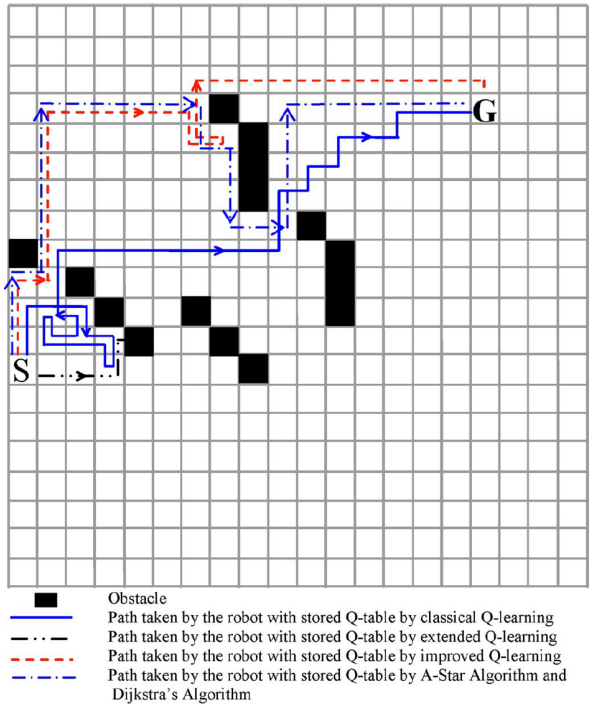


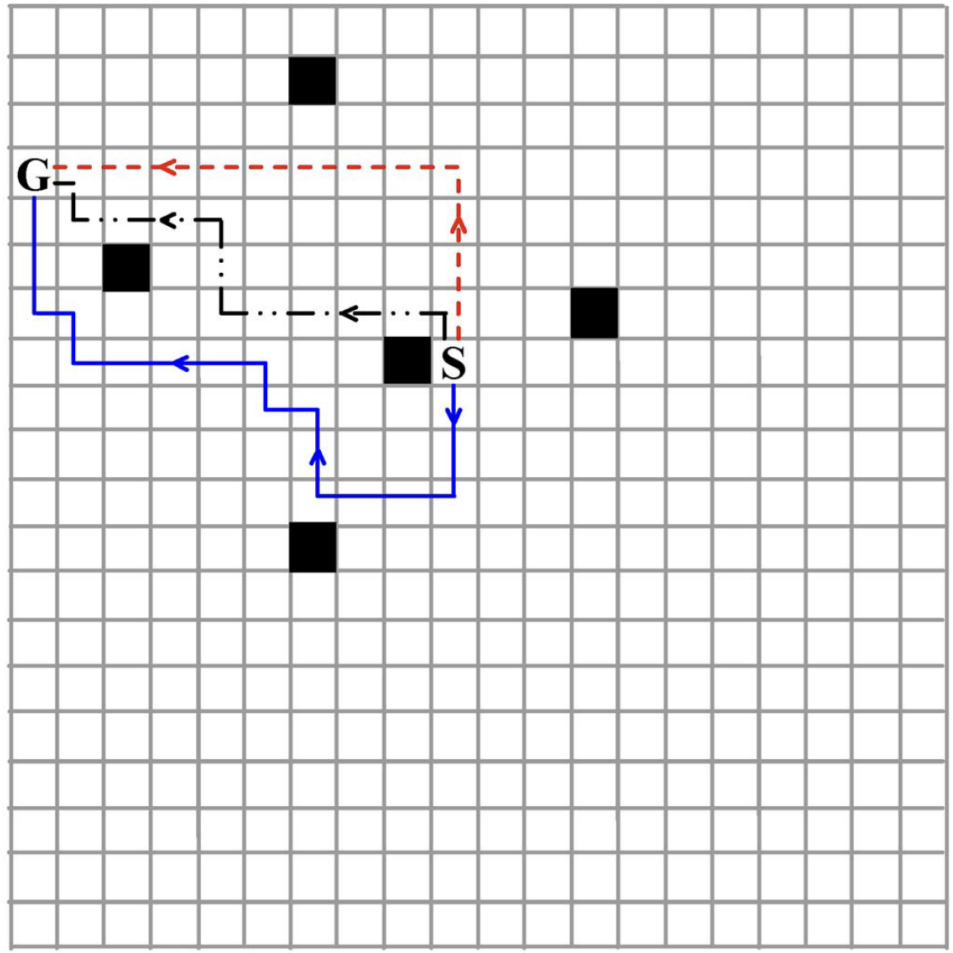
Fig. 3. Partial World map 3 with obstacles. It shows the paths taken by the robot with the stored Q-table of different algorithms.



Institute for Aerospace Studies
UNIVERSITY OF TORONTO

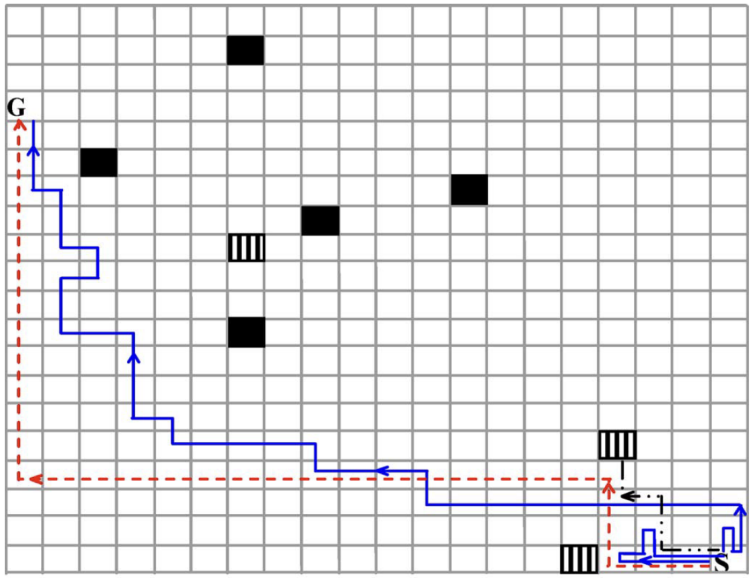
Experiment 3

Execute the algorithm in a world map with obstacles during both the training and planning phases.

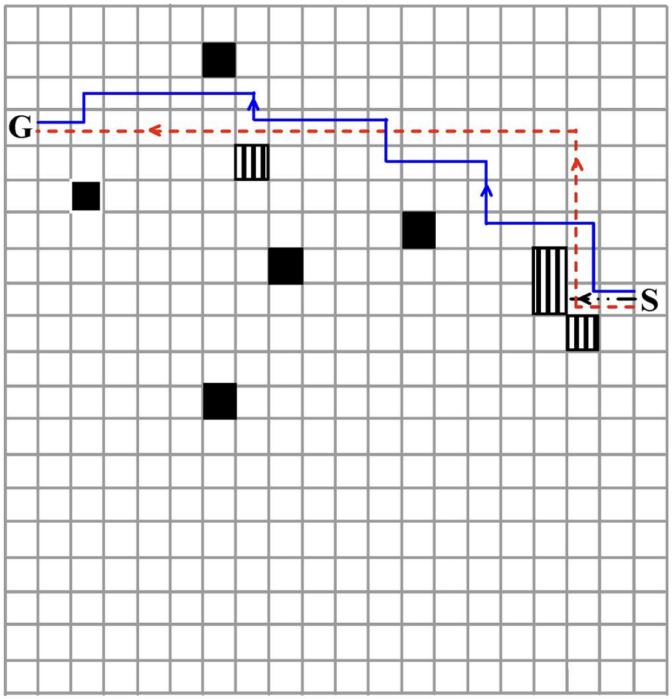


Experiment 4

Train in the map with 5 obstacles, and plan trajectory with 3 additional obstacles.



New obstacle added
Obstacle present
Path taken by the robot with stored Q-table by classical Q-learning
Path taken by the robot with stored Q-table by extended Q-learning
Path taken by the robot with stored Q-table by improved Q-learning, A start and Dijkstra's Algorithm

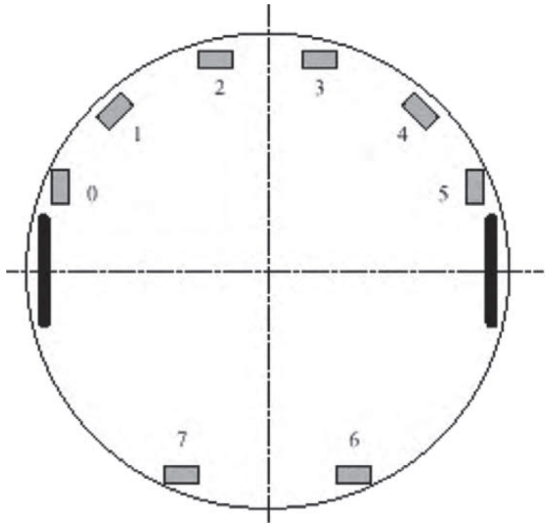


New obstacle added
Obstacles present
Path taken by the robot with stored Q-table by classical Q-learning
Path taken by the robot with stored Q-table by extended Q-learning
Path taken by the robot with stored Q-table by improved Q-learning, A-Star Algorithm and Dijkstra's Algorithm

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Real Wolrd Experiment Setup on Khepera platform

- Onboard microprocessor: Motorola 68331 25Mhz processor 512KB flash memory
- Robot equipped with 8 infrared proximity sensors, and 2 encoders .

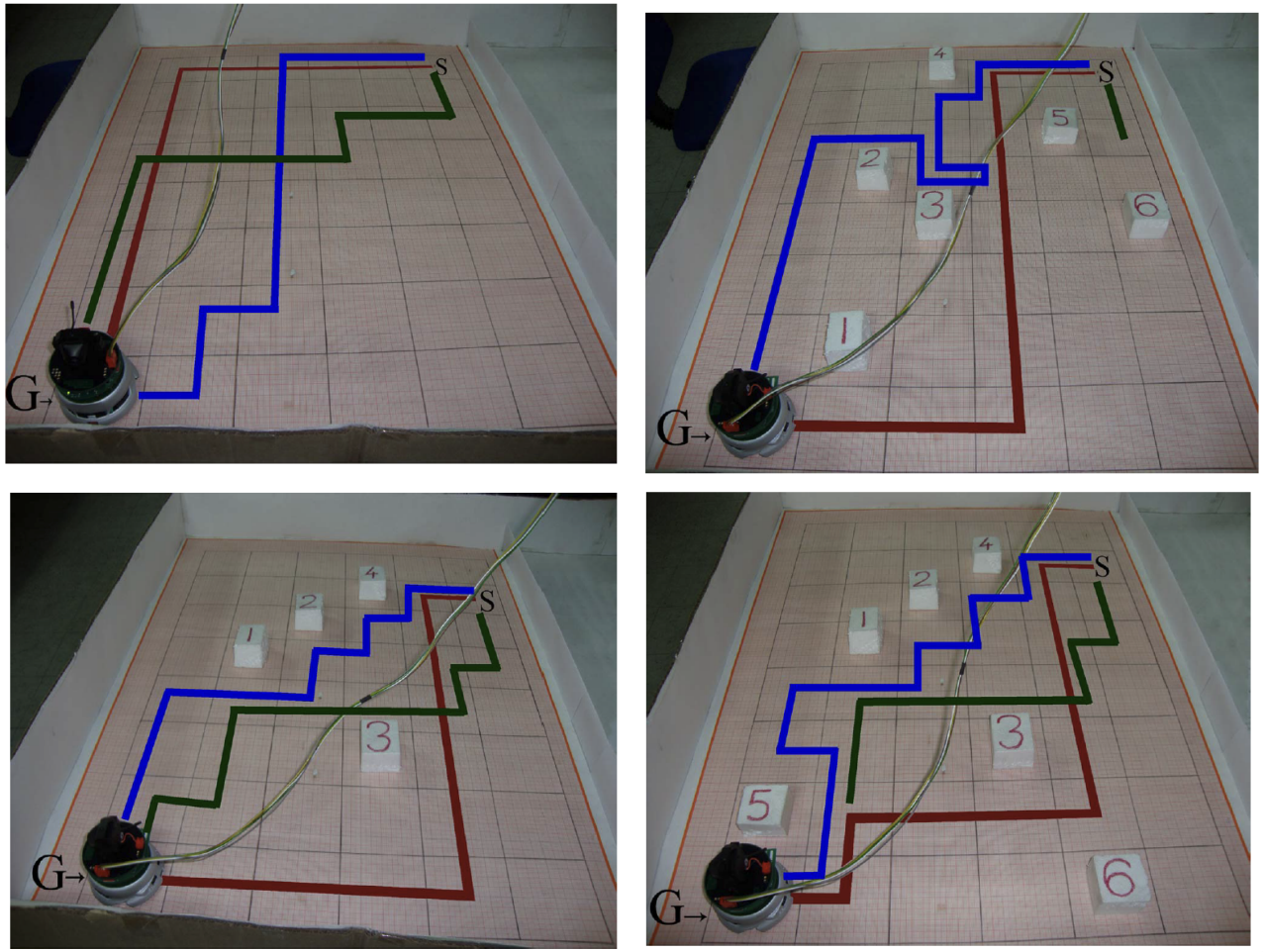


- Starting and goal states S and G are marked in experimental maps
- Experiment 1: A real-world map without obstacles
- Experiment 2: Q-learning and path planning in the same world map, add 6 rectangular obstacles after the learning phase.
- Experiment 3: Learning and planning in the world map with 4 obstacles
- Experiment 4: Learning and planning in the world map with 4 obstacles, add 2 more obstacles.

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Experiment Results on Real World map

Red line - IQL; Green Line - CQL; Blue line - EQL;



Experiment Results on Real World map

TABLE II

COMPARISON OF TIME TAKEN BY THE ROBOT, NUMBER OF 90° TURNS REQUIRED BY THE ROBOT, AND NUMBER OF STATES TRAVERSED BY THE ROBOT

Fig. No	Time Taken in sec			No. of 90° turn			No. of states traversed		
	IQL	*EQL*	*CQL*	*IQL*	*EQL*	*CQL*	*IQL*	*EQL*	*CQL*
13	40.73	48.44	47.40	1	5	4	12	12	12
14	43.74	-	71.17	2	-	9	12	-	16
15	39.62	48.40	48.40	2	7	7	11	11	11
16	43.72	-	59.52	4	-	10	11	-	13

IQL Improved Q-learning
EQL Extended Q-learning
CQL Classical Q-learning
- Goal cannot be reached

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Conclusion

- This paper proposes an alternative RL algorithm that results in a significant saving in time complexity compared to the classical Q-learning. Experiments were performed both in simulation and the Khepera platform to prove that IQL outperforms CQL and EQL in all three metrics. As such, it is concluded that IQL has good potential in the path planning applications of mobile robots.
- Future works: Should be using a neural network to replace the Q table, Deep Q learning. Other papers:
- Only comparing IQL against CQL, not comparing it to the state of art RL algorithm like deep Q learning.

Conclusion

Pros:

- Paper well organized and Math well explained.
- Good number of Simulations and real-world experiments.
- Explained reduction in time and space complexity.
- The reduction in space and time complexity would only be beneficial in embedded systems with low storage and computing resources.

Cons:

- Little details on how the real world experiments are conducted.
- Discrepancy on A* computation time compared to other papers.
- EQL was added on purpose to demonstrate how badly it performed. EQL was developed in their previous paper.
- IQL is more closely aligned to A* or Dijkstra's heuristic-based algorithm, not traditional RL algorithms.

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Extra Resources and Work Distribution

- Alejandro: Implementing the Classical Q learning and providing plots and animation.
- Swapnil: Implementing the Improved Q learning and providing plots and animation.
- Zihan: Put together the contents in this slide, and deliver the presentation.
- Additional links and resources:
  - `https://colab.research.google.com/drive/1E2RViy7xmor0mhqskZV14_NUj2jMpJz3`
  - `https://github.com/Swapnil949/AER1516_Robot_Motion_Planning_paper`

References

- Indrani Goswami (Chakraborty) et al. "Extended Q-Learning Algorithm for Path-Planning of a Mobile Robot". In: *Simulated Evolution and Learning*. Ed. by Kalyanmoy Deb et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 379–383. ISBN: 978-3-642-17298-4.
- Pradipta KDas et al. "An Improved Q-learning Algorithm for Path-Planning of a Mobile Robot". In: *International Journal of Computer Applications* 51 (Aug. 2012), pp. 40–46. DOI: 10.5120/8073-1468.
- Avinash Sharma et al. "Model based path planning using Q-Learning". In: *2017 IEEE International Conference on Industrial Technology (ICIT)*. 2017, pp. 837–842. DOI: 10.1109/ICIT.2017.7915468.
- Richard S. Sutton, Francis Bach, and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT Press Ltd, 2018.

Institute for Aerospace Studies
UNIVERSITY OF TORONTO