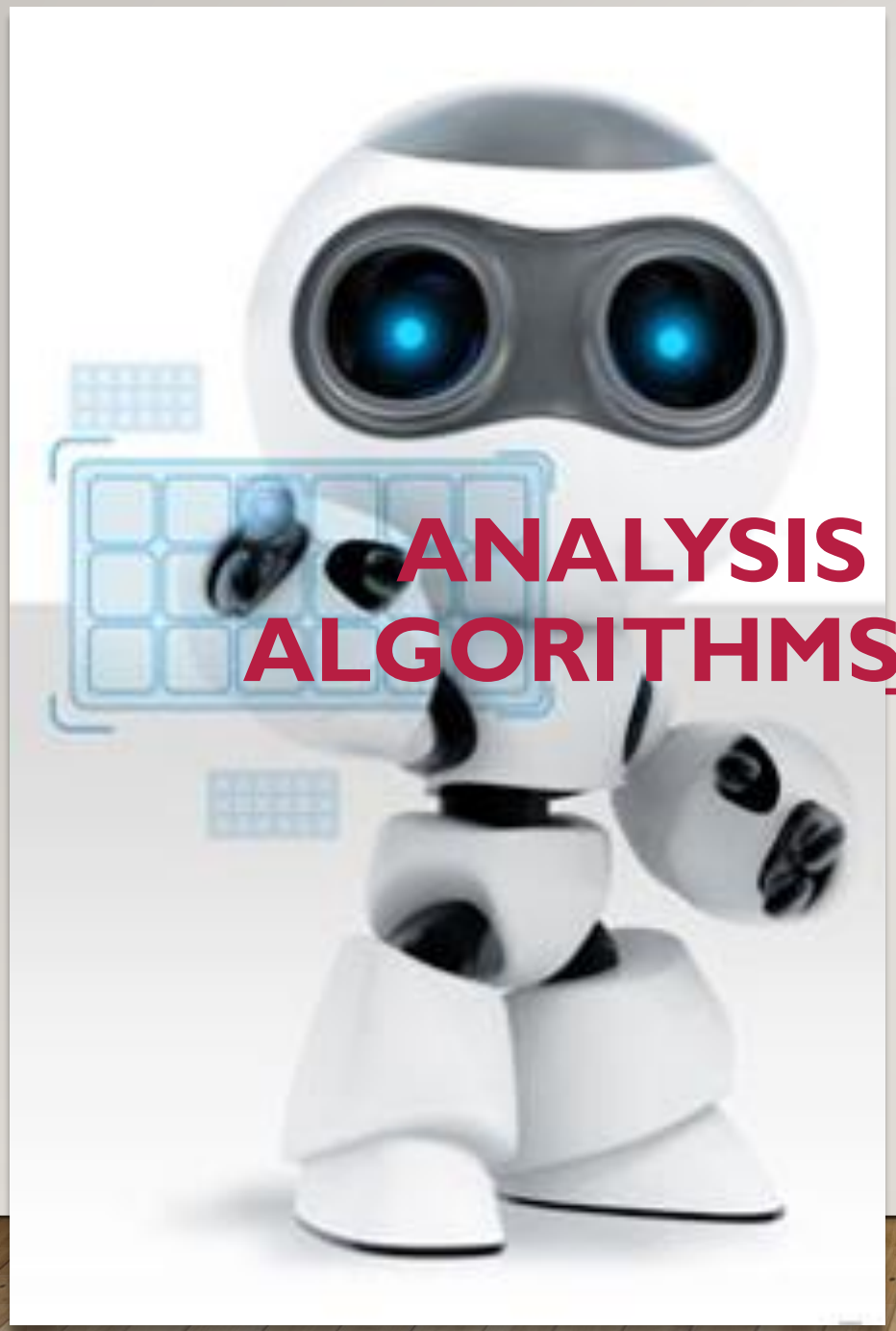


ANALYSIS OF REINFORCEMENT LEARNING ALGORITHMS FOR ROBOTIC PATH PLANNING

Dr. P.SUPRIYA

SUPERVISOR

Associate Professor

Electrical & Electronics Engineering

Amrita Vishwa Vidyapeetham

Coimbatore

LAYA HARWIN

CB.EN.P2EBS17014

MTech 2nd Year

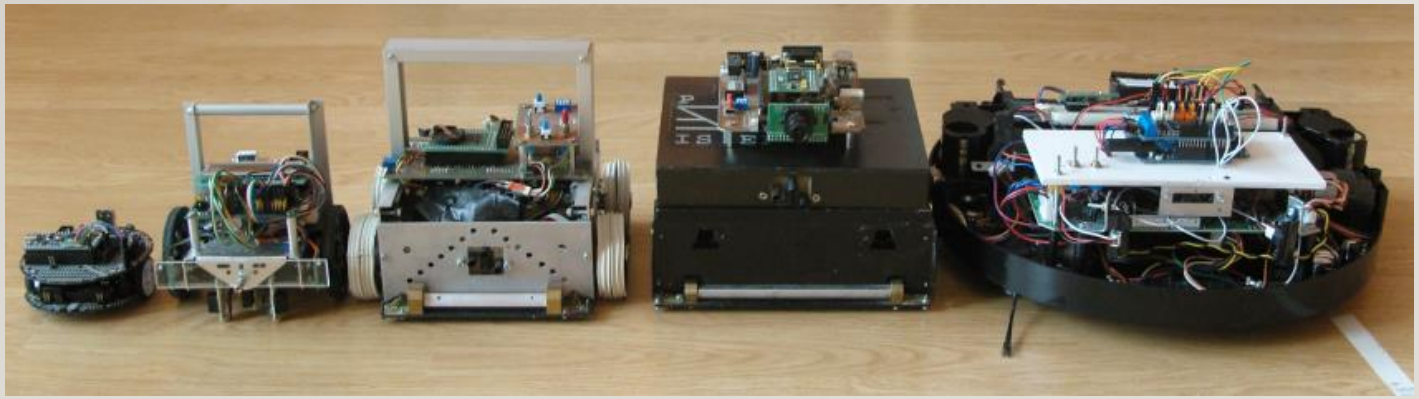
Electrical & Electronics Engineering

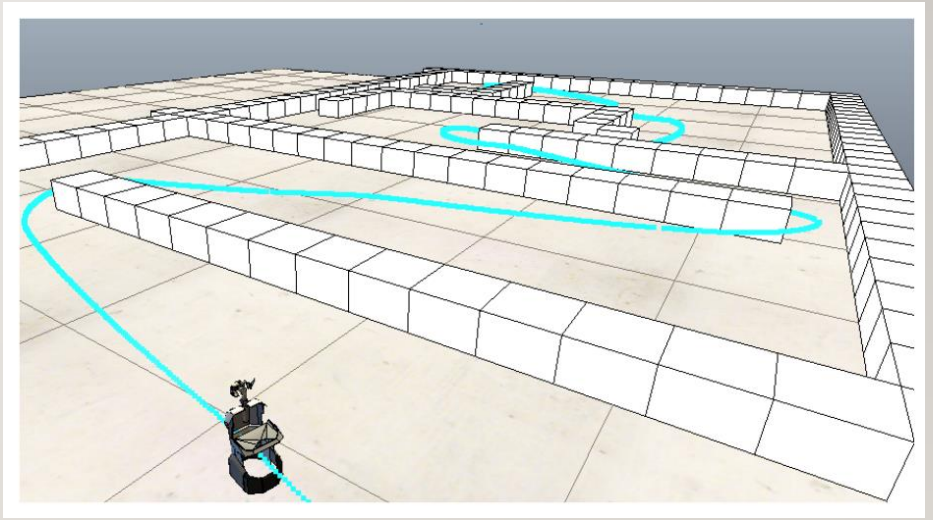
Amrita Vishwa Vidyapeetham

Coimbatore

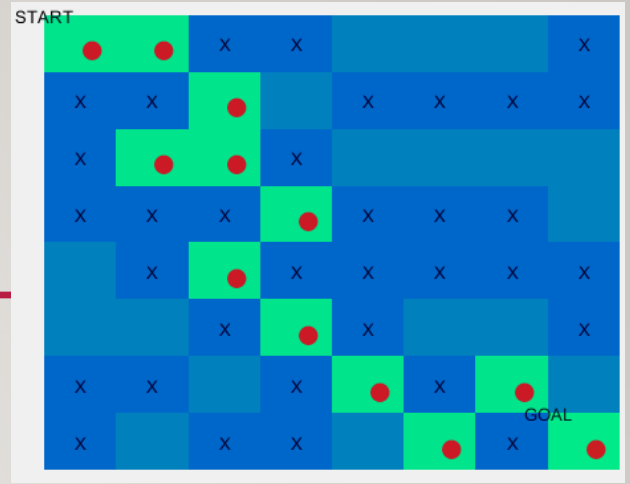
INTRODUCTION

- ✓ Automation is turning into a vital part of human day to day life.
- ✓ Robotics is a versatile domain with various areas open.
- ✓ Robotic path planning can be done by various techniques.
- ✓ Path planning is an important requirement for autonomous mobile robots



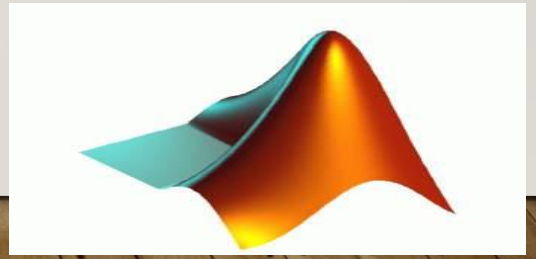


OBJECTIVE



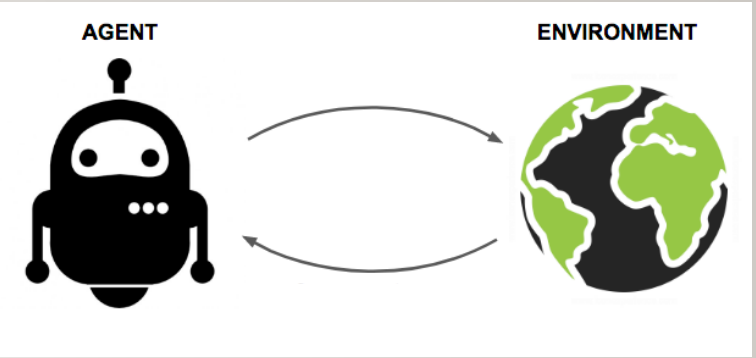
- The main objective of this project is to find the shortest path between source and destination with optimal path planning using reinforcement learning algorithms in an indoor environment for static obstacles and dynamic obstacles.
- Simulating the same in MATLAB and Python IDLE
- Implementation of this algorithm using an iRobot Create interfaced to NXP LPC1768 Cortex M3 controller.

python™



AMRITA VISHWA VIDYAPEETHAM

REINFORCEMENT LEARNING



- **WHAT IS RL ALGORITHM?**

✓ Type of Machine Learning, and also a branch of AI.

✓ Allows machines and software agents-determines ideal behaviour-maximize its performance.

✓ Simple reward feedback -agent to learn its behaviour- known as the reinforcement signal

✓ agent learns by trial-and-error method

✓ Punishes for wrong action-Agent decides best action to select based on his current state.

- **WHY RL ALGORITHM?**

✓ Learn its behaviour based on feedback from the environment.

✓ Can be learnt once and for all, or keep on adapting as time goes by.

✓ If the problem is modelled with care, some RL algorithms can converge to the global optimum.

✓ Little need for a human expert who knows about the domain of application.

✓ Much less time will be spent designing a solution-no complex sets of rules

RELATED WORK : SARSA ALGORITHM

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
H. Iima and Y. Kuroe	*2008*	Swarm reinforcement learning algorithms based on Sarsa method	*SICE Annual Conference*, Tokyo, pp. 2045-2049	• Individual learning and learning through exchanging information • Multiple agents
C. Li, M. Wang, S. Yang and Z. Zhang,	*2009*	Urban Traffic Signal Learning Control Using SARSA Algorithm Based on Adaptive RBF Network	*International Conference on Measuring Technology and Mechatronics Automation*, Zhangjiajie, Hunan, pp. 658-661.doi: 10.1109/ICMTMA.2009.445	• By training self-adapted non-linear processing unit • Realizing online and adaptive constructing of state space • The approximation is improved and thus the control of traffic signal at single intersections is solved
N. Lilith and K. Dogancay	2005	Distributed reduced-state SARSA algorithm for dynamic channel allocation in cellular networks featuring traffic mobility	*ICC* Seoul, 2005, pp. 860-865 Vol. 2. doi: 10.1109/ICC.2005.1494473	• Can achieve superior new call and handoff blocking probabilities. Capable of producing call blocking probabilities that are comparable to those obtained by the centralised learning agent.
M. R. Tousi, S. H. Hosseinian, A. H. Jadidinejad and M. B. Menhaj,	2008	Application of SARSA learning algorithm for reactive power control in power system	*IEEE 2nd International Power and Energy Conference*, Johor Bahru, pp. 1198-1202. doi: 10.1109/PECON.4762658	• Able to provide optimal or near optimal control settings for power system under varying system conditions.

RELATED WORK : REINFORCEMENT LEARNING

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
Zhang, Qian et al	2012	Reinforcement Learning in Robot Path Optimization	*JSW* 7 : 657-662	• Aims at Markovian decision process • Single robot's path planning, static environment based on Qlearning • Describes the application of this algorithm on the path planning by setting off state space and action space reasonably and designing reinforcement function.
D. P. Romero-Martí, J. I. Núñez-Varela, C. Soubervielle-Montalvo and A. Orozco-de-la-Paz,.	2016	Navigation and path planning using reinforcement learning for a Roomba robot	*XVIII Congreso Mexicano de Robotica*, Sinaloa, pp. 1-5	• Robot is provided with a topological map of a building floor (environmental map). • Using this map the robot learns a path from one location to another by means of reinforcement learning.

RELATED WORK : REINFORCEMENT LEARNING

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
Shreyas J and Sandeep J	2016	Modern Machine Learning Approaches For Robotic Path Planning	IJCSIT:256-259	• Discusses about the various path planning algorithms • Machine learning algorithms which emerged in the past decade and which were continuously improvised and were perfected to have a very high success rate. • Working of the algorithms and how these algorithms have developed and improvised over time can be seen.

RELATED WORK : PATH PLANNING

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
P. Joshy and P Supriya	2016	Implementation of robotic path planning using Ant Colony Optimization Algorithm	*ICICT*, Coimbatore, pp. 1-6	• Mobile robot-shortest route source-destination • ACO single robot, static obstacles • limitation -to find the optimal path • To avoid- more robots by inter robot communication.
D. Davis and P. Supriya	2016	Implementation of Fuzzy-Based Robotic Path Planning	ICETECH	• Fuzzy Logic-obstacle avoidance-unknown environment • Input- distance from obstacle (L,F and R) and angle • Fuzzy rules-to control velocity of L and R wheels of the robot • Accurate-uses more inputs-more precise.
J. K. Goyal and K. S. Nagla,	2014	A new approach of path planning for mobile robots	*ICACCI*, New Delhi, pp. 863-867. doi: 10.1109/ICACCI.2014.6968200	• modified A* algorithm-to find a most safe path for mobile robots of particular dimension. • Virtual obstacle is used to enhance the size of obstacles in respect to the size of the mobile robot.
M. Lin, K. Yuan, C. Shi and Y. Wang	2017	Path planning of mobile robot based on improved A∗ algorithm	*CCDC*, Chongqing, pp. 3570-3576. doi: 10.1109/CCDC.2017.7979125	• Improved A* algorithm based on the introduction of the parent node and the change of the heuristic cost weight is proposed to control the path planning of intelligent robots

REINFORCEMENT LEARNING ALGORITHMS

TEMPORAL DIFFERENCE LEARNING ALGORITHM

- **Equation: Q_[s_, a_] = R[a_] + r[s,a] +γ * max(Q_[s_, a_])**
- **Doesn't interact with the environment**
- **Take the action with maximum value**
- **More concentrated on the optimal path and not on the consequences**

SARSA ALGORITHM

- **Equation: Q_[s_, a_] = Q[s, a] + λ * (r + γ * (Q_[s_, a_]) – Q[s, a])**
- **Interacts with the environment**
- **Takes the actual path**
- **SARSA will tend to avoid a dangerous optimal path**

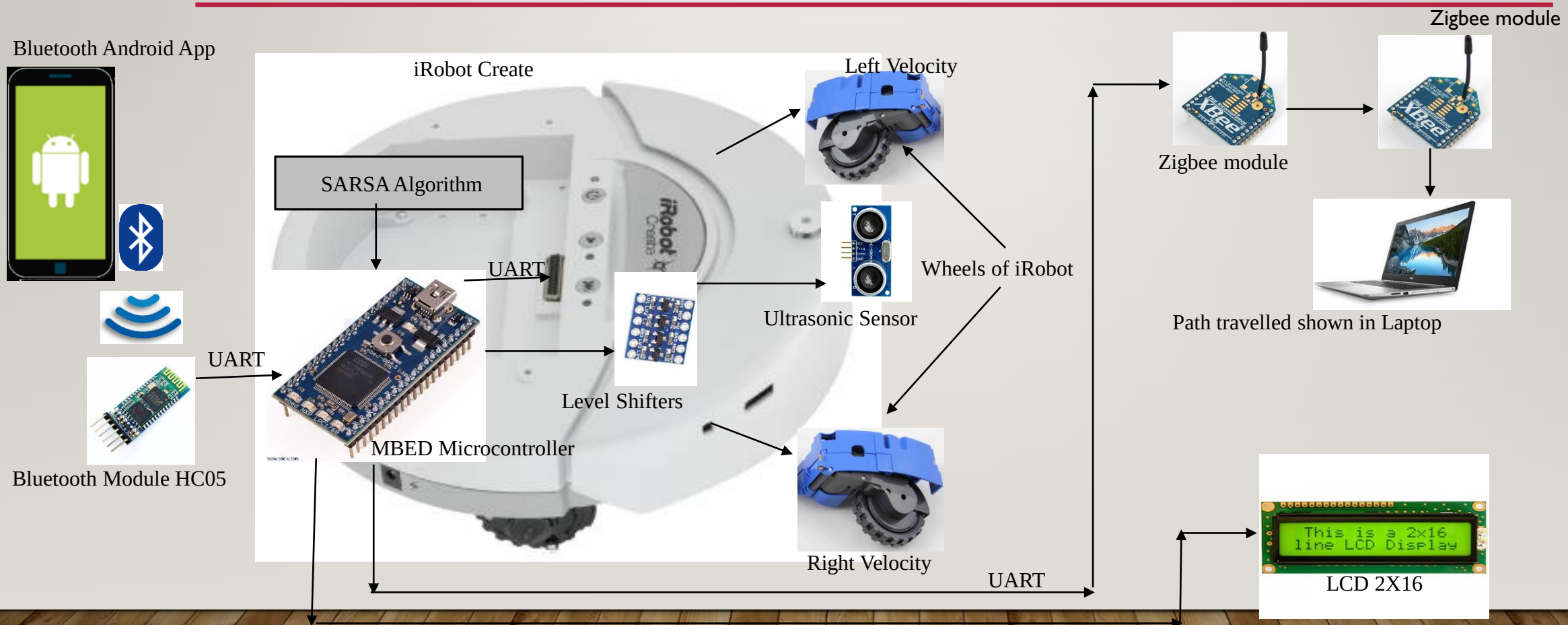
Q LEARNING ALGORITHM

- **Equation: Q_[s_, a_] = Q[s, a] + λ * (r + γ * max (Q_[s_, a_]) – Q[s, a])**
- **Interacts with the environment**
- **Take the action with maximum value**
- **More concentrated on the optimal path and not on the consequences**

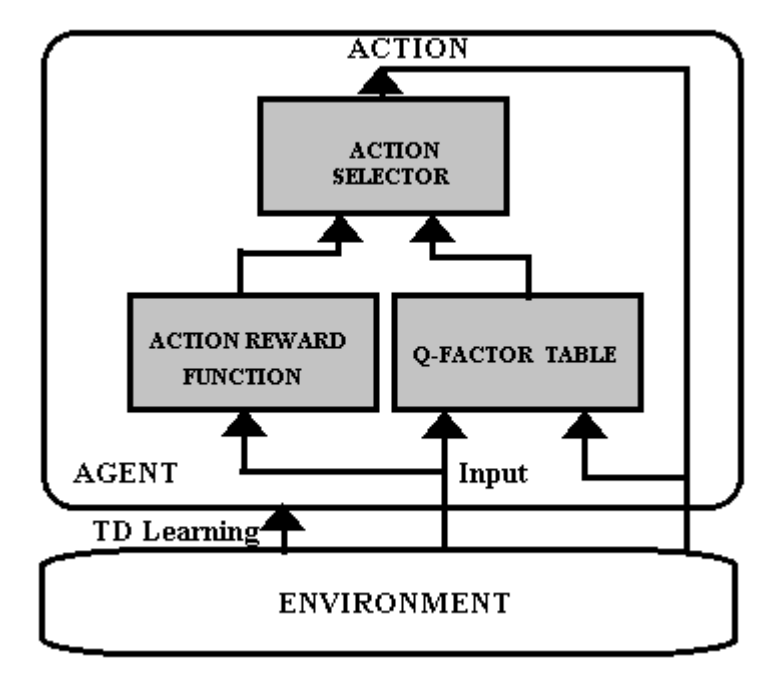
DEEP Q NETWORK ALGORITHM

- **Equation: Q_[s_, a_] = Q[s, a] + λ * (r + γ * max(Q_[s_, a_]) – Q[s, a]) +Neural Networks**
- **Interacts with the environment**
- **Take the action with maximum value**
- **More concentrated on the optimal path and not on the consequences**

BLOCK DIAGRAM FOR SARSA ALGORITHM



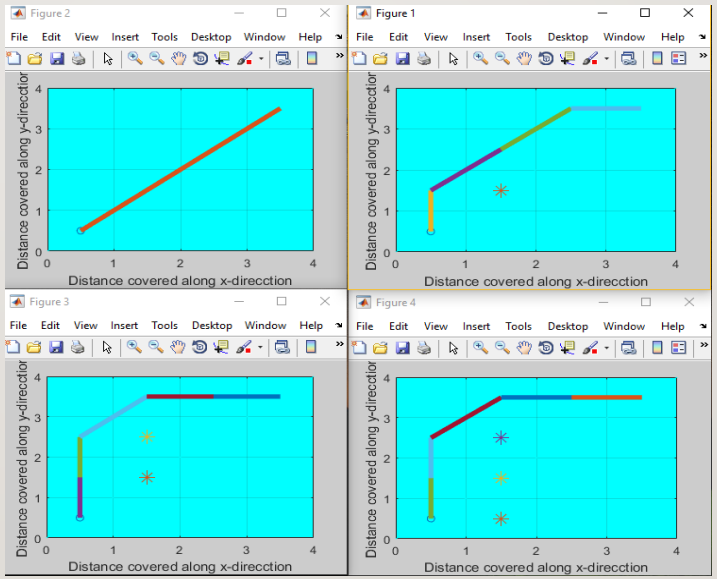
ALGORITHM 1 : TEMPORAL DIFFERENCE LEARNING ALGORITHM

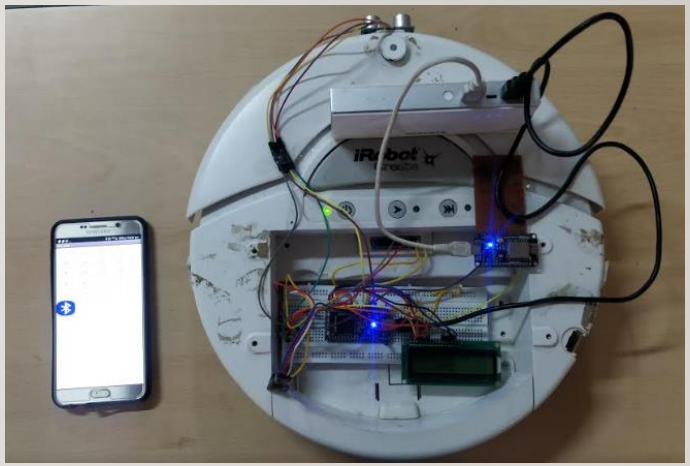


- **TD Learning algorithm calculates the Q factor value for each possible movement from each state for the given environment and inputs like starting point, destination point and obstacle position.**
- **A Reward function assigns reward for each possible action from current state based on the priority of movement and obstacle positions.**
- **The action selector will provide the action to be carried next from the current state by analysing both Q-factor and Reward function.**

IMPLEMENTATION OF TEMPORAL DIFFERENCE LEARNING ALGORITHM

- **TD algorithm was implemented for a 4x4 a with different source points in MATLAB R2018b**
- **Also, compared with conventional algorithms like A* and Dijkstra in MATLAB R2018b**
- **The size of the grid was extended for 5x5 and 6x6 and also compared with conventional algorithms in MATLAB R2018b**
- **It was then implemented in iRobot interfaced with Mbed microcontroller for a 4x4 grid with one dynamic obstacle**





ALGORITHM 2 : SARSA ALGORITHM (STATE ACTION REWARD NEXT STATE NEXT ACTION)

- ✓ **Uses <state, action, reward, state, action> experiences rather than the <state, action, reward, state> used by Q-learning.**
- ✓ **Instead of looking for the best action at every step, it evaluates the actions suggested by the current policy**
- ✓ **Uses this info to revise it**
- ✓ **On-policy learning learns the value of the policy being followed.**
- ✓ **e.g., act greedily 80% of the time and act randomly 20% of the time**

begin

initialize $Q[S, A]$ arbitrarily

observe current state s

select action a using a policy based on Q

repeat forever:

carry out an action a

observe reward r and state s'

select action a' using a policy based on Q

$Q[s, a] \leftarrow Q[s, a] + \alpha\,(r + \gamma Q[s', a'] - Q[s, a])$

$s \leftarrow s'$;

$a \leftarrow a'$;

end-repeat

end

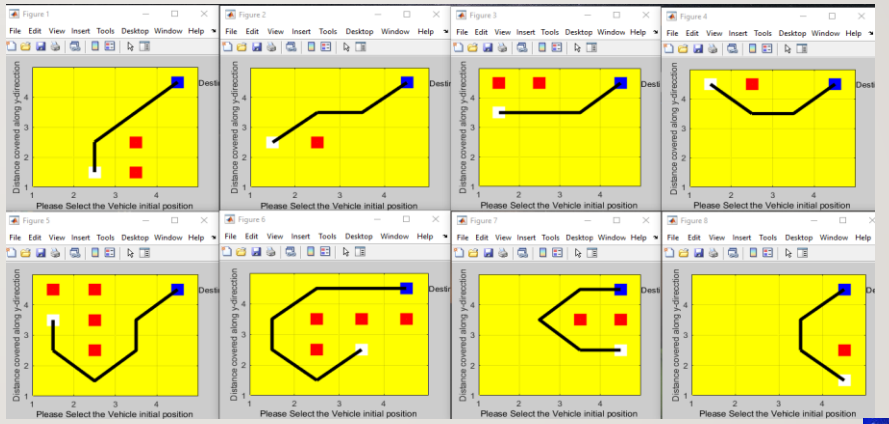
This could be, for instance any ε-greedy strategy: -Choose random ε times, and max the rest

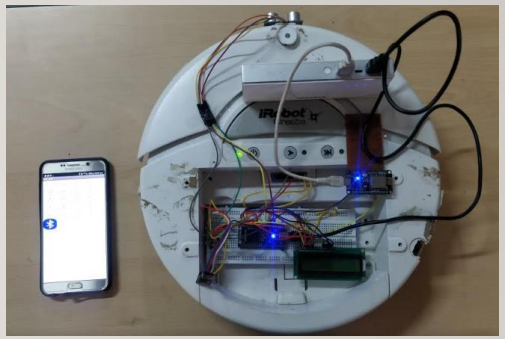
If the random step is chosen here, and has a bad negative reward, this will affect the value of Q[s,a].

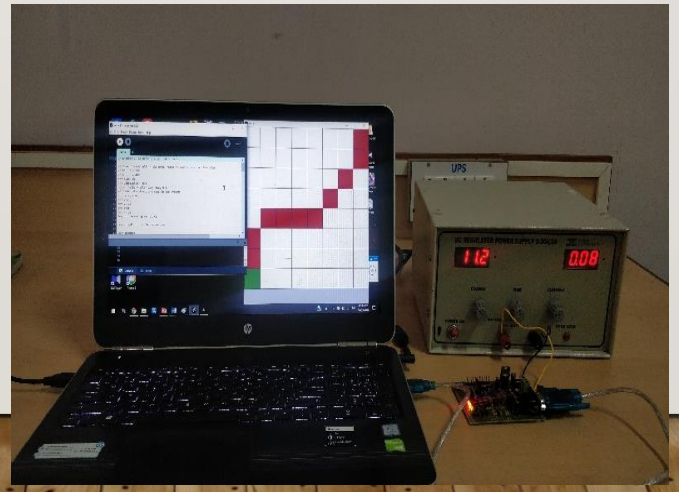
Next time in *s*, *a'* may no longer be the action selected because of its lowered Q value

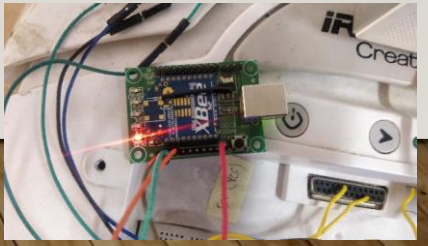
IMPLEMENTATION OF SARSA (STATE ACTION REWARD NEXT STATE NEXT ACTION) ALGORITHM

- **SARSA algorithm was implemented in MATLAB R2018b from 4x4 to 20x20 a with altering source and destination points**
- **It is implemented to solve complex mazes with more than 100 obstacles**
- **It was then implemented in iRobot for a 4x4 grid with one dynamic obstacle**
- **The size of grid is extended to 8x8 with more than one dynamic obstacles with a GUI displaying the path travelled by the robot in a PC.**







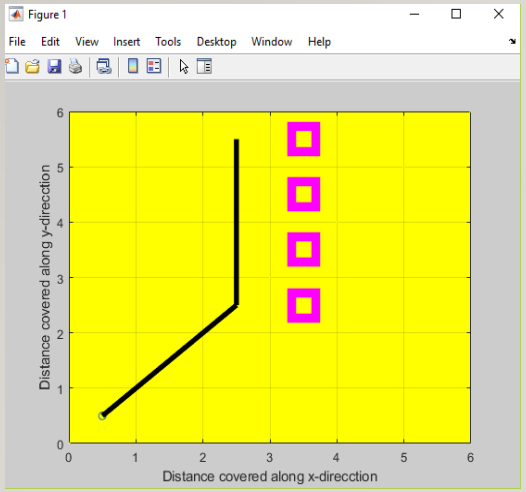


COMPARISON OF SARSA ALGORITHM AND TEMPORAL DIFFERENCE LEARNING ALGORITHM FOR ROBOTIC PATH PLANNING FOR STATIC OBSTACLES

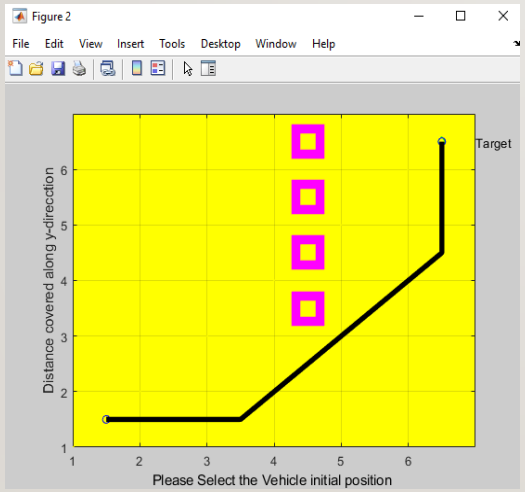
- **Comparison based on Success rate**

❑ **Cases where SARSA works and TD fails**

CASE 1:

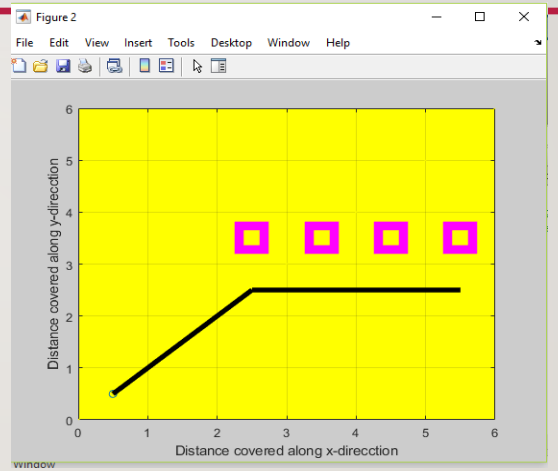


Case where TD fails

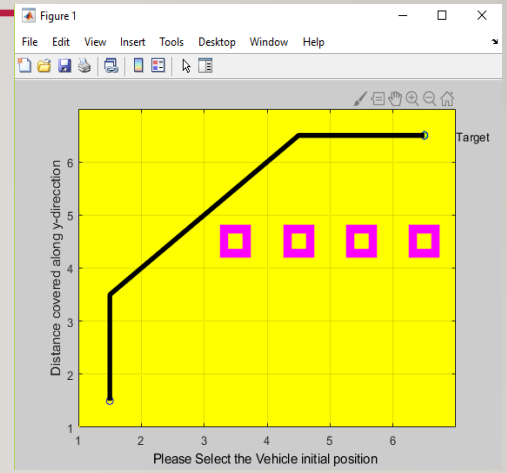


Case where SARSA works

CASE 2:

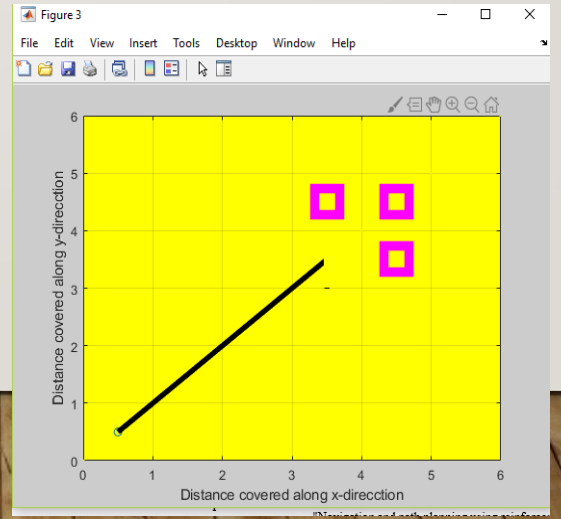


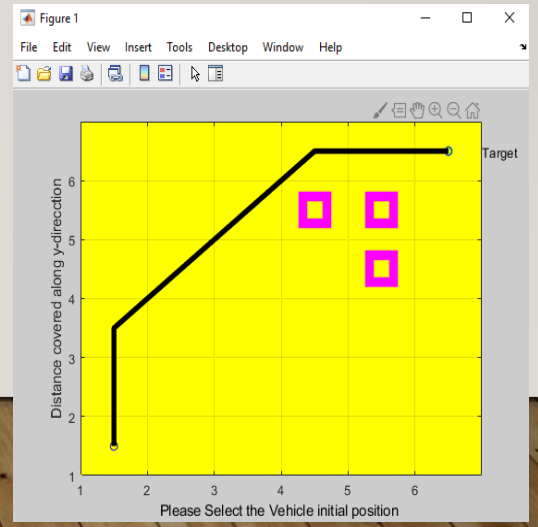
Case where TD fails



Case where SARSA works

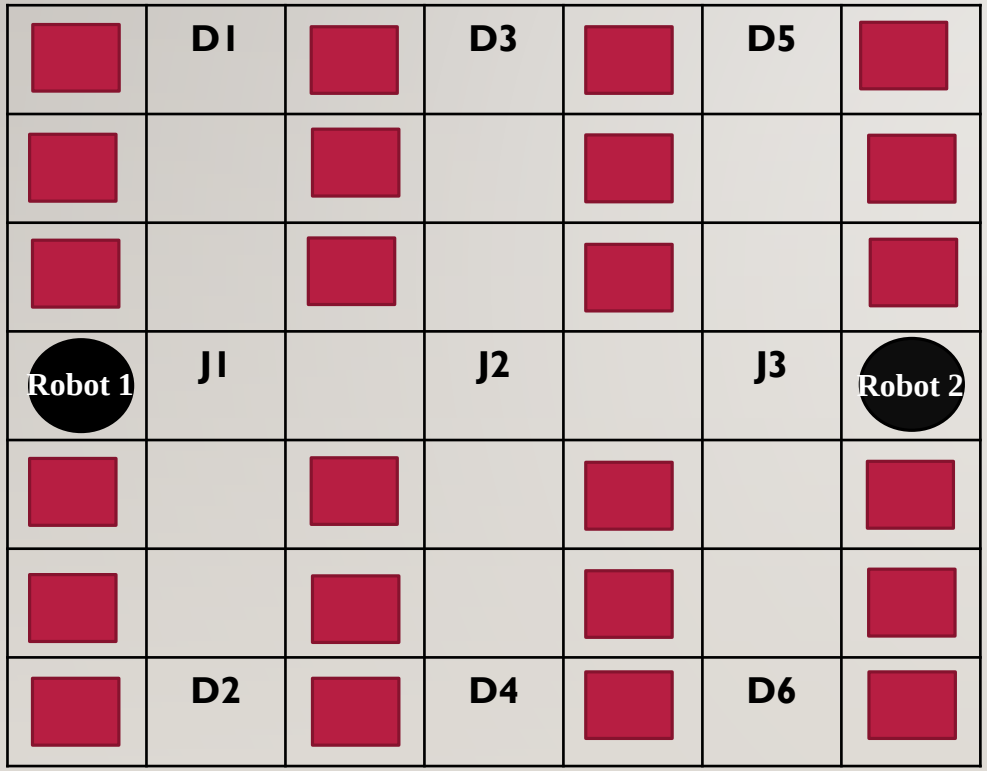
CASE 3:



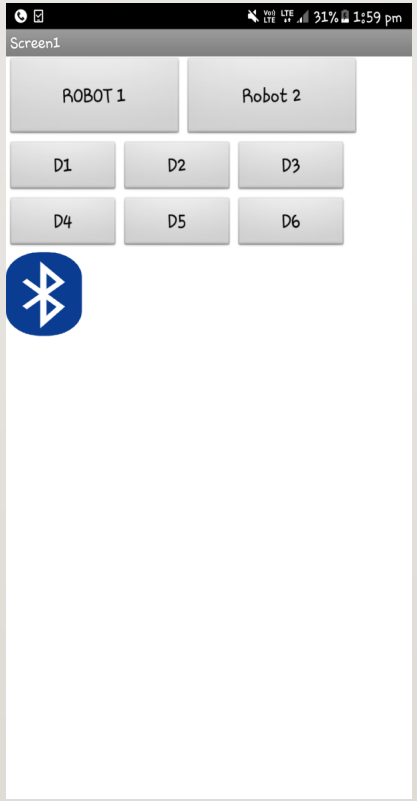


APPLICATION 1 : ROBOT TO ROBOT COMMUNICATION USING WI-FI

a. System Layout



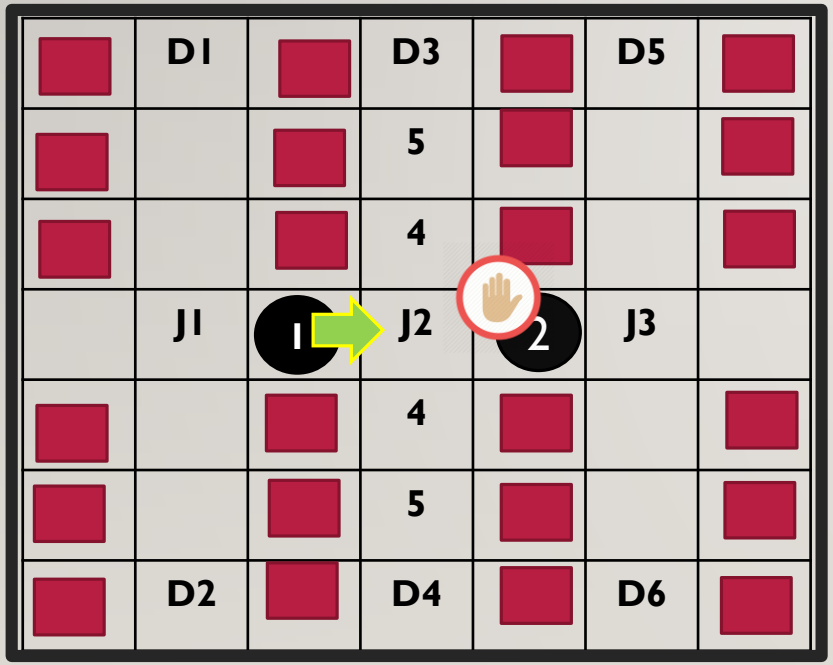
b. App



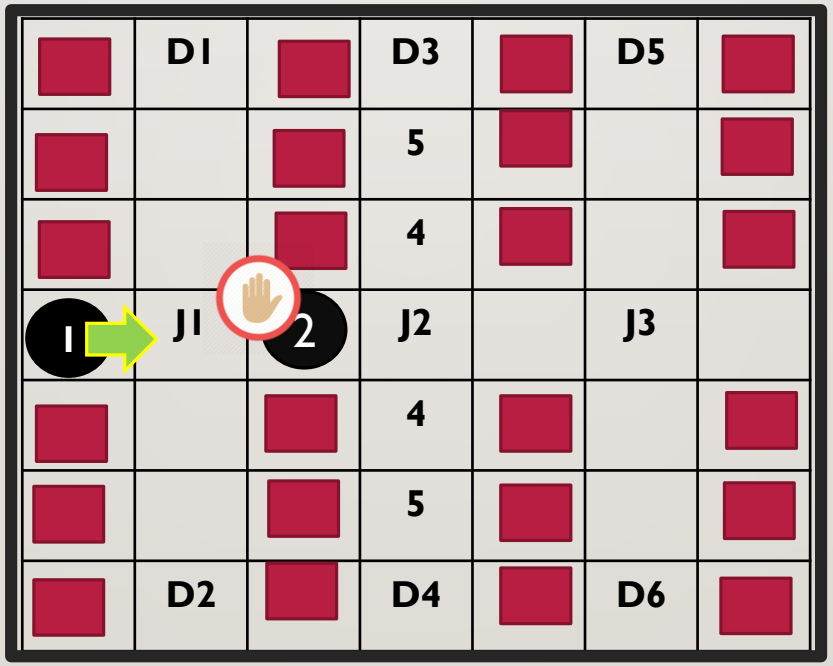
- 7x7 Grid
- Two iRobots- Robot 1 and Robot 2
- Three Junctions- J1, J2 and J3
- 6 destinations- D1, D2, D3, D4, D5 and D6
- Obstacles -
- Possible paths do not have obstacles
- Rest all the grids are assumed to have obstacles
- Communicated to one another using Wi-Fi
- Path planning is done with SARSA algorithm

IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI

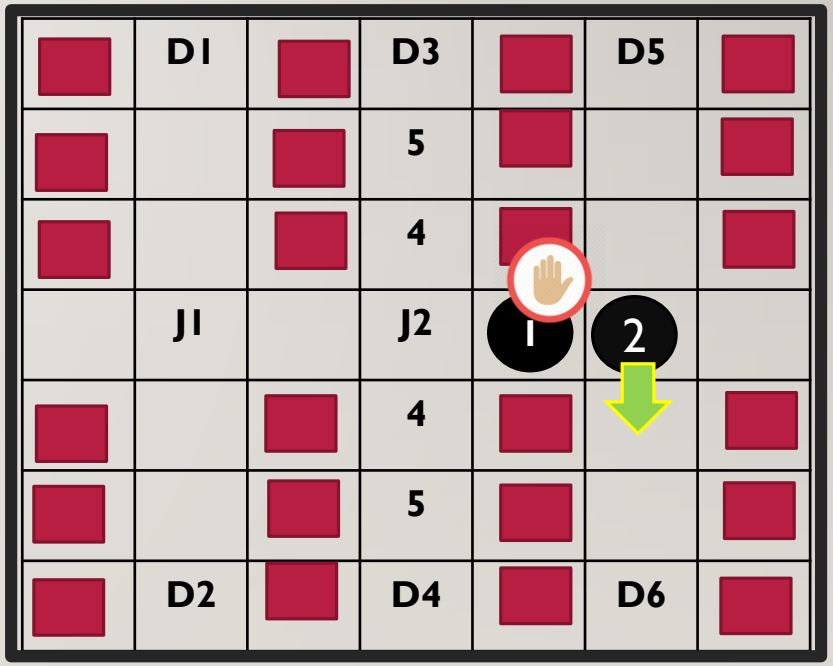
a. Case 1 where R1 goes to D3 and R2 goes to D4



b. Case 2 where R1 goes to D2 and R2 goes to D1

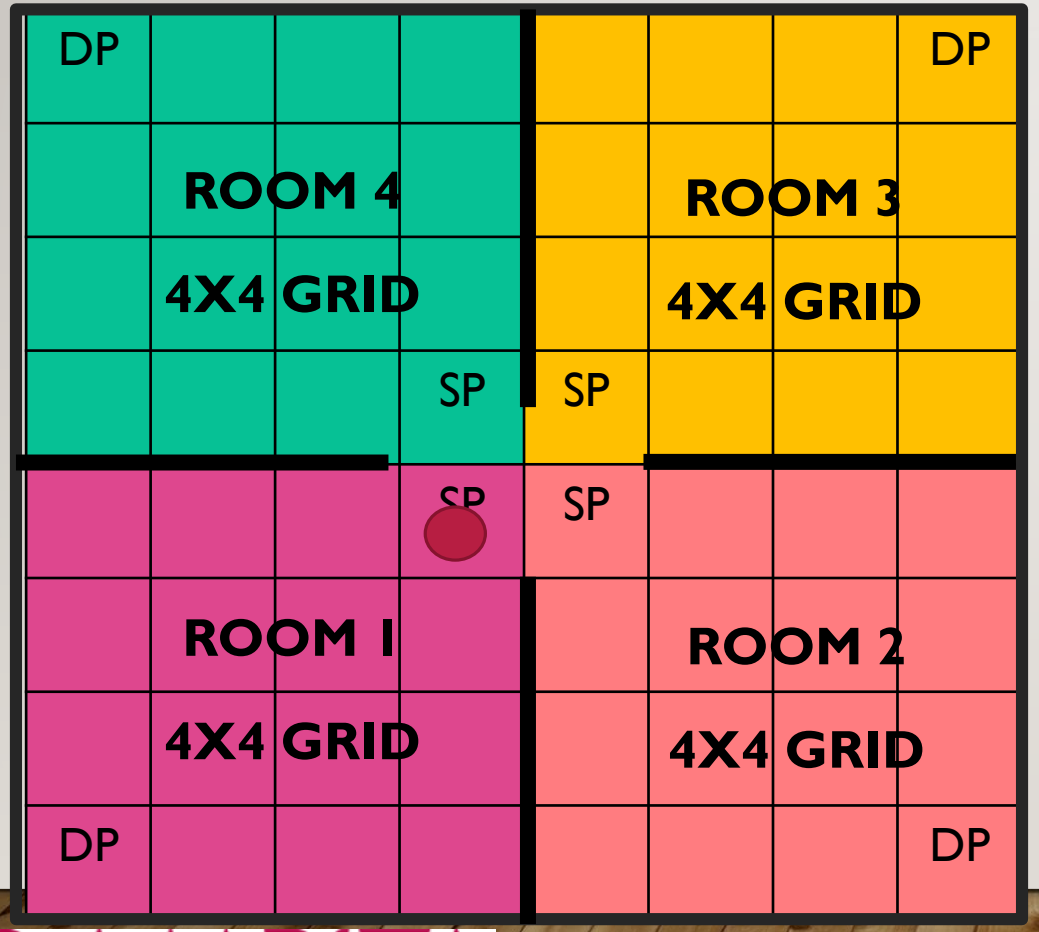


c. Case 3 where R1 goes to D5 and R2 goes to D6

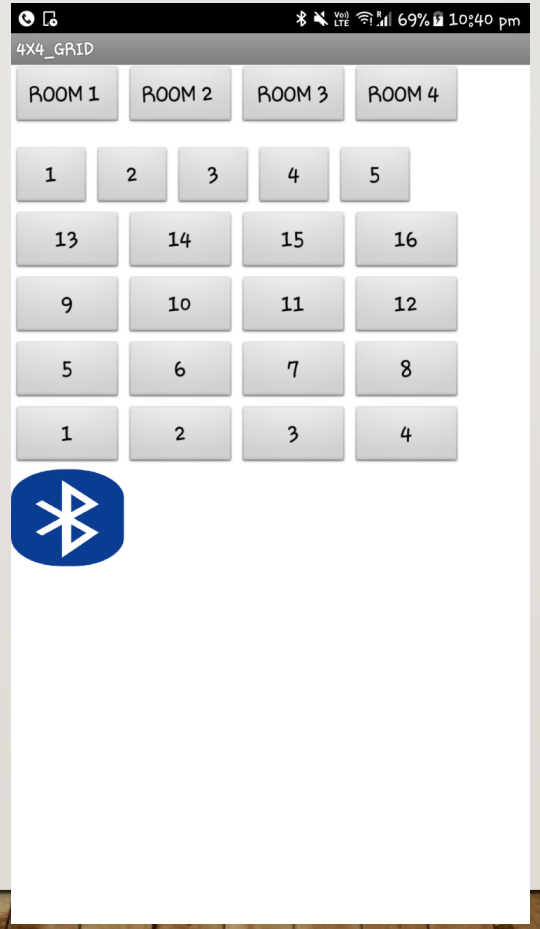


APPLICATION 2 : LOAD CARRYING ROBOT IN AN INDUSTRY

a. System Layout



b. App

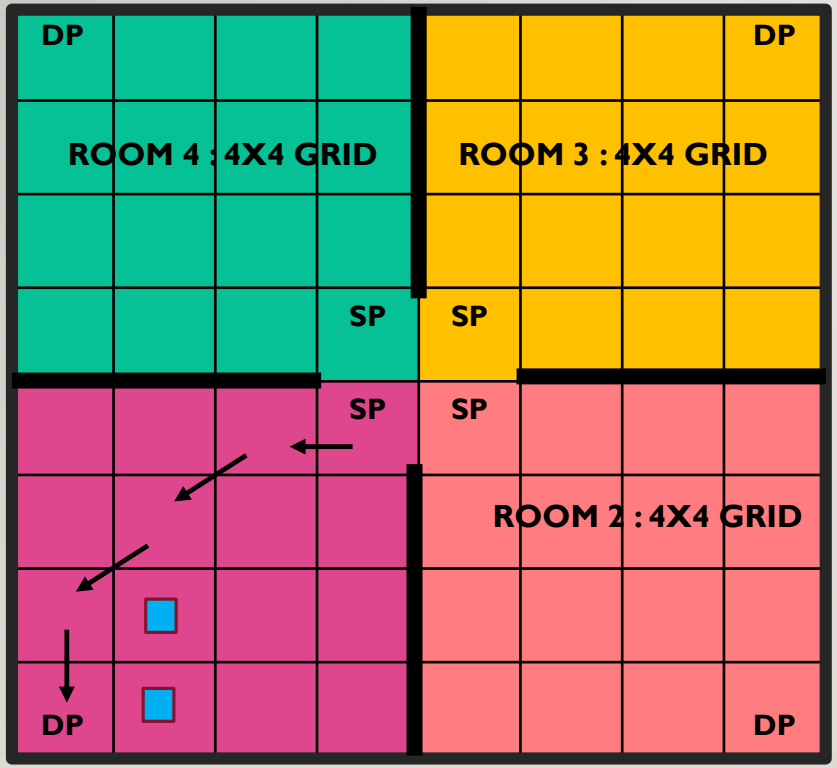


- **SP- Source point**
- **DP- Destination point**
- **4 rooms with 4X4 grid size**
- **Load carrying to all the room starts from the source point of Room 1**
- **App for communicating the room number, position of obstacles and number of obstacles to the robot**

IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

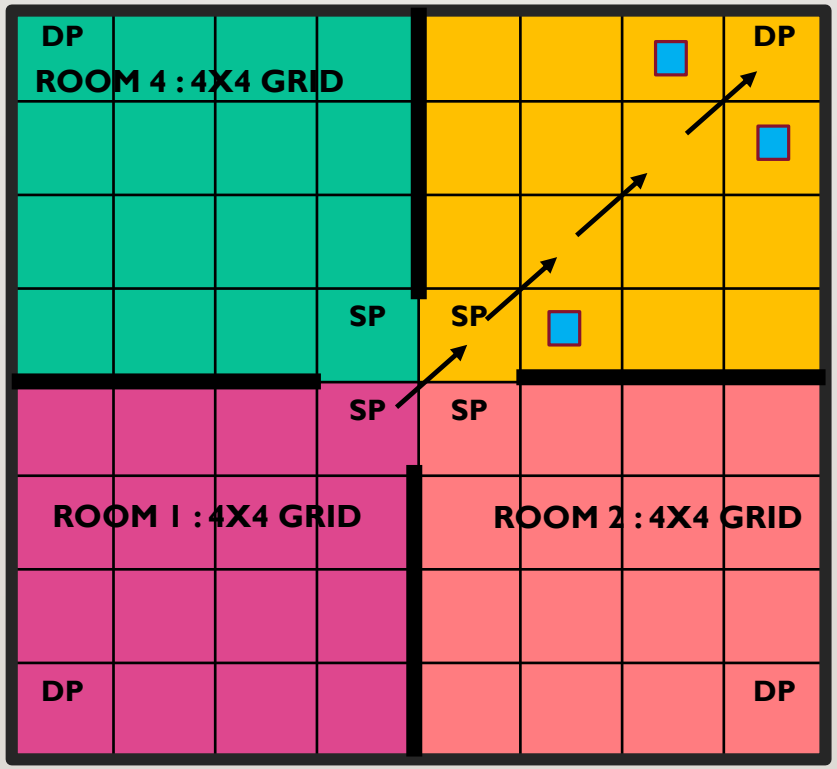
a. Obstacle Occurrence scenarios tested in Room 1

Path Planning with two obstacles



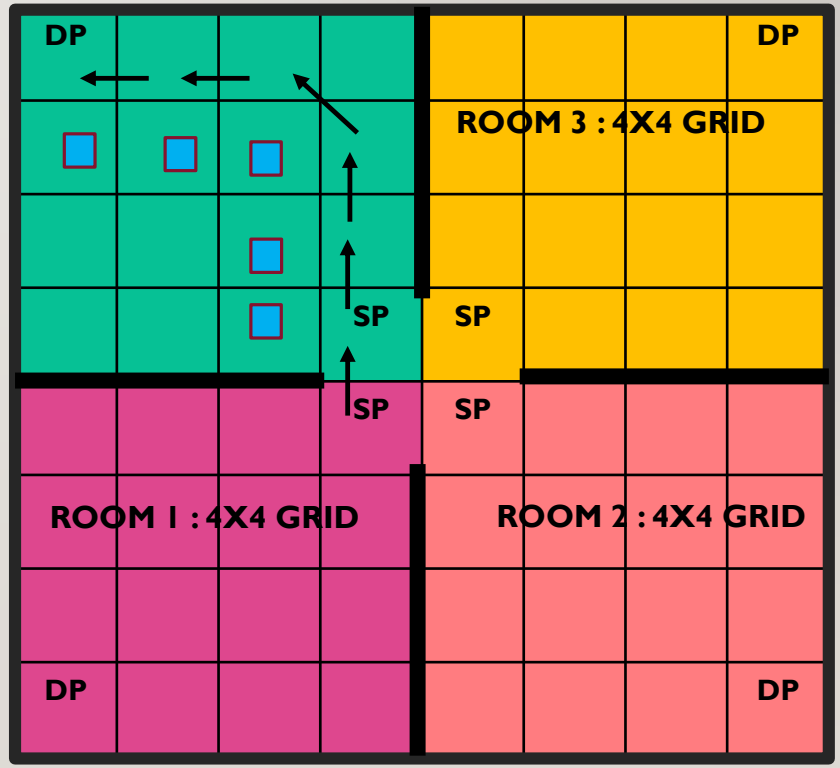
b. Obstacle Occurrence scenarios tested in Room 3

Path Planning with 3 obstacles



c. Obstacle Occurrence scenarios tested in Room 4

Path planning with 5 obstacles



AMRITA VISHWA VIDYAPEETHAM

ALGORITHM 3 : Q LEARNING ALGORITHM

a. Algorithm

begin

initialize $Q[S, A]$ arbitrarily

observe current state s

select action a using a policy based on Q

repeat forever:

carry out an action a

observe reward r and state s'

select action a' using a policy based on Q

Q_[s_, a_] = Q[s, a] + λ * (r + γ * max(Q_[s_, a_]) – Q[s, a])

$s \leftarrow s'$;

$a \leftarrow a'$;

end-repeat

end

b. Tool Used : Python 3.7.3

c. Assumptions Made

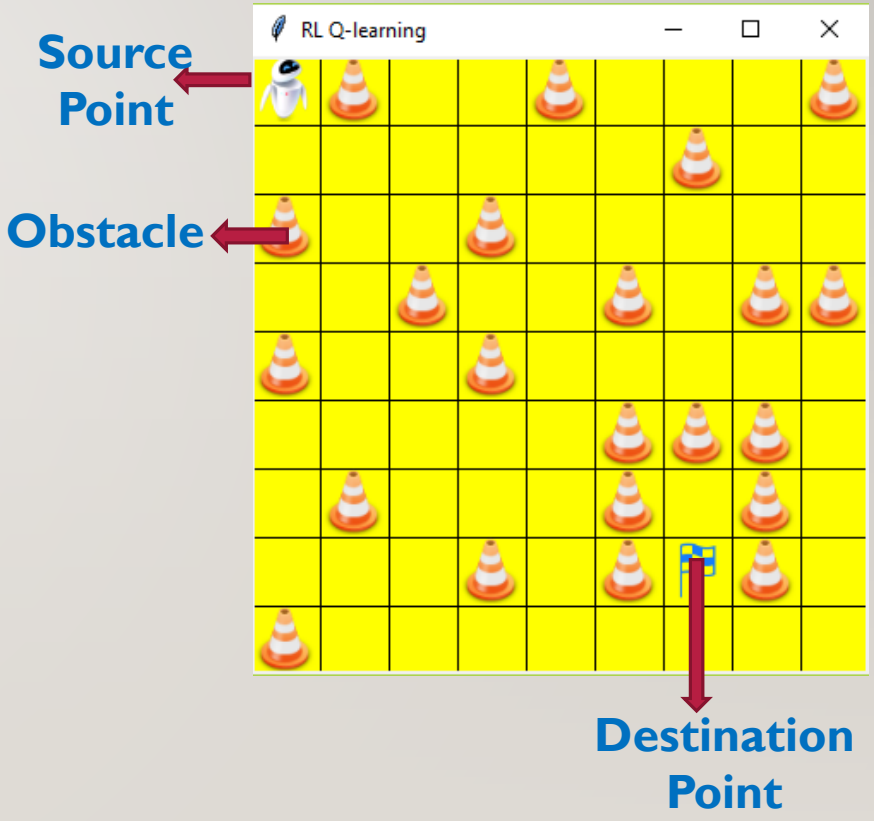
- **9x9 grid**
- **Altering source and destination points**
- **22 Obstacles**
- **Environment is same for both the algorithm**

IMPLEMENTATION AND COMPARISON OF RL ALGORITHMS: SARSA AND Q LEARNING ALGORITHM

a. Comparison between SARSA and Q learning based on time of computation

SL NO	SOURCE POINT	DESTINATION POINT	TIME TAKEN BY SARSA (seconds)	TIME TAKEN BY Q LEARNING (seconds)
1	(0,0)	(6,7)	88.5889801979065	97.14508700370789
2	(1,8)	(8,4)	63.47014451026 91	58.96404004096985
3	(8,1)	(1,8)	85.5343763828277	97.9881820678711
4	(2,0)	(8,7)	96.5203764438629	117.8740122318267
5	(0,3)	(5,0)	41.1859951019287	40.31235694885254

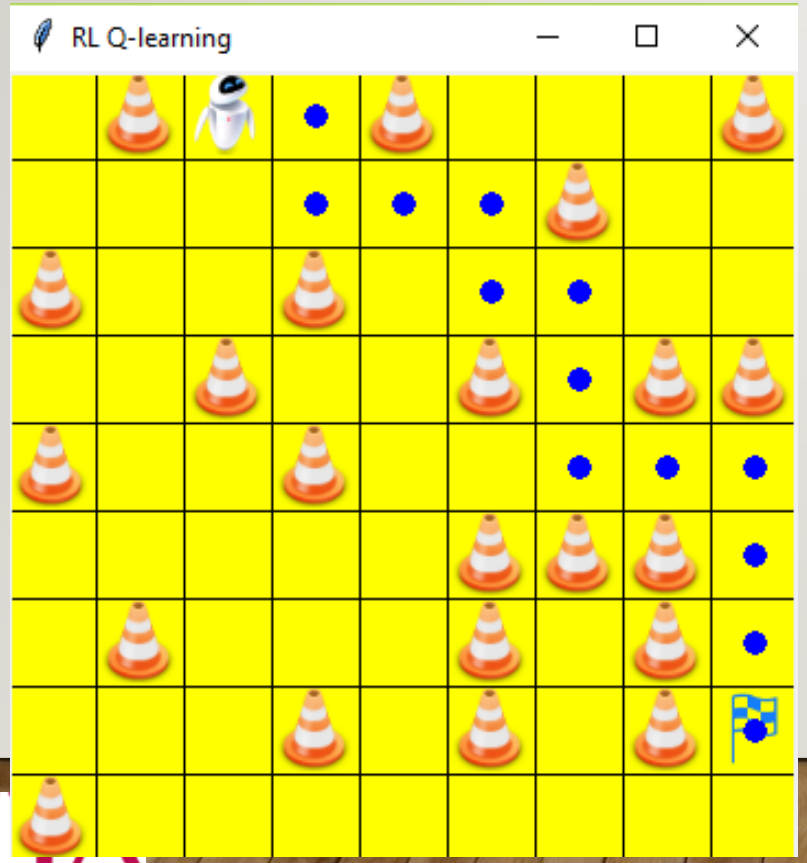
b. Environment Created

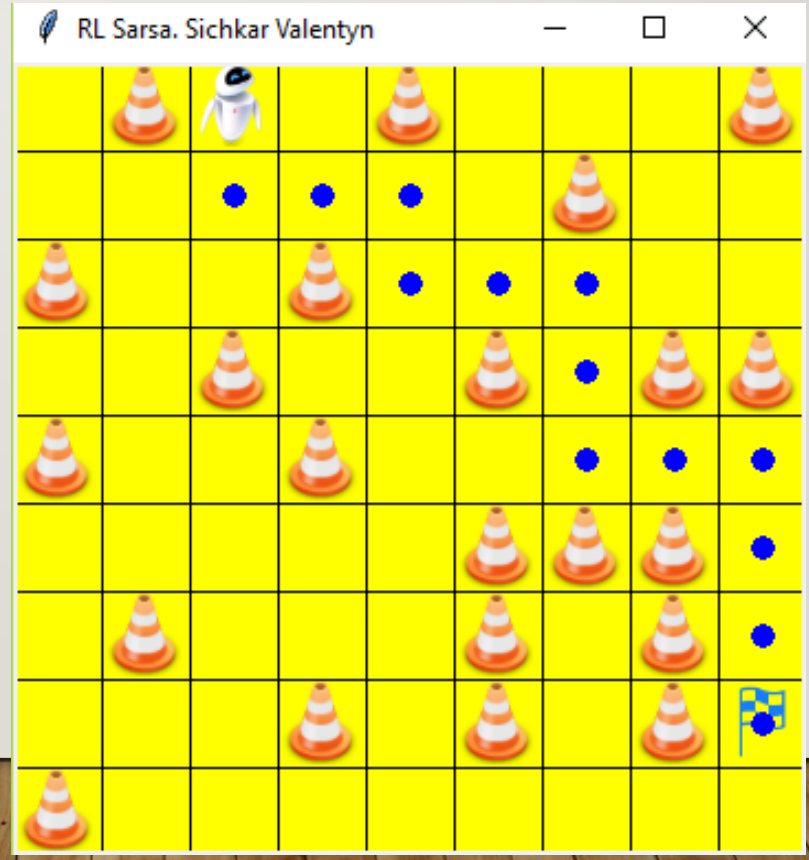


IMPLEMENTATION AND COMPARISON OF RL ALGORITHMS: SARSA AND Q LEARNING ALGORITHM

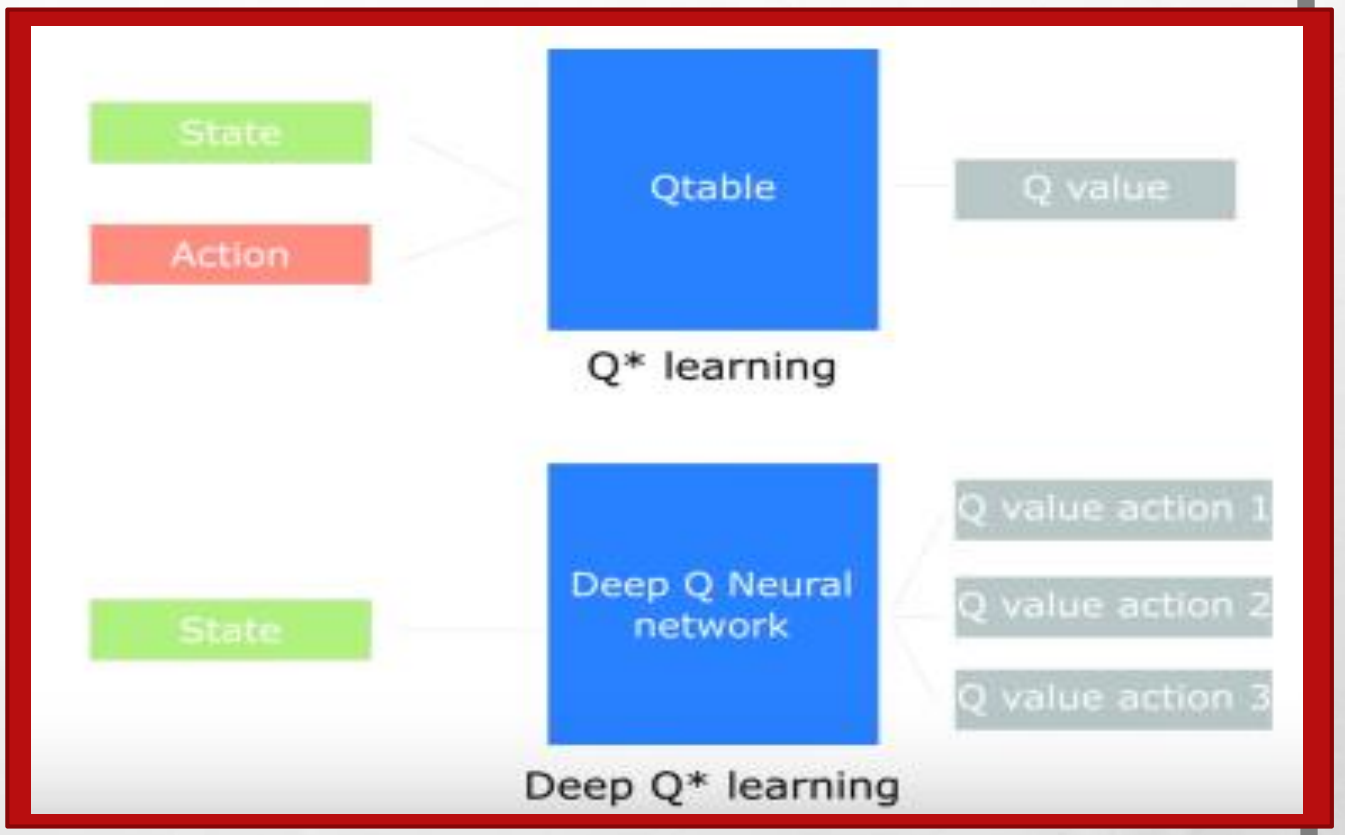
d. Comparison between SARSA and Q learning based on the path taken

SP: (2,0)	DP: (8,7)



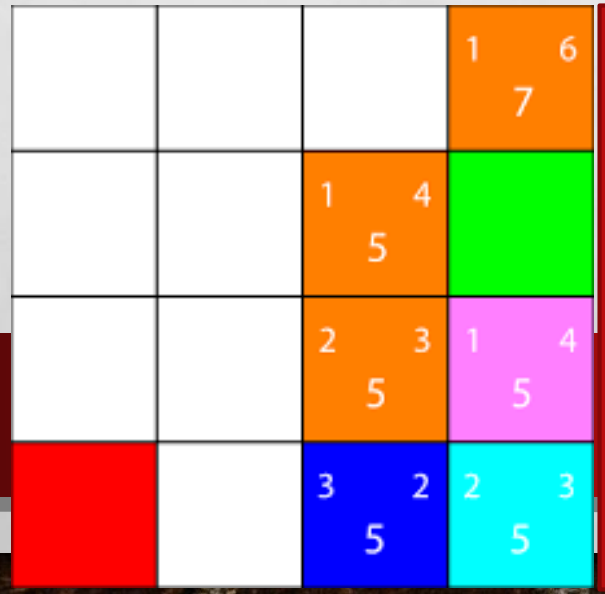


DEEP Q NETWORK ALGORITHM



DISADVANTAGES OF Q LEARNING:

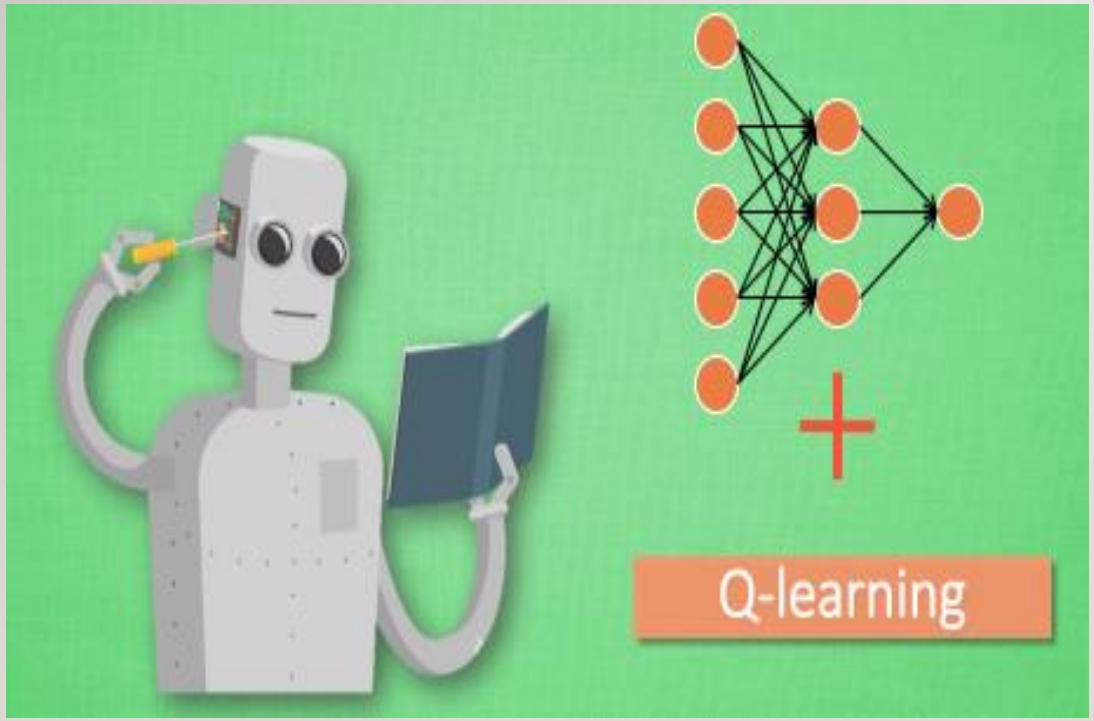
- NOT ENOUGH MEMORY
- LARGE TABLES



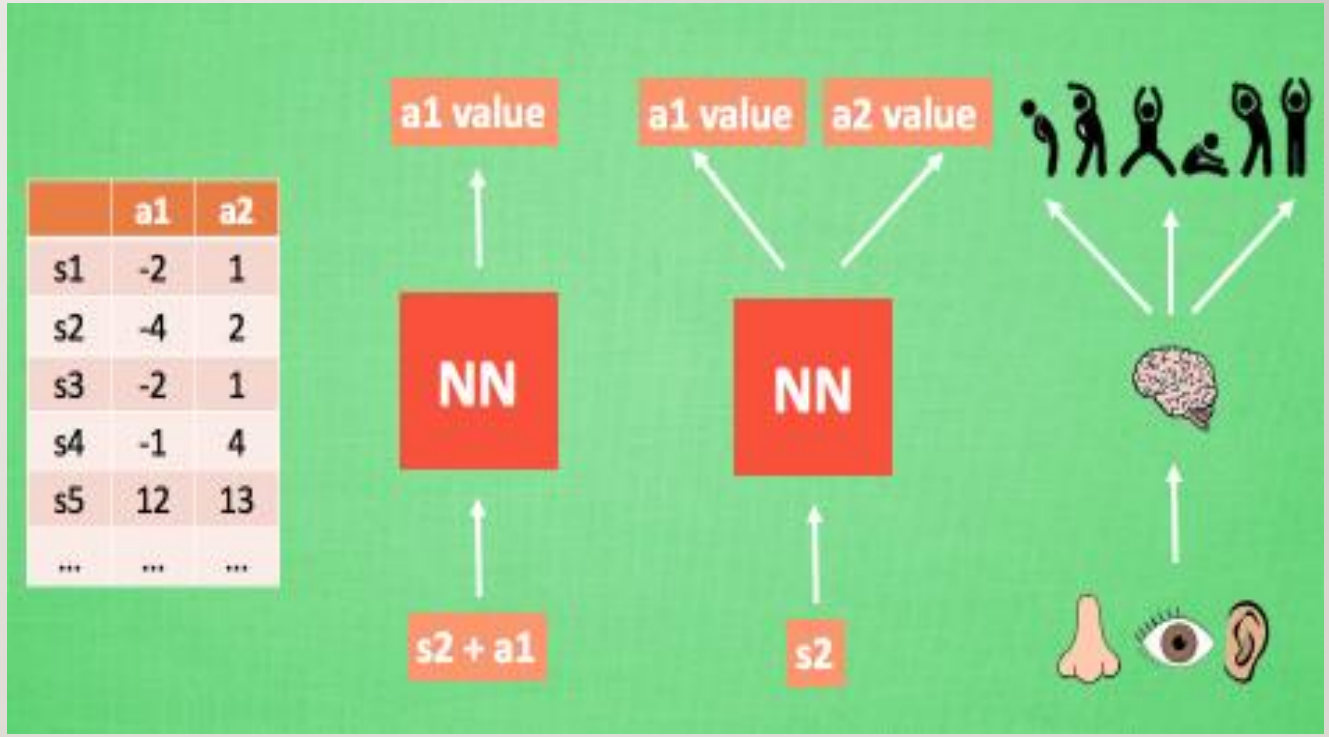
ALGORITHM 4 : DEEP Q NETWORK

a. What is Deep Q Network Algorithm (DQN)

Neural Network + Q learning algorithm = DQN



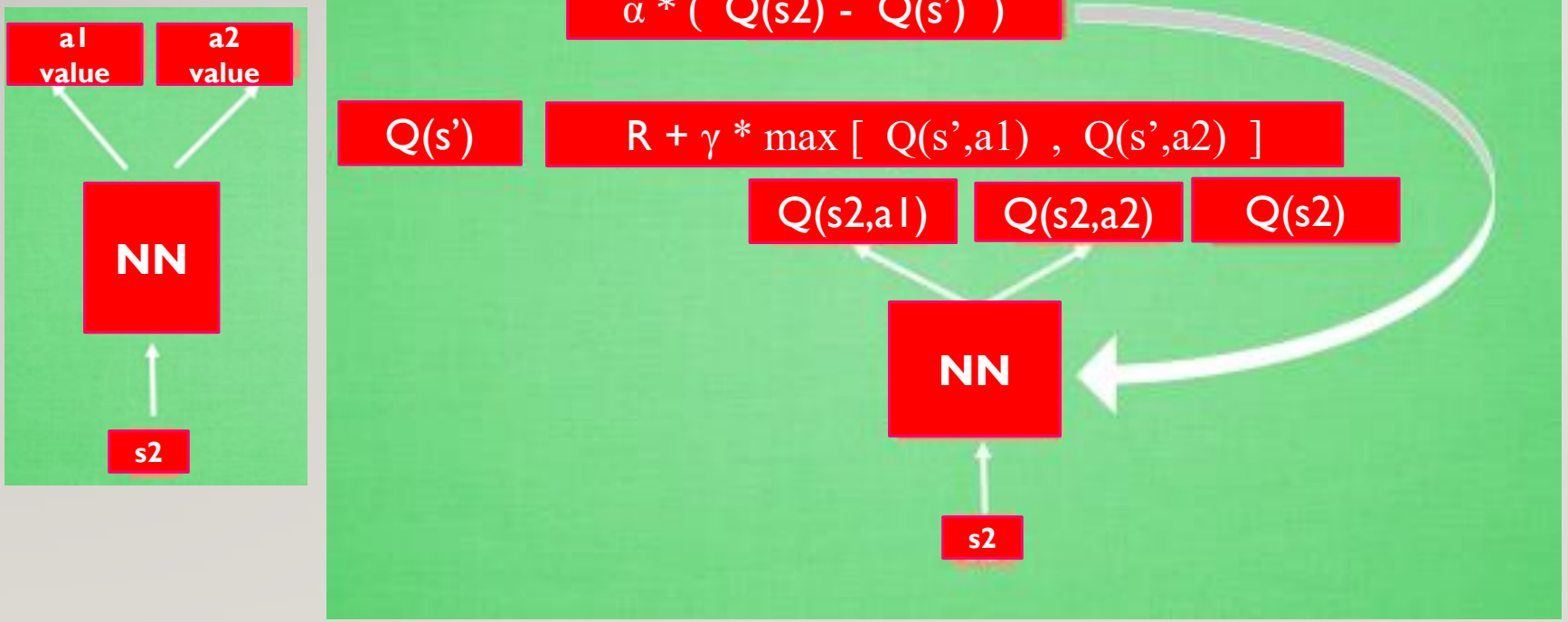
b. Role of Neural Networks



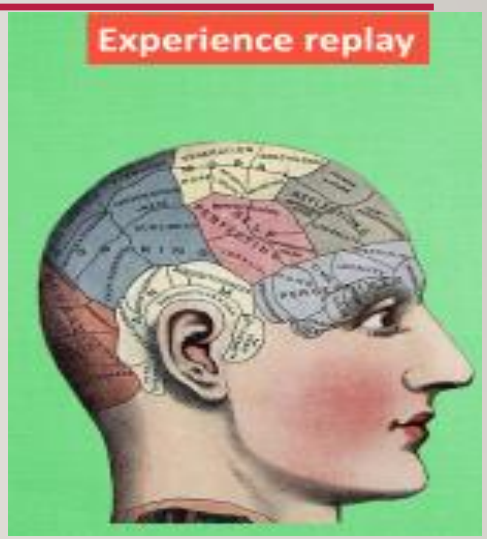
	a1	a2
s1	-2	1
s2	-4	2
s3	-2	1
s4	-1	4
s5	12	13
...	...	...

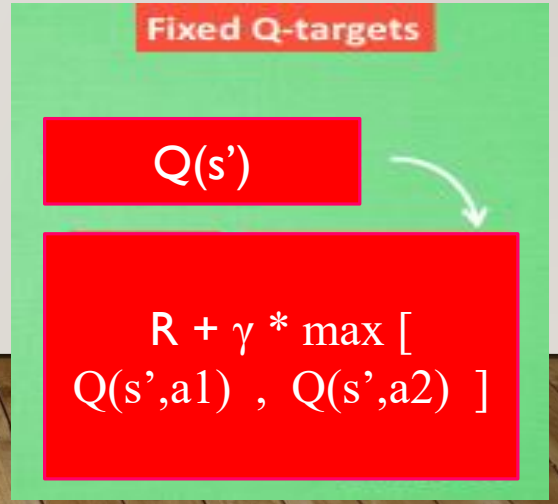
IMPLEMENTATION OF DEEP Q NETWORK ALGORITHM FOR AN 4X4 GRID

c. Update Neural Network



d. Two main factors of DQN



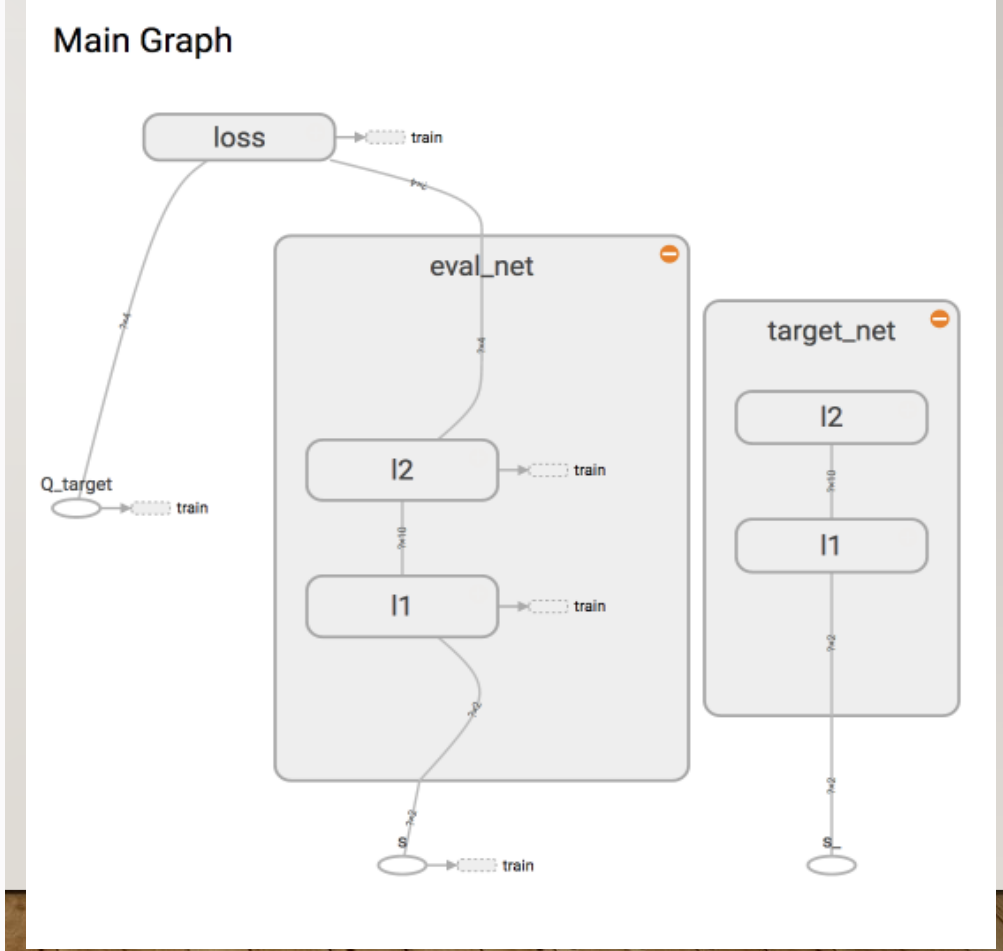


IMPLEMENTATION OF DEEP Q NETWORK ALGORITHM FOR AN 4X4 GRID

e. DQN Algorithm

1. Initialize replay memory capacity.
2. Initialize the network with random weights.
3. *For each episode:*
    1. Initialize the starting state.
    2. *For each time step:*
        1. Select an action.
            - *Via exploration or exploitation*
        2. Execute selected action in an emulator.
        3. Observe reward and next state.
        4. Store experience in replay memory.
        5. Sample random batch from replay memory.
        6. Preprocess states from batch.
        7. Pass batch of preprocessed states to policy network.
        8. Calculate loss between output Q-values and target Q-values.
            - Requires a second pass to the network for the next state
        9. Gradient descent updates weights in the policy network to minimize loss.

f. Two Neural Networks



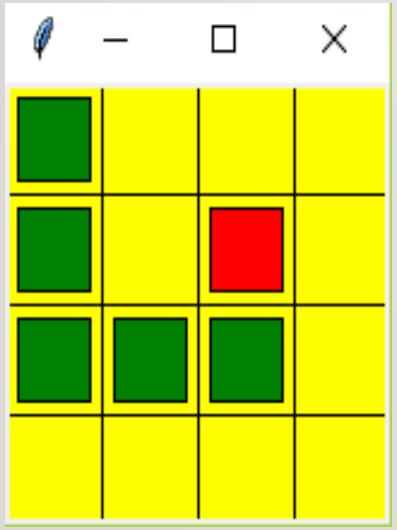
IMPLEMENTATION OF DEEP Q NETWORK ALGORITHM FOR AN 4X4 GRID

g. Output

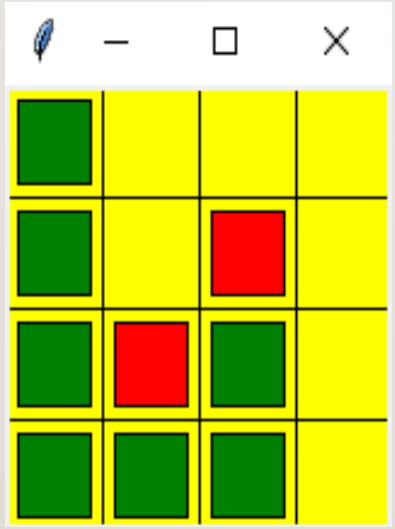
Agent Learning

Cases Tested

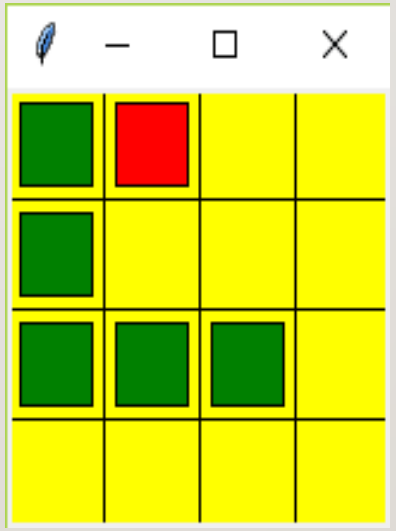
Case 1



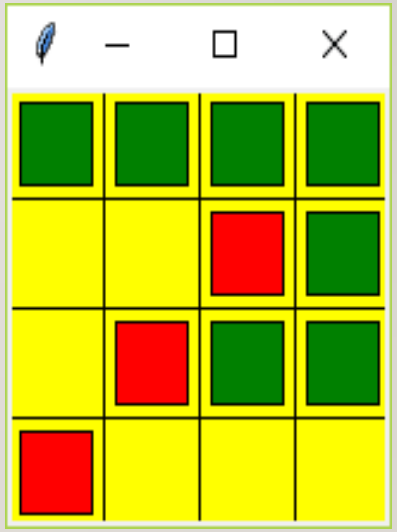
Case 2



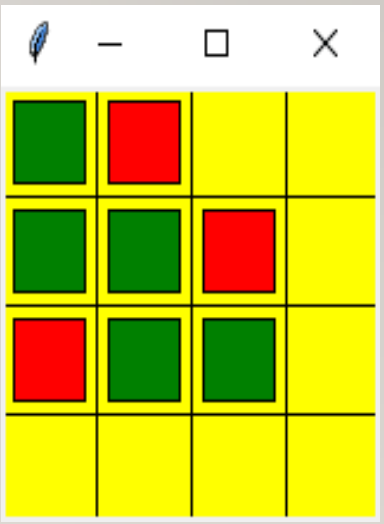
Case 3



Case 4



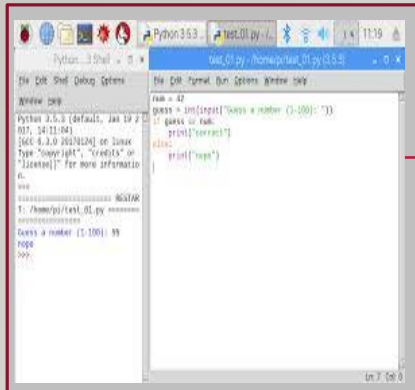
Case 5



CONTRIBUTION

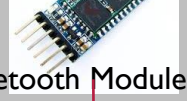
IMPLEMENTATION OF SARSA ALGORITHM FOR A RECTANGULAR GRID OF SIZE 4X5 IN PYTHON 3.7.3

Block Diagram



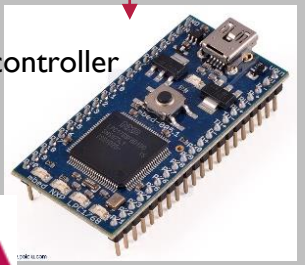
PYTHON IDLE

BLUETOOTH COMMUNICATION

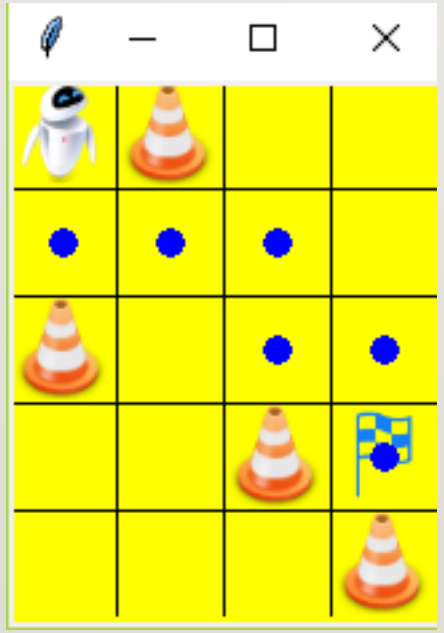


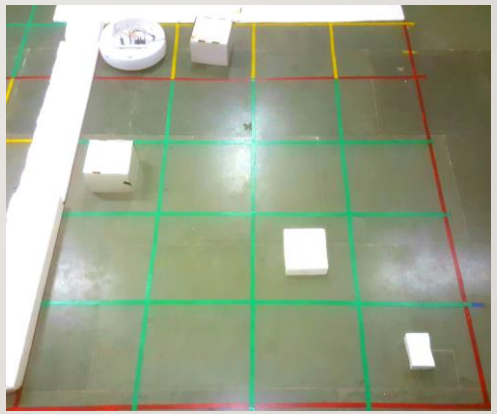
Bluetooth Module HC 05

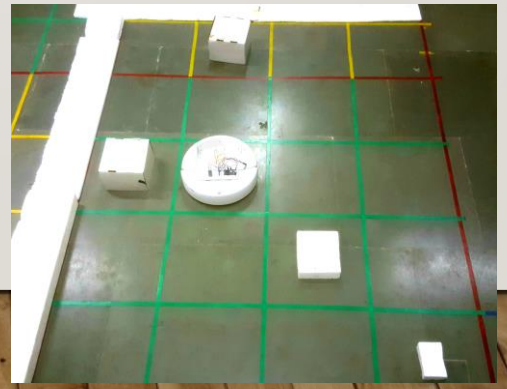
MBED Microcontroller

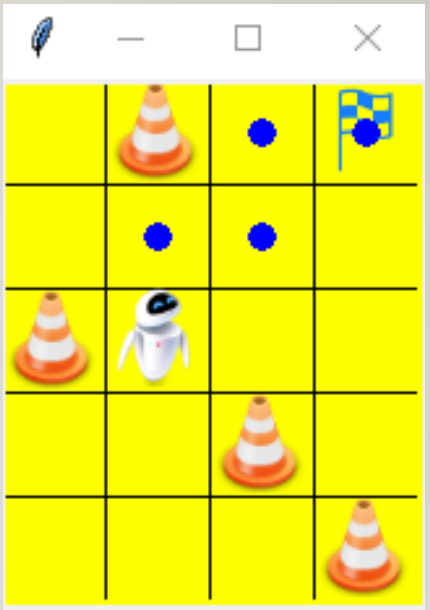


Simulation and Hardware Results for altering source and destination points









AMRITA VISHWA VIDYAPEETHAM

CONCLUSIONS

- Implemented four RL based algorithm
- Compared with conventional algorithms
- Compared with each other based on their time of computation, path travelled and other features depending on the algorithm

FUTURE SCOPE

- Implementation Deep Q Network

(a) For increased size of grid

(b) Both static and dynamic obstacles

(c) In iRobot Create

RESOURCES

1. Zhang, Qian et al. "Reinforcement Learning in Robot Path Optimization." *JSW* 7 (2012): 657-662.
2. D. P. Romero-Martí, J. I. Núñez-Varela, C. Soubervielle-Montalvo and A. Orozco-de-la-Paz, "Navigation and path planning using reinforcement learning for a Roomba robot," *2016 XVIII Congreso Mexicano de Robotica*, Sinaloa, 2016, pp. 1-5.
3. Shreyas J and Sandeep J," Modern Machine Learning Approaches For Robotic Path Planning", IJCSIT(2016):256-259
4. P. Joshy and P. Supriya, "Implementation of robotic path planning using Ant Colony Optimization Algorithm," *2016 International Conference on Inventive Computation Technologies (ICICT)*, Coimbatore, pp. 1-6 , 2016.
5. D. Davis and P. Supriya, "Implementation of Fuzzy-Based Robotic Path Planning", ICETECH, 2016.
6. RAVISHANKAR, N. R.;VIJAYAKUMAR, M.V."Reinforcement Learning Algorithms: Survey and Classification". *ISSN: 0974-5645, 2017.*
7. J. K. Goyal and K. S. Nagla, "A new approach of path planning for mobile robots," *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, New Delhi, 2014, pp. 863-867. doi: 10.1109/ICACCI.2014.6968200
8. M. Lin, K. Yuan, C. Shi and Y. Wang, "Path planning of mobile robot based on improved A∗ algorithm," *2017 29th Chinese Control And Decision Conference (CCDC)*, Chongqing, 2017, pp. 3570-3576. doi: 10.1109/CCDC.2017.7979125
9. https://stats.stackexchange.com/questions/326788/when-to-choose-sarsa-vs-q-learning

PAPER WORK

1. Laya Harwin and P Supriya, "Comparison of SARSA and Temporal Difference Learning Algorithm for Robotic Path Planning for Static Obstacles", Presented the paper in 3rd International Conference on Inventive Systems and Control (ICISC Jan 10, 2019).
2. Writing of journal on progress

DEMO

1. **TD ALGORITHM** for a 4x4 grid A. SIMULATION B. HARDWARE
2. **SARSA ALGORITHM** for an 8x8 grid A. SIMULATION B. HARDWARE
3. **ROBOT TO ROBOT COMMUNICATION** A. HARDWARE
4. **LOAD CARRYING ROBOT** in an industry A. HARDWARE
5. **Q LEARNING ALGORITHM** in python IDLE for an 9x9 grid A. SIMULATION
6. **DEEP Q NETWORK** in python IDLE for an 4x4 grid A. SIMULATION
7. **SARSA ALGORITHM** for a 4x5 grid A. HARDWARE


Thank you

RELATED WORK : SARSA ALGORITHM

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
Ganesha D, Venkatamuni, Vijayakumar Maragal	2017	Implementation of modified SARSA learning technique in EMCAP	International Journal of Engineering & Technology, [S.l.], v. 7, n. 1.5, p. 274-278, dec. 2017. ISSN 2227-524X	• Explains SARSA algorithm is also an approach of learning markov decision process.
D. Xu, Y. Fang, Z. Zhang and Y. Meng	2017	Path Planning Method Combining Depth Learning and Sarsa Algorithm	*10th International Symposium on Computational Intelligence and Design (ISCID)*, Hangzhou, 2017	• SARSA algorithm is also an approach of learning markov decision process (MDP) • Combines depth learning and SARSA algorithm

RELATED WORK : APP DEVELOPMENT

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
Libin M George1, Annu Mariam Abraham2, Ananthapadmanabhan J3 , Vineetha Anna Saji4 , Anil A R5	2017	Implementation of IoT In Smart Robotics: A Survey	International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 6 Issue 3 March 2017	• Android applications are used to get voice and gesture commands for smart robot assistant applications
Mackellar, Bonnie	2010	App inventor for android in a healthcare IT cours	SIGITE'12 - Proceedings of the ACM Special Interest Group for Information Technology Education Conference	• Android apps are easily created using applications like app inventor
F. Turbak, D. Wolber and P. Medlock-Walton	2014	The design of naming features in App Inventor 2	*IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*	• App inventor2 is used to create applications
J. Tyler	2011	*App Inventor for Android*	*App Inventor for Android*	• Android apps are easily created using applications are an open source tool.

RELATED WORK : REINFORCEMENT LEARNING

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
Ravishankar, N.R, Vijayakumar and M.V	2017	Reinforcement Learning Algorithms: Survey and Classification	[SI],2017, doi:10.17485/ /2017 /v10i1/109385.	• Ccomparison between RL algorithms • A variety of path planning algorithms have been discussed which help these robots to find the ideal path
Zhang, Qian et al	2012	Reinforcement Learning in Robot Path Optimization	Journal of Software(*JSW7*)(2012):657- 662.	• Learns through trial and error method • Discusses different RL algorithms
Nihal Altuntas 1, Erkan Imal2, Nahit Emanet1, Ceyda Nur Ozturk	2012	Reinforcement learning based mobile robot navigation	Turkish Journal of Electrical Engineering and Computer Sciences 24(3)	• The primary concept of RL algorithm is communication with environment through learning and also explained how it can be used in mobile robot navaigation
Jingwei Zhang, Jost Tobias Springenberg, Joschka Boedecker, Wolfram Burgard	2016	Deep Reinforcement Learning with Successor Features for Navigation across Similar Environments	2016arXIv1 ,61205533Z}2016, arXiv e-prints,arXiv:1612.05533.	• All autonomous robots reach the end point through simultaneous localization, mapping and path planning taking the most advantageous path • Explains the successor features for navigating in different environments
L. P. Kaelbling, M. L. Littman, and A. P. Moore	1996	Reinforcementlearning: A survey	Journal of Artificial Intelligence Research	• Learns through trial and error method
Kwon, Woo Young & Suh, Il Hong & Lee, Sanghoon & Cho, Young-Jo	2018	Fast reinforcement learning using stochastic shortest paths for a mobile robo	*JSW*7-82 - 87. 1	• Here the robots regularly balance the actions in order to increase its performance.

RELATED WORK : PATH PLANNING

AUTHOR	YEAR	TITLE	DETAILS	SUMMARY
Chen Xia	2015	Intelligent Mobile Robot Learning in Autonomous Navigation. Automatic Control Engineering	Ecole Centrale de Lille, (2015):2015ECL10026.	• Designing of such robots that are self- governing in amorphous, dynamic and for onerous environments
Devika S. Nair and P. Supriya	2016	Comparison of Temporal Difference Learning Algorithm and Dijkstra's Algorithm for Robotic Path Planning	International Conference on Intelligent Computing and Control Systems, Madurai.	• TD is compared with Dijkstra for different obstacle occurring scenarios based on their time of computation.
Tran Xuan Sang, TranQuoc Kiet, Nguyen ThiUyen	2017	Path Finding Algorithm for Autonomous Robots Based on Reinforcement Learning	International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)	• Algorithms learns through trial and error method • Path planning algorithms using RL
Wei Wu, Zhang Qi Sen, J. B. Mbede and Huang Xinhan	2001	Research on path planning for mobile robot among dynamic obstacles	*Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference*	• In an environment where both unknown and moving obstacles are present fuzzy based path planning is used
W. Yu, C. Yang, K. Su and Y. Tu	2014	Dynamic path planning under randomly distributed obstacle environment	*2014 CACS International Automatic Control Conference (CACS 2014)*	• D* lite path planner is also used for path planning with dynamic obstacles in a haphazardly partitioned environment
Mourtzis, Dimitris & Vlachou, Katerina & Zogopoulos, Vasilios	2018	An IoT-based Platform for Automated Customized Shopping in Distributed Environments	Procedia CIRP. 72. 892-897. 10.1016/j.procir.2018.03.199.	• A prototype version of customised shopping system was built where the information about the destination point is communicated through mobile applications by the customers and the products will be distributed by the distribution centres

STUDY ON THE DIFFERENCES BETWEEN A*, DIJKSTRA AND TD ALGORITHM

A*

- Same as Dijkstra
- A* is faster because it uses Best First Search
- Heuristic Function, f(n)=g(n)+h(n), g(n) represents the cost of the path from the starting point to the vertex n. h(n) represents the heuristic estimated cost from vertex n to the g.

DIJKSTRA

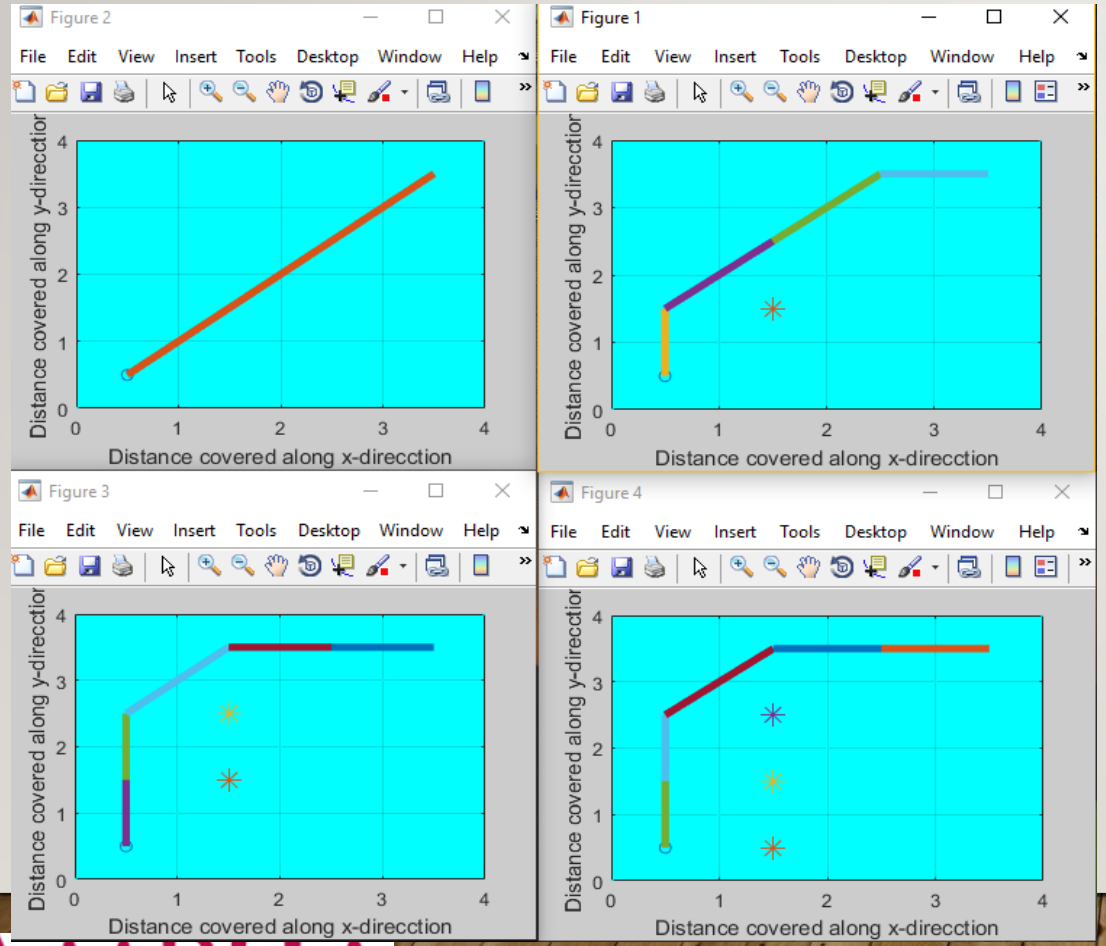
- It is Simple as compare to A*
- Slower because it uses Greedy Best First Search
- f(n)=g(n), g(n) represents the cost of the path from the starting point to the vertex n.
- Dijkstra Algorithm is the worst case of A star Algorithm.

TD

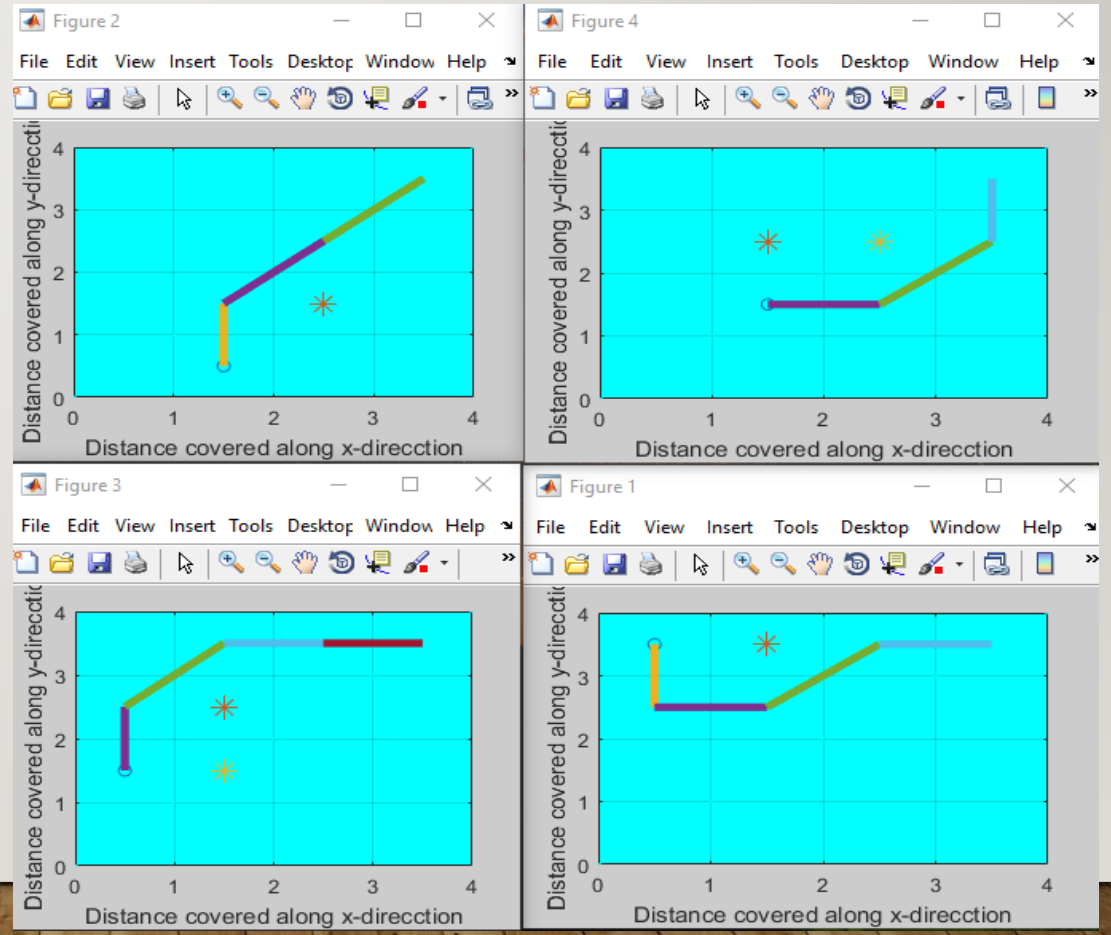
- Fastest
- Estimate the Q- function according to the reward function
- Less stable and may converge to the wrong solution
- $Q(S,a)=r(s,a)+Ymax(Q(S',a'))$

SIMULATION RESULTS FOR TEMPORAL DIFFERENCE LEARNING ALGORITHM

Path Planning with 0,1,2 and 3 obstacles

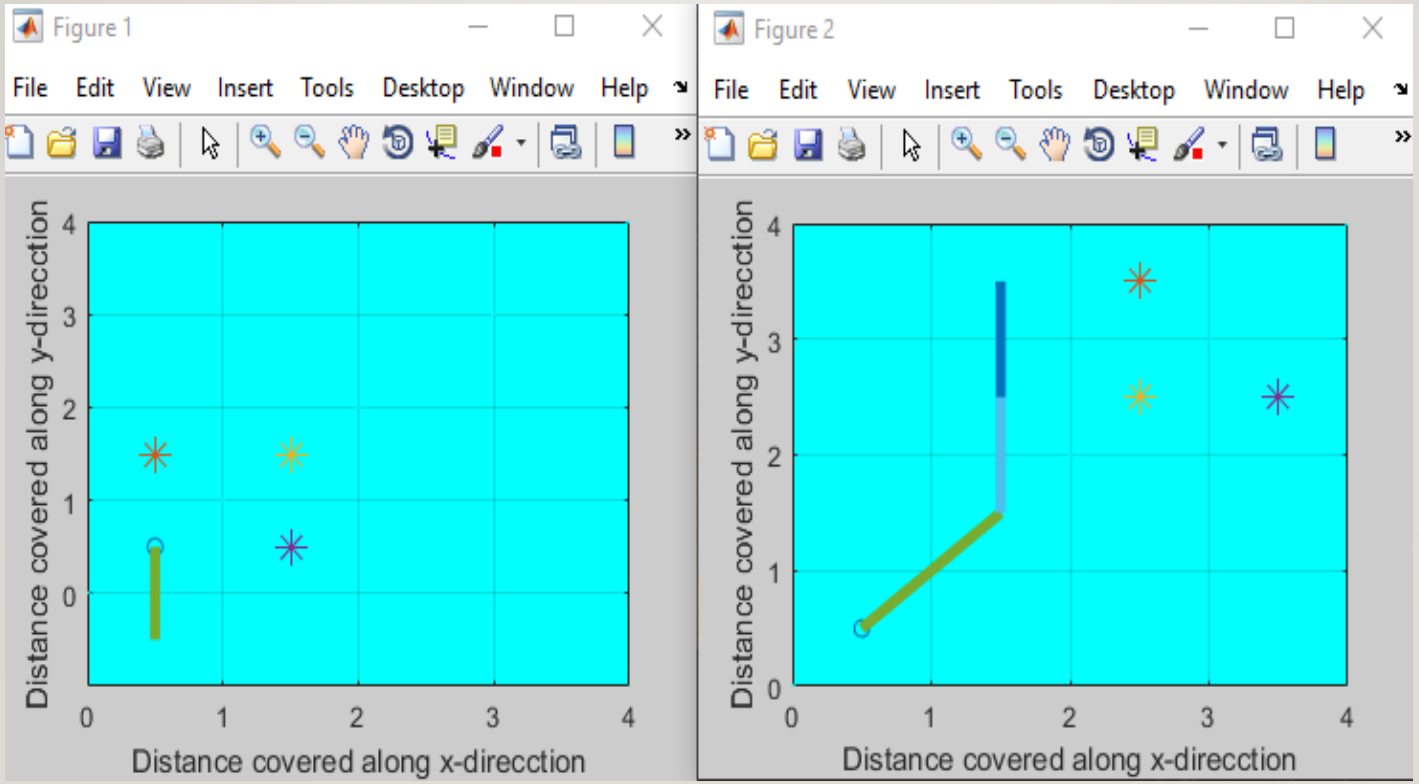


Path Planning for different source points

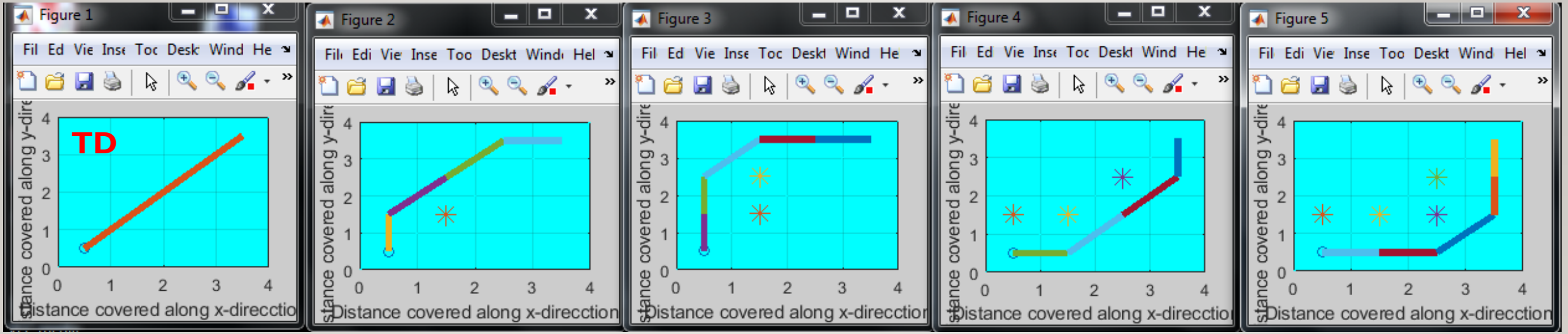


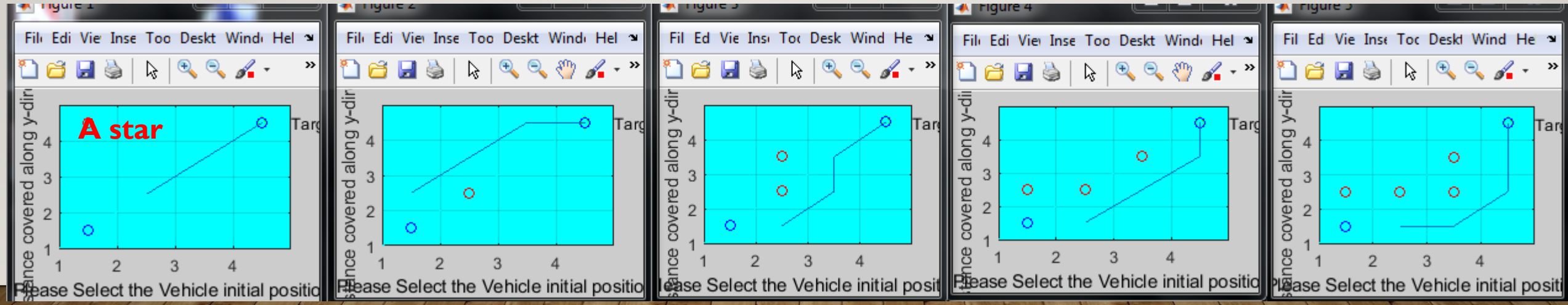
SIMULATION RESULTS FOR TEMPORAL DIFFERENCE LEARNING ALGORITHM

Cases where TD algorithm fails

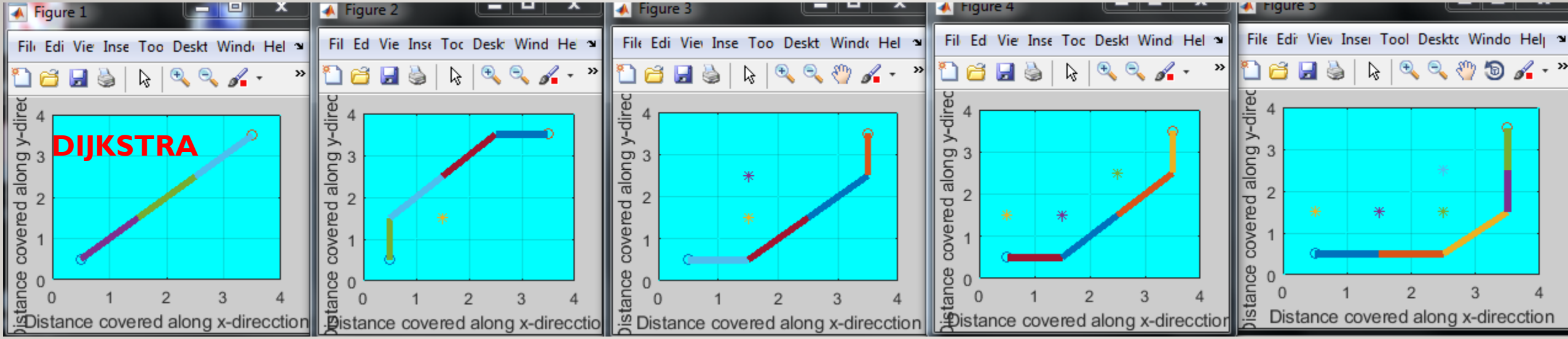


SIMULATION RESULTS FOR A*,DIJKSTRA AND TD ALGORITHM FOR 4X4 GRID





SIMULATION RESULTS FOR A*,DIJKSTRA AND TD ALGORITHM FOR 4X4 GRID(CONTINUED)



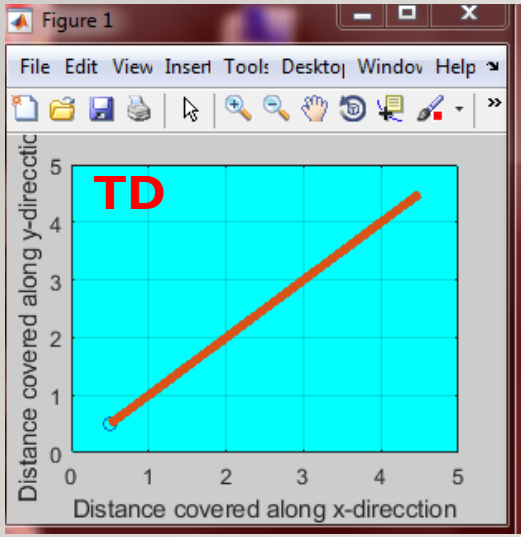
COMPARITIVE STUDY BETWEEN A*,DIJKSTRA AND TD ALGORITHM BASED ON ELAPSED TIME

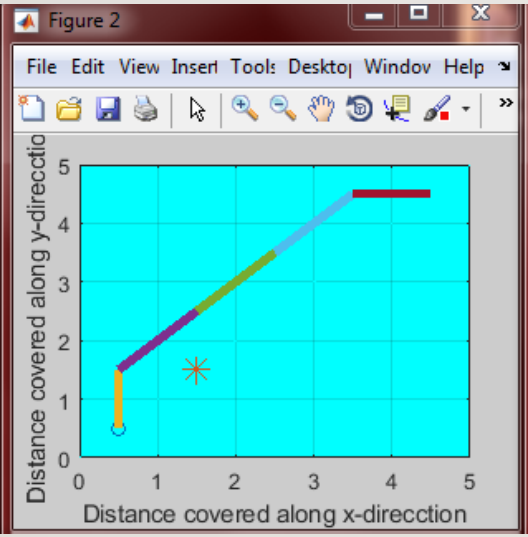
4X4 GRID

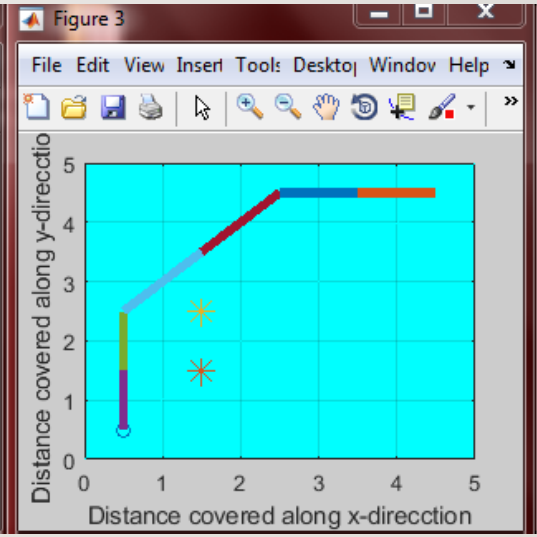
Number of Obstacles	Position of Obstacles	Time taken for Dijkstra algorithm	Time taken for A* algorithm	Time taken for TD algorithm
0	-	0.50636	0.15438	0.005119
1	(1.5,1.5)	0.75862	0.08102	0.005244
2	(1.5,1.5),(1.5,2.5)	0.75906	0.04286	0.007160
3	(0.5,1.5),(1.5,1.5),(2.5,2.5)	0.76045	0.08604	0.007299
4	(0.5,1.5),(1.5,1.5),(2.5,1.5),(2.5,2.5)	1.10113	0.08709	0.005827

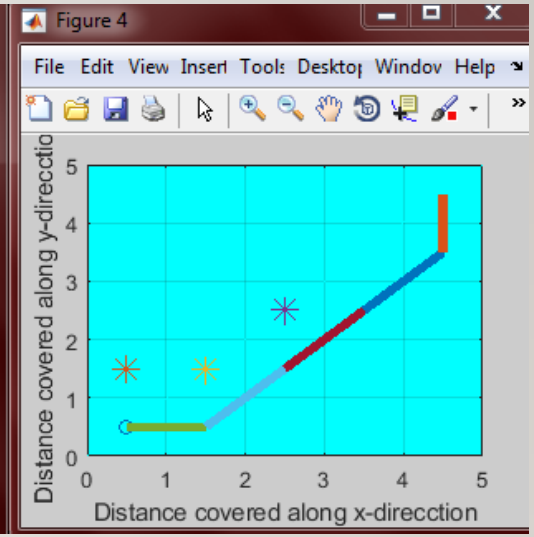
TIME IS MEASURED IN SECONDS

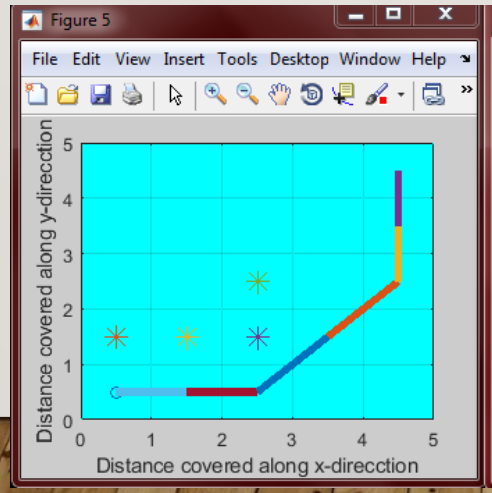
SIMULATION RESULTS FOR A* AND TD ALGORITHM FOR 5X5 GRID

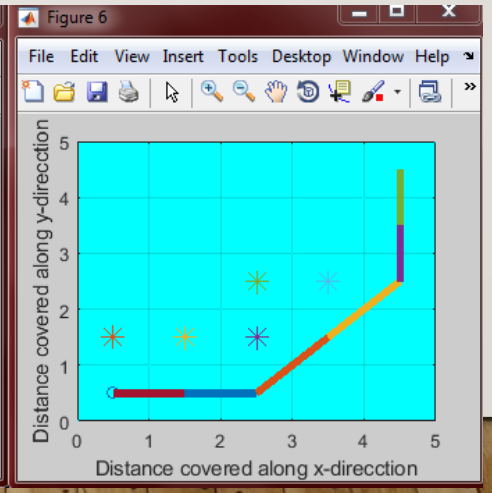




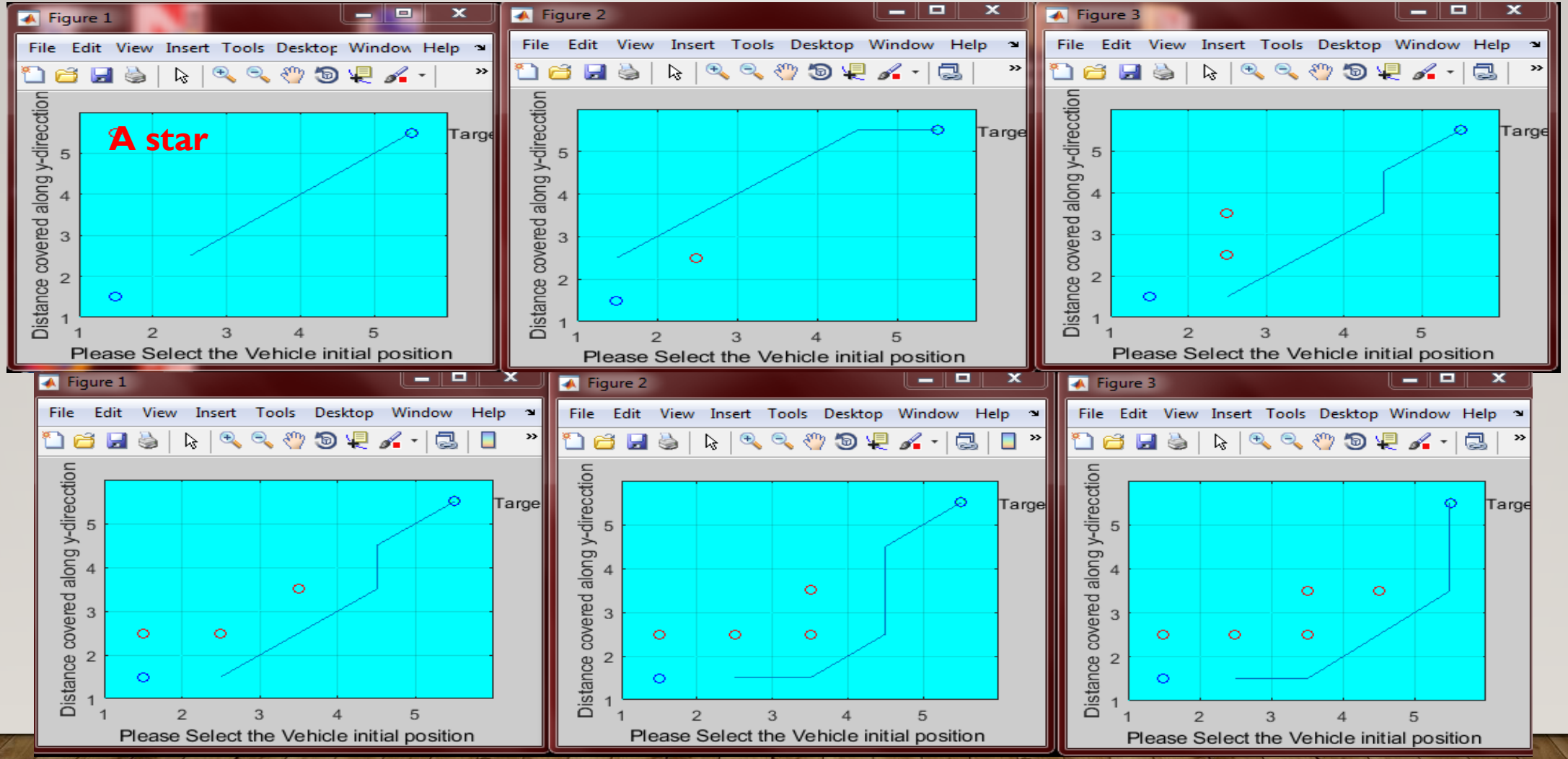








SIMULATION RESULTS FOR A* AND TD ALGORITHM FOR 5X5 GRID(CONTINUED)



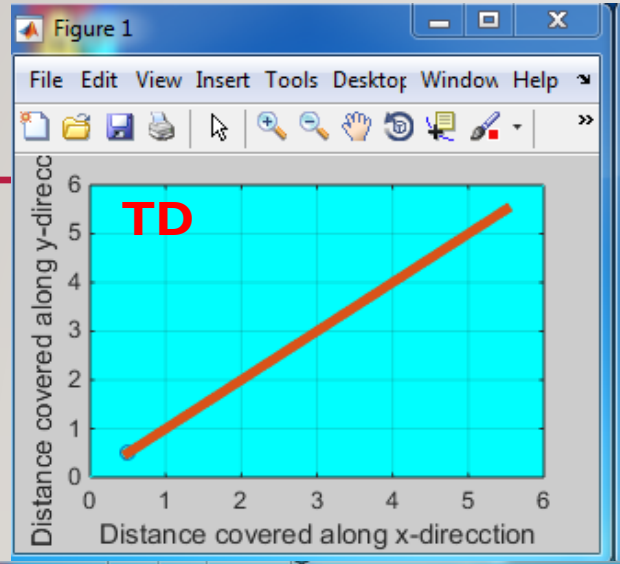
COMPARITIVE STUDY BETWEEN A* AND TD ALGORITHM BASED ON ELAPSED TIME

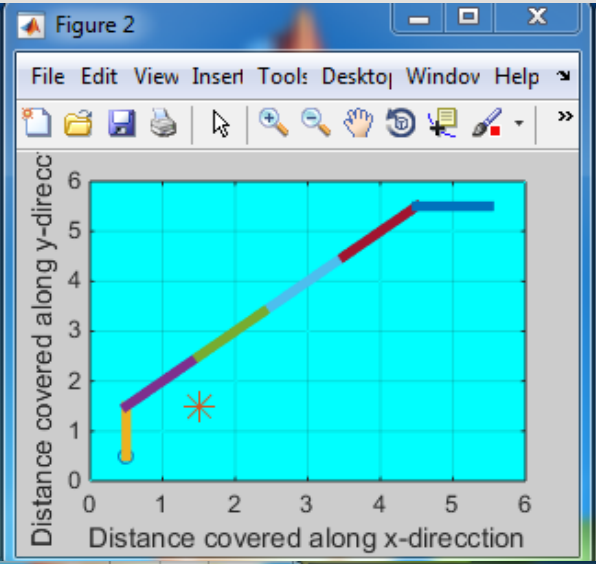
5X5 GRID

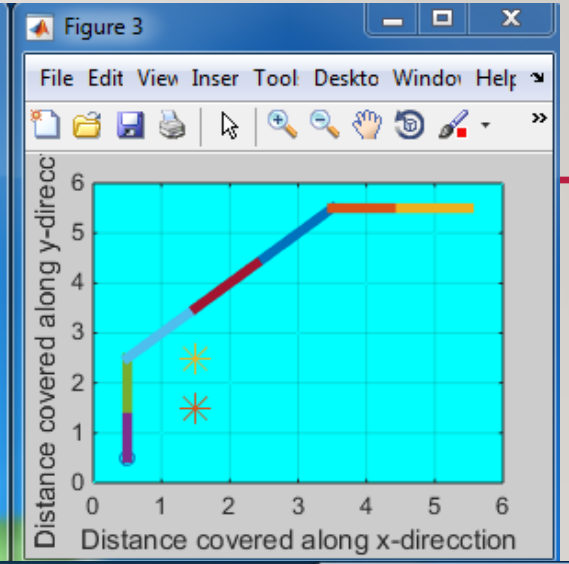
Number of Obstacles	Position of Obstacles	Time taken for A* algorithm	Time taken for TD algorithm
0	-	0.76336	0.000857
1	(1.5,1.5)	1.01342	0.008074
2	(1.5,1.5),(1.5,2.5)	1.02009	0.009722
3	(0.5,1.5),(1.5,1.5),(2.5,2.5)	1.01378	0.006524
4	(0.5,1.5),(1.5,1.5),(2.5,1.5),(2.5,2.5)	1.26427	0.009797
5	(0.5,1.5),(1.5,1.5),(2.5,1.5),(2.5,2.5),(3.5,2.5)	1.26301	0.009710

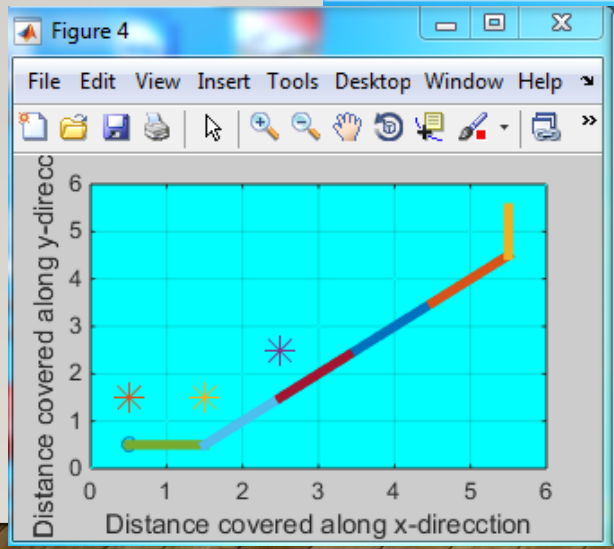
TIME IS MEASURED IN SECONDS

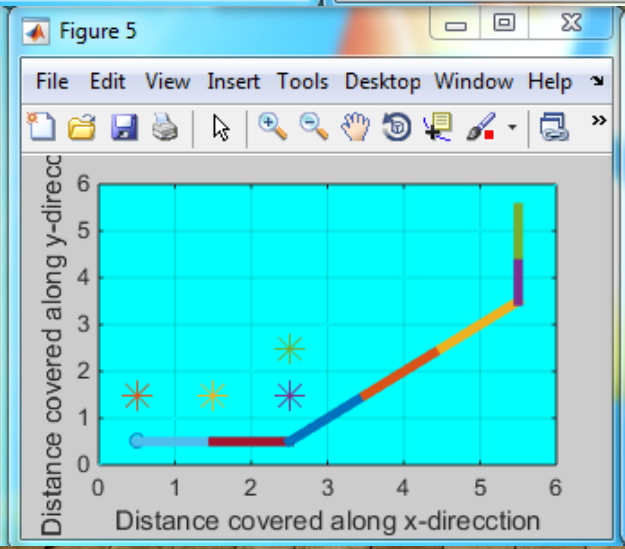
SIMULATION RESULTS FOR A* AND TD ALGORITHM FOR 6X6 GRID

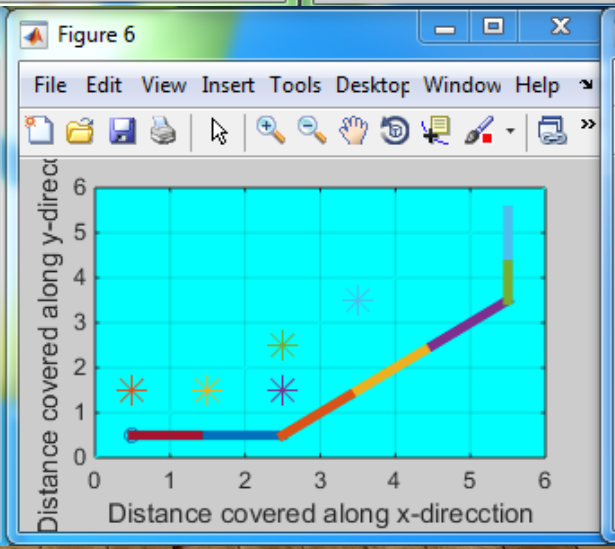


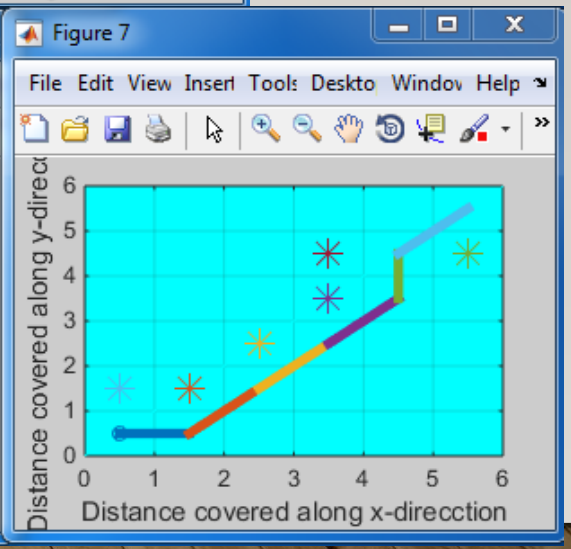






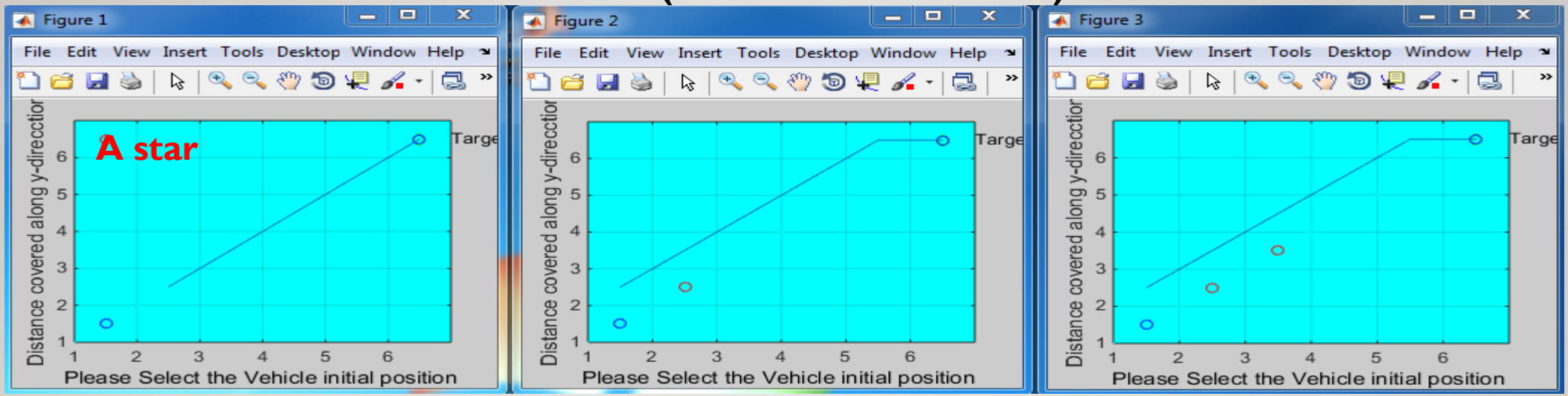


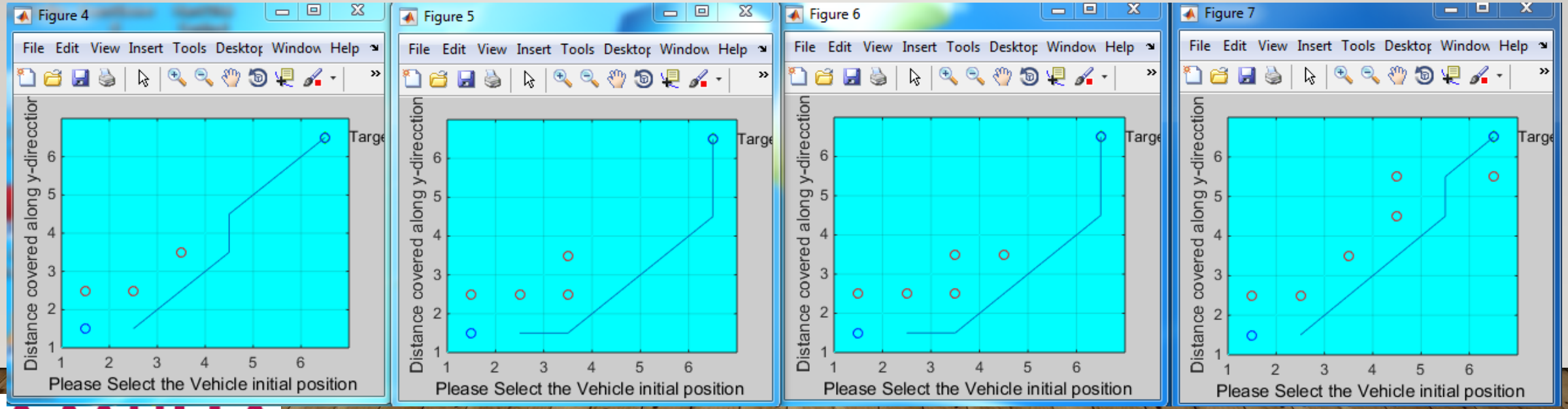




AMRITA VISHWA VIDYAPEETHAM

SIMULATION RESULTS FOR A* AND TD ALGORITHM FOR 6X6 GRID(CONTINUED)





COMPARITIVE STUDY BETWEEN A*,DIJKSTRA AND TD ALGORITHM BASED ON ELAPSED TIME

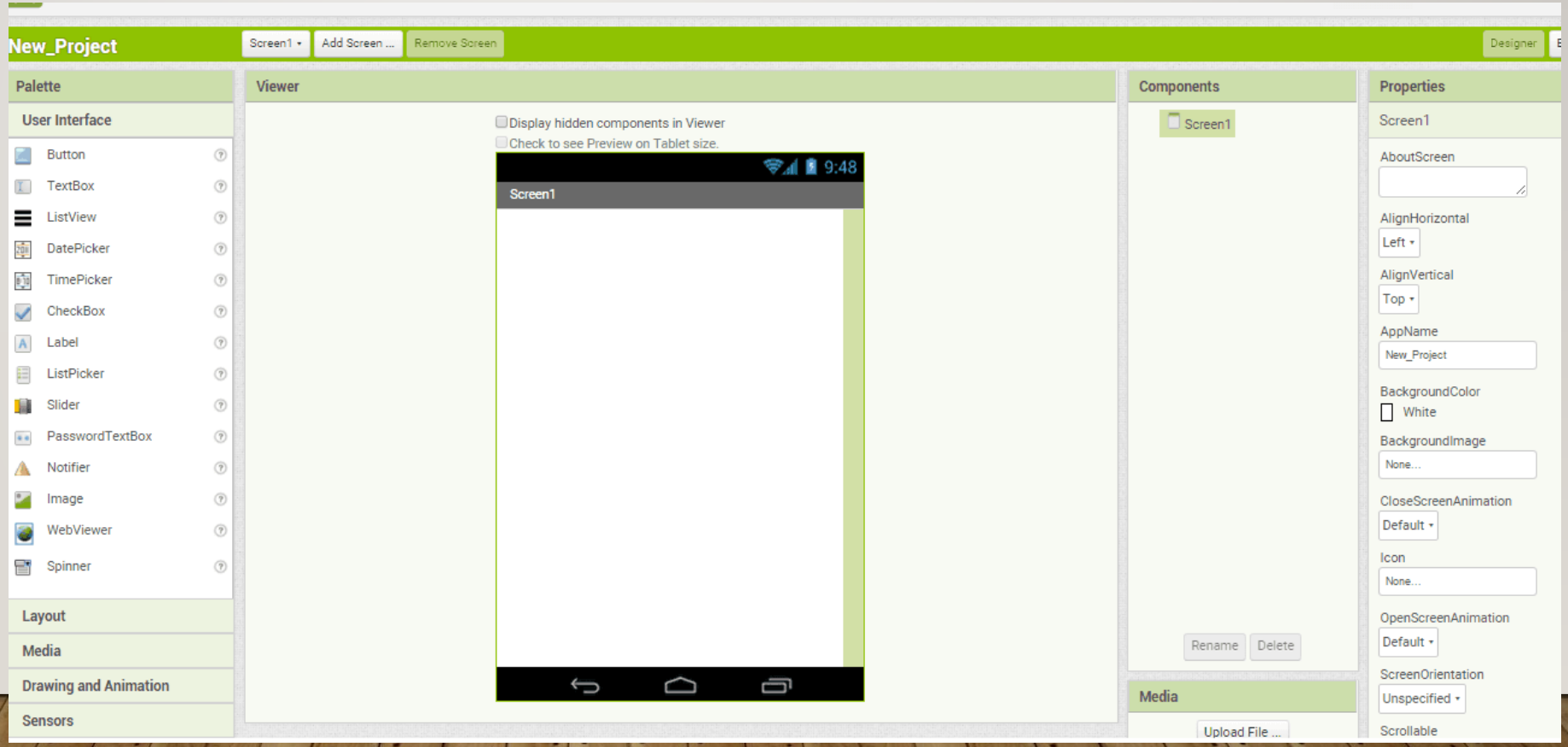
6X6 GRID

Number of Obstacles	Position of Obstacles	Time taken for A* algorithm	Time taken for TD algorithm
0	-	1.052016	0.001834
1	(1.5,1.5)	1.328514	0.018820
2	(1.5,1.5),(1.5,2.5)	1.329846	0.018579
3	(0.5,1.5),(1.5,1.5),(2.5,2.5)	1.309244	0.019258
4	(0.5,1.5),(1.5,1.5),(2.5,1.5),(2.5,2.5)	1.588463	0.021461
5	(0.5,1.5),(1.5,1.5),(2.5,1.5),(2.5,2.5),(3.5,2.5)	1.566313	0.016919
6	(1.5,1.5),(2.5,2.5),(3.5,3.5),(5.5,4.5),(0.5,1.5),(3.5,4.5)	1.574019	0.015919

DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

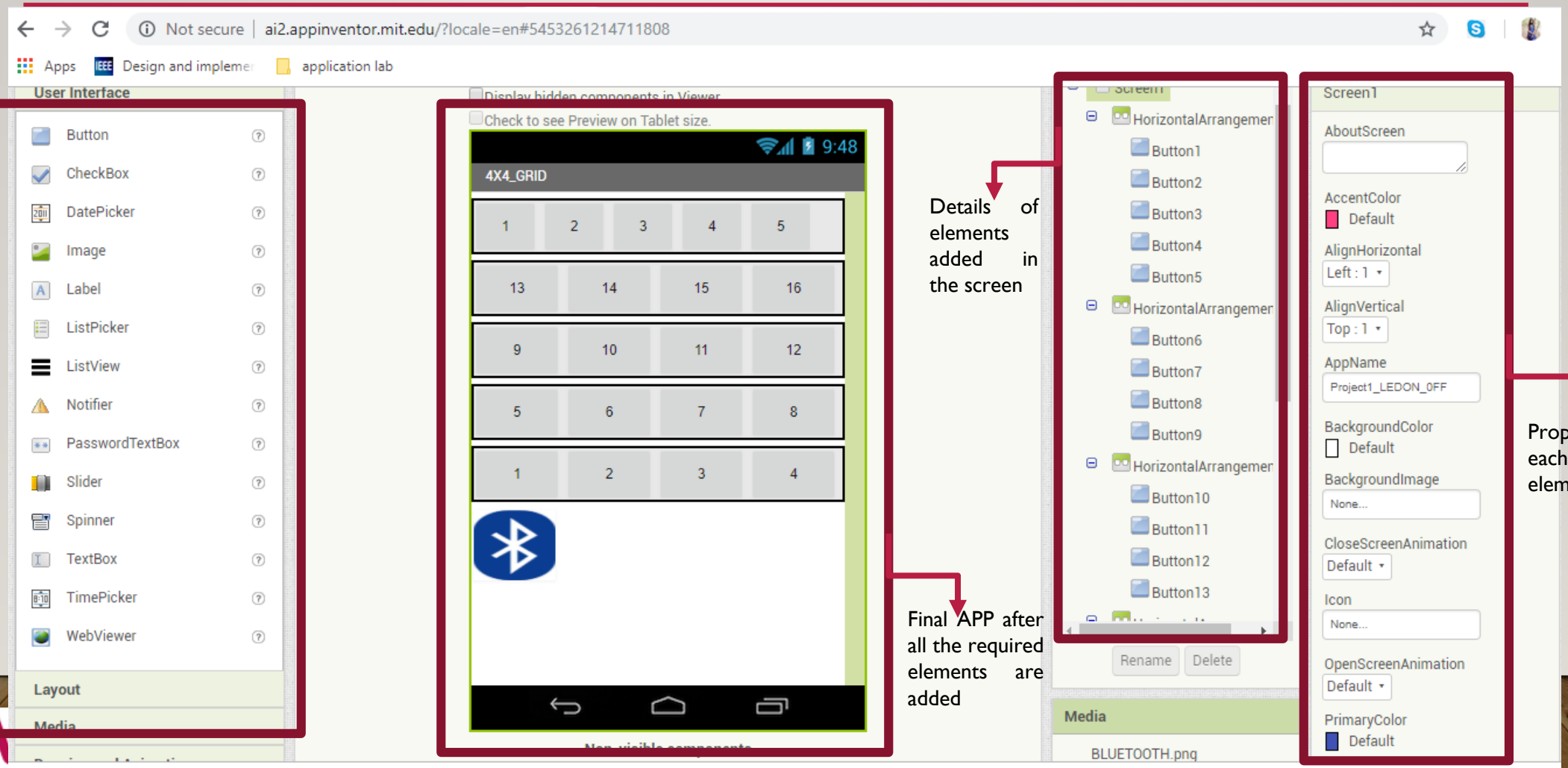
APP DEVELOPMENT

- Online Tool called MIT App Inventor



DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

APP DEVELOPMENT



Elements that we could add to our app

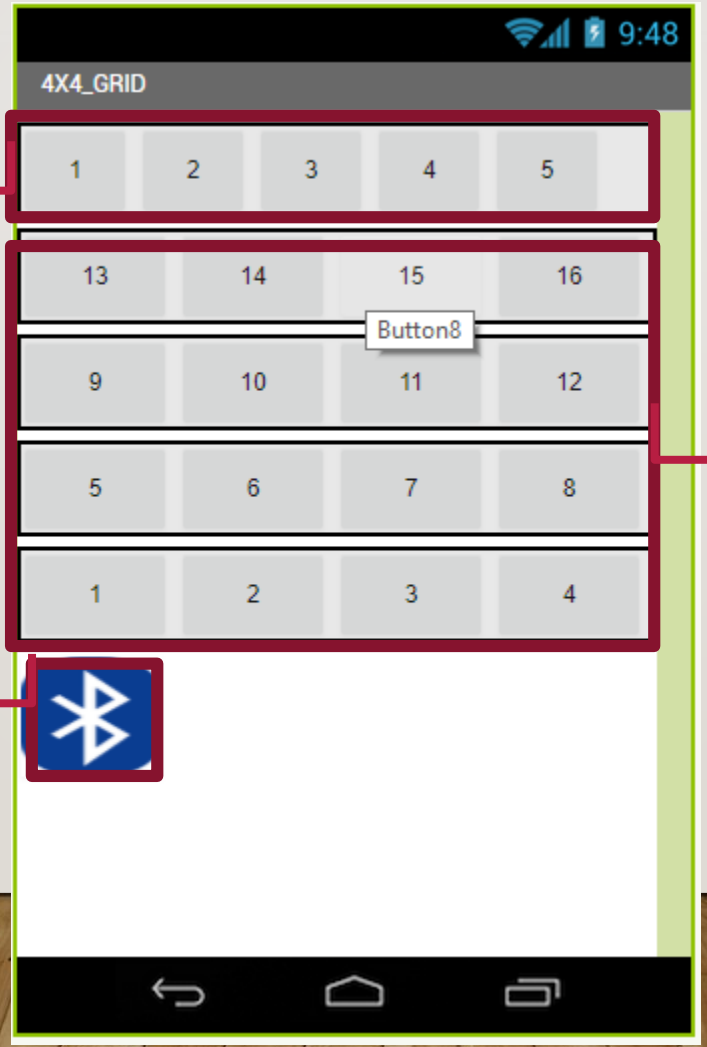
Details of elements added in the screen

Final APP after all the required elements are added

Properties of each inserted element

DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

APP DEVELOPMENT



Buttons for selecting the number of obstacles. Up to 5 obstacles are possible

Buttons for selecting the position of obstacles.

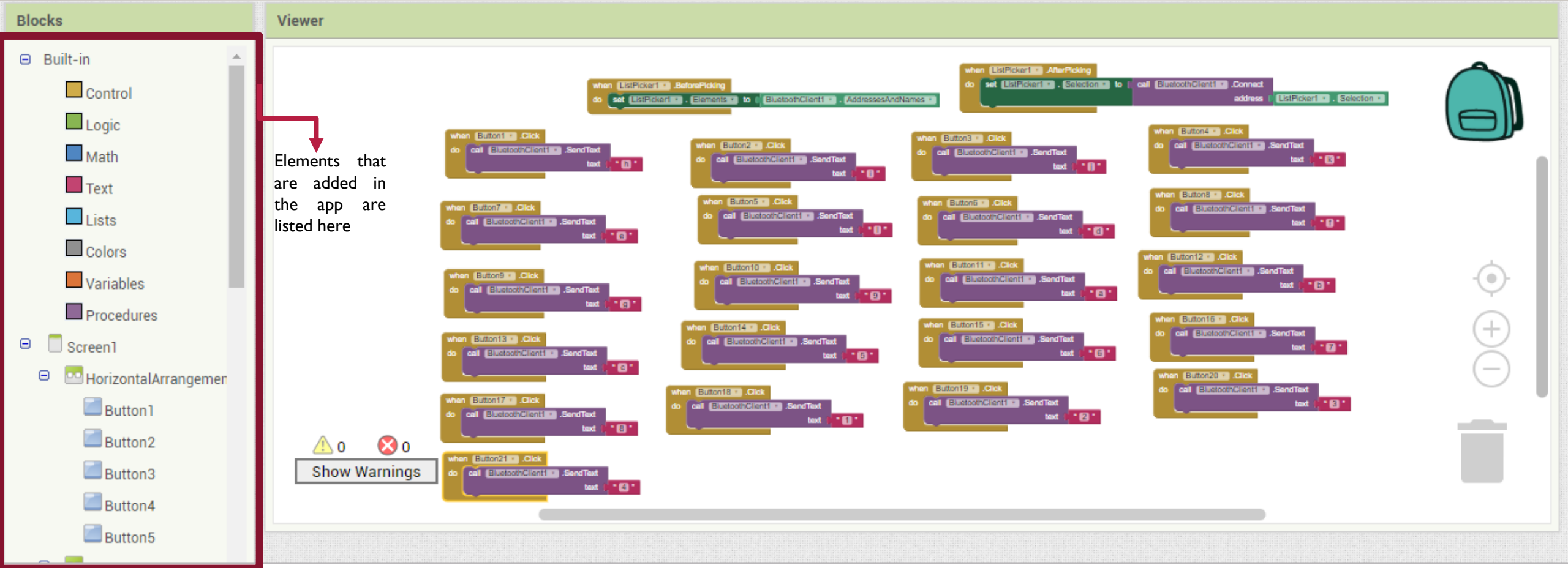
Bluetooth client(which is available in connectivity tab) for connecting with HC 05 module

APP Design is ready

DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

APP DEVELOPMENT

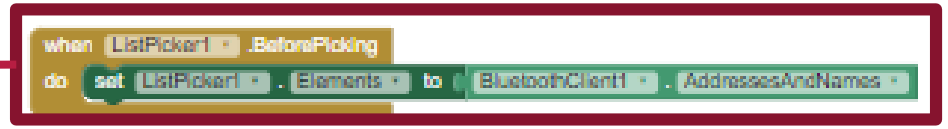
Second part of APP creator • BLOCKS

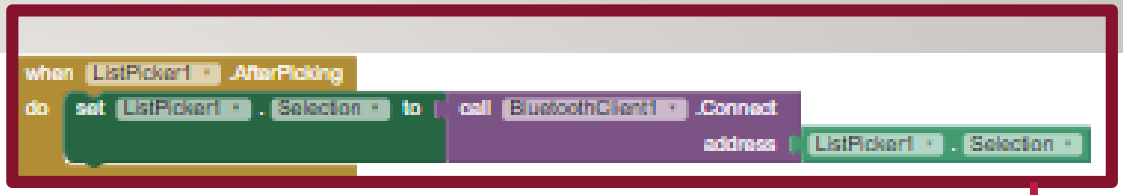


DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

APP DEVELOPMENT

This block is to see the list of linked clients when the Bluetooth is open





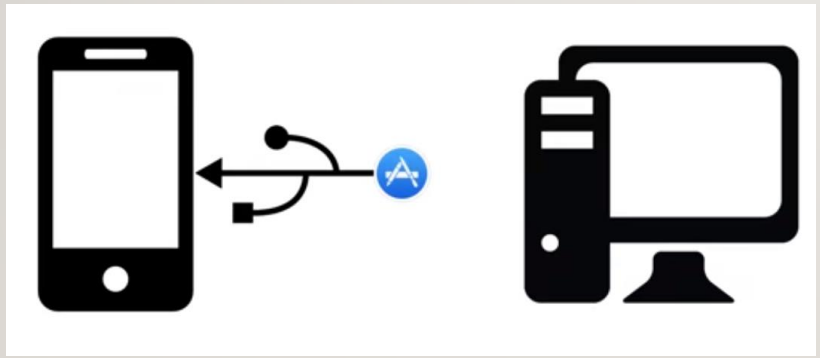
This block is to select the client after picking

This block contains all the other buttons(for the number of obstacles and for the position of obstacles). All these buttons send a unique value to identify each button.

DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

APP DEVELOPMENT

BUILD → SAVE THE APP TO MY COMPUTER → A .apk FILE IS GENERATED

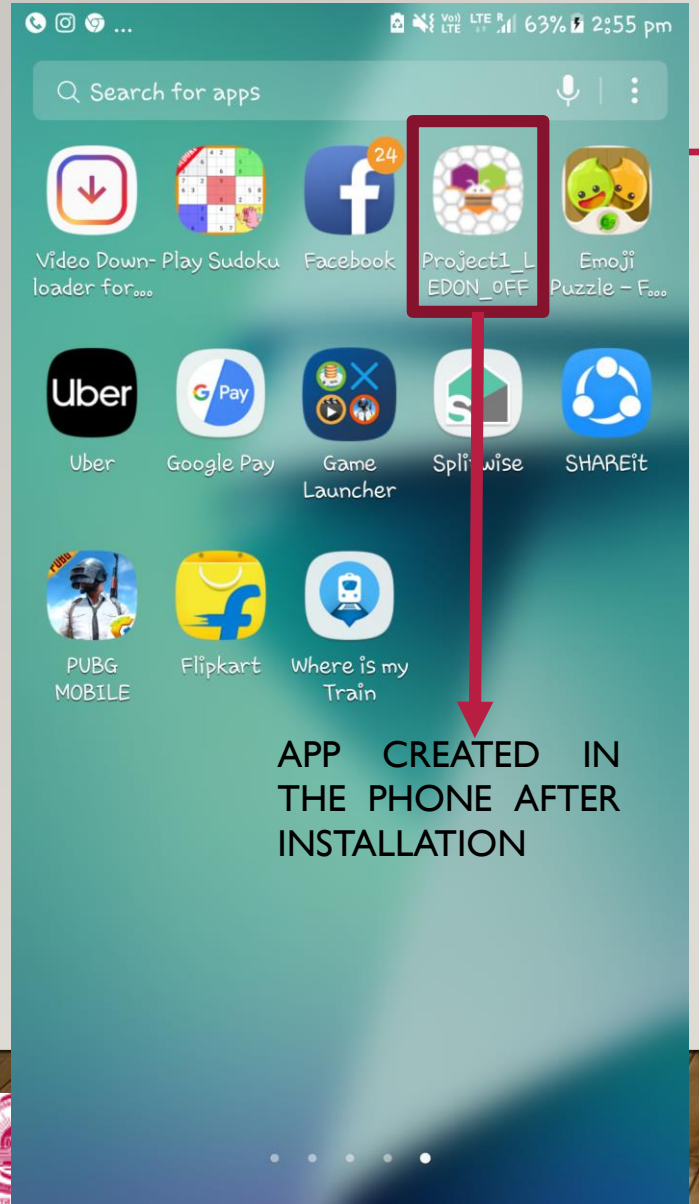


COPY THE .apk FILE TO YOUR SMARTPHONE → OPEN THE FILE MANAGER AND INSTALL THE APP → OPEN YOUR APP

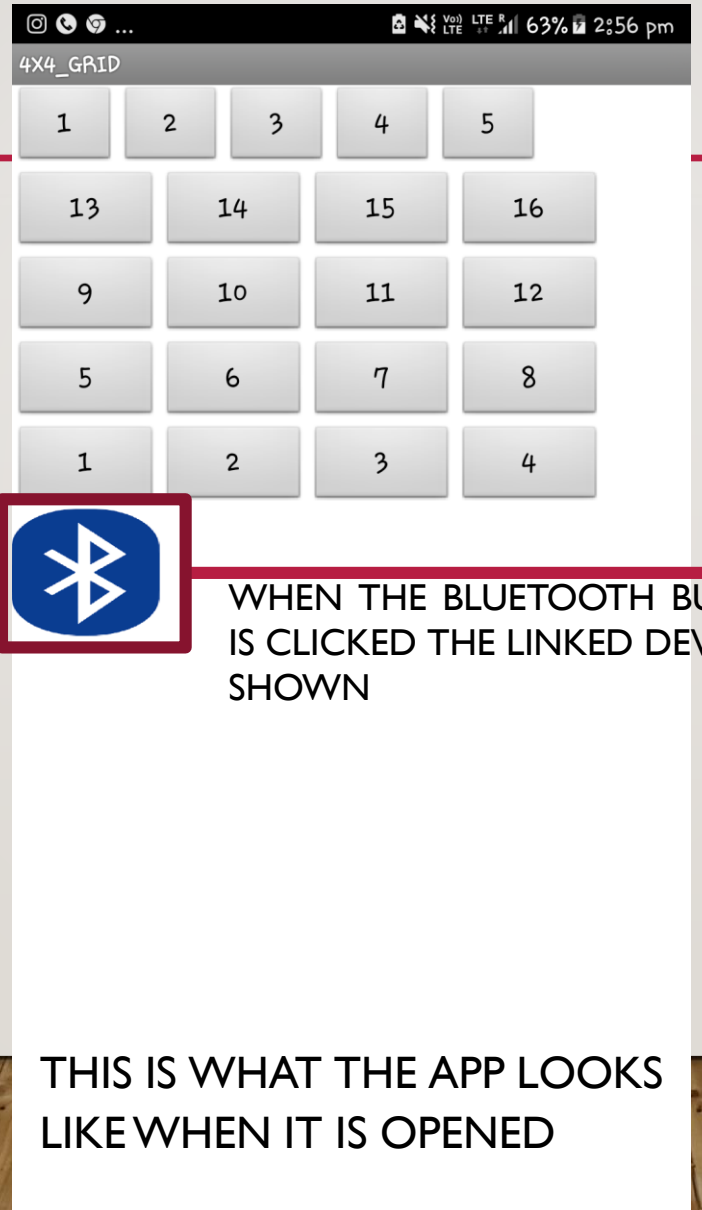
DEVELOPMENT OF ANDROID BASED BLUETOOTH APP

APP DEVELOPMENT

- SCREENSHOTS OF THE APP:

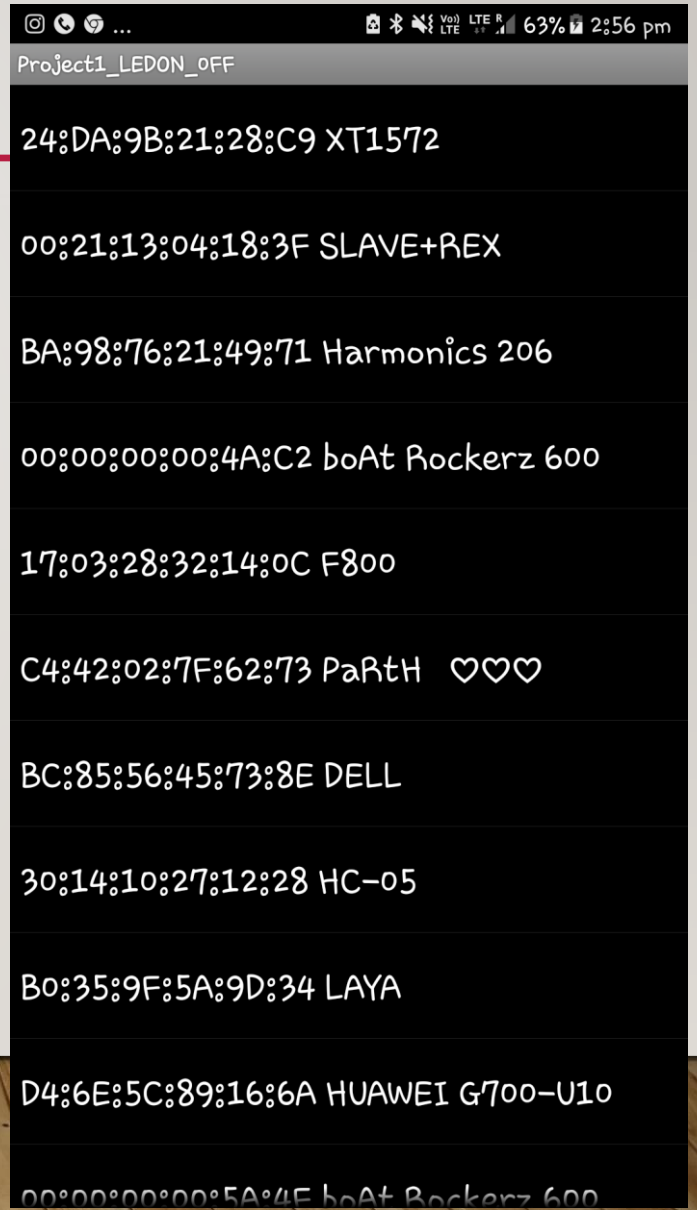


APP CREATED IN THE PHONE AFTER INSTALLATION



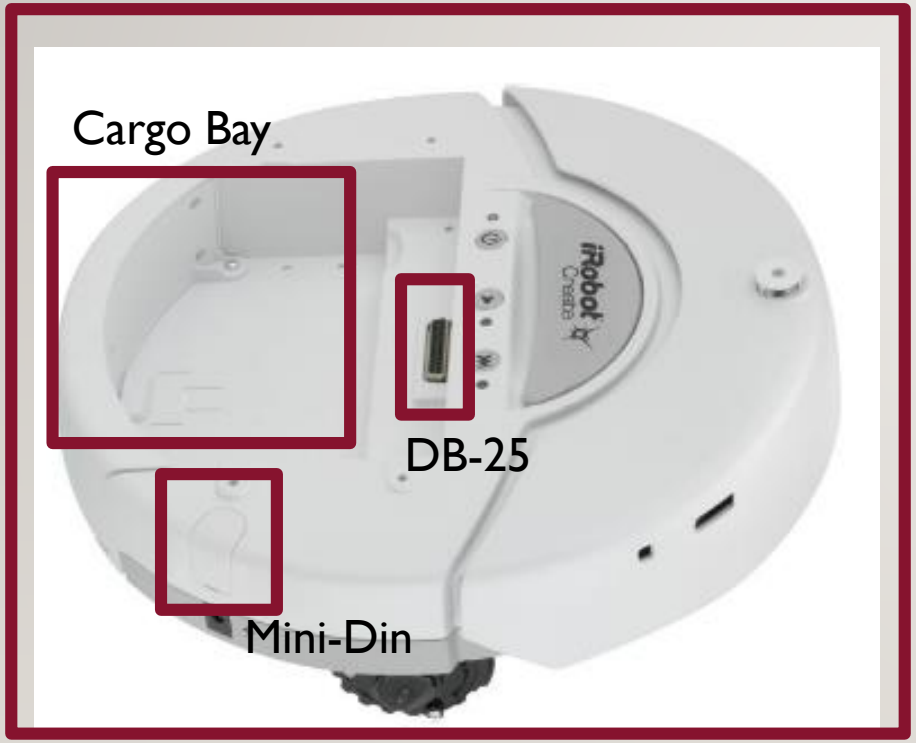
WHEN THE BLUETOOTH BUTTON IS CLICKED THE LINKED DEVICES IS SHOWN

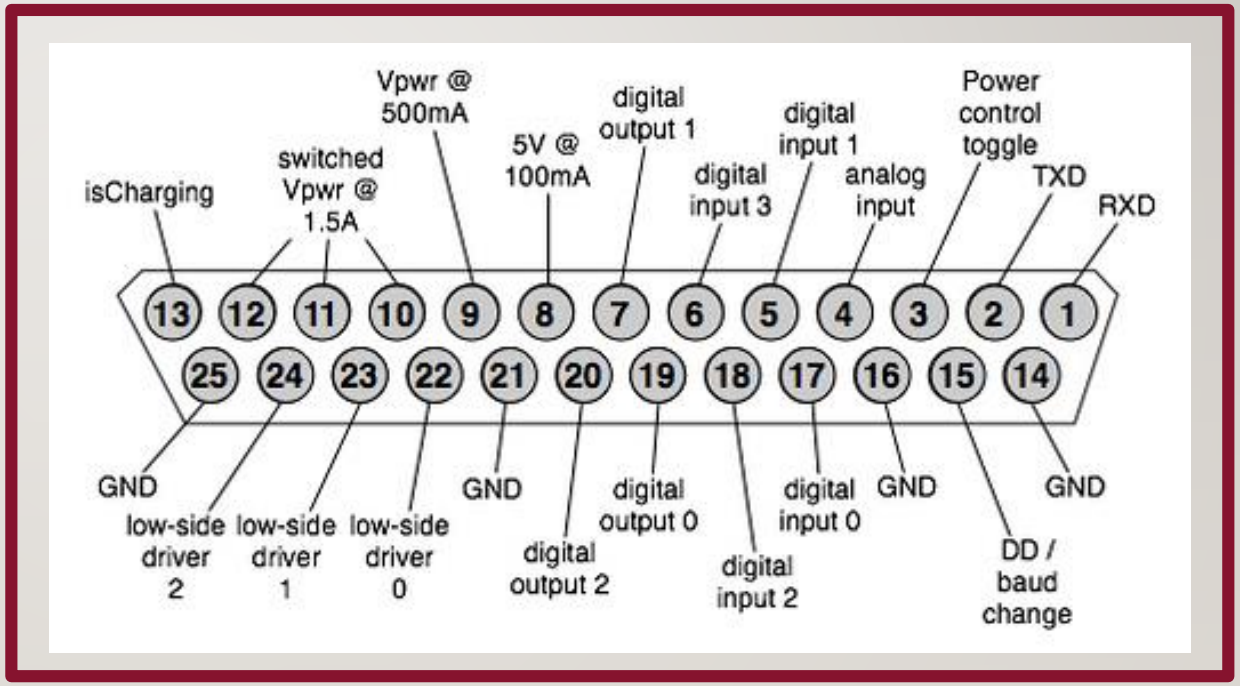
THIS IS WHAT THE APP LOOKS LIKE WHEN IT IS OPENED



STUDY OF IROBOT

FEATURES OF IROBOT





STUDY OF IROBOT

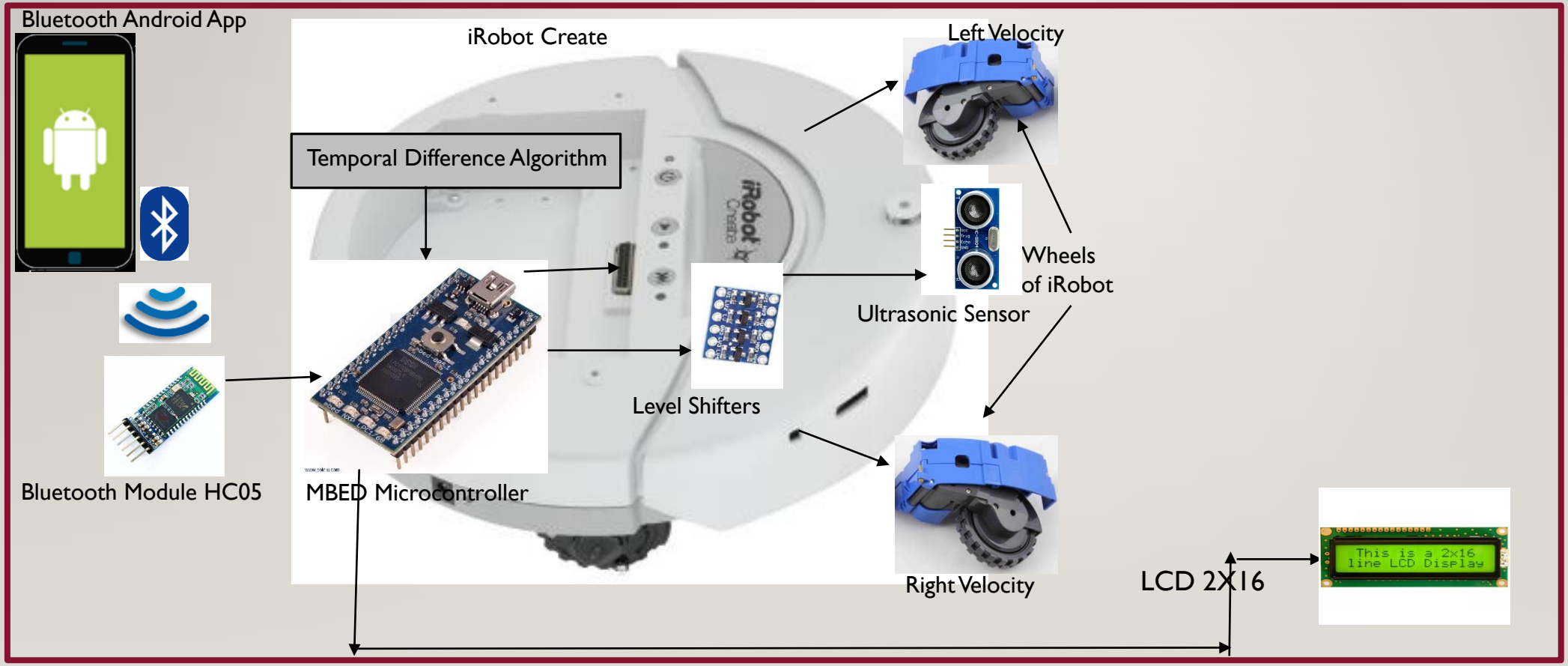
COMMANDS OF IROBOT

OPEN INTERFACE COMMANDS OF IROBOT

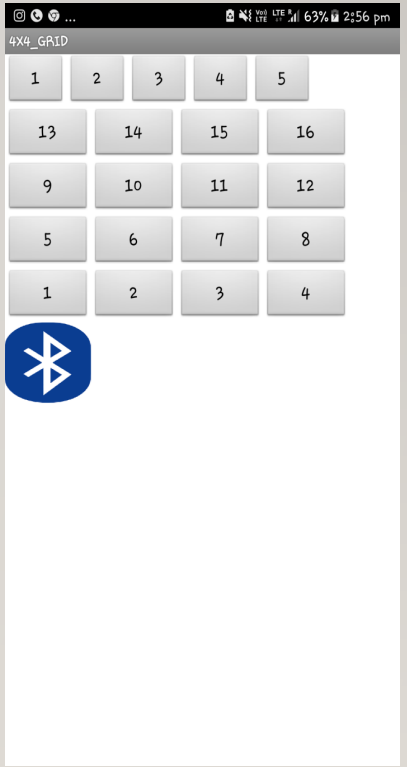
- 128: Start
- 131:Safe
- 137:Drive
- 145:Drive Direct
- 148:Sensor Stream

BLOCK DIAGRAM OF TEMPORAL DIFFERENCE LEARNING ALGORITHM FOR A 4X4 GRID

a. System Layout



b. App Layout



IMPLEMENTATION OF TD ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

c. TD Algorithm

Priority of Movements :

SL.NO	DIRECTION OF MOVEMENT	PRIORITY
1	DIAGONAL	1
2	VERTICAL FORWARD	2
3	HORIZONTAL RIGHT	3

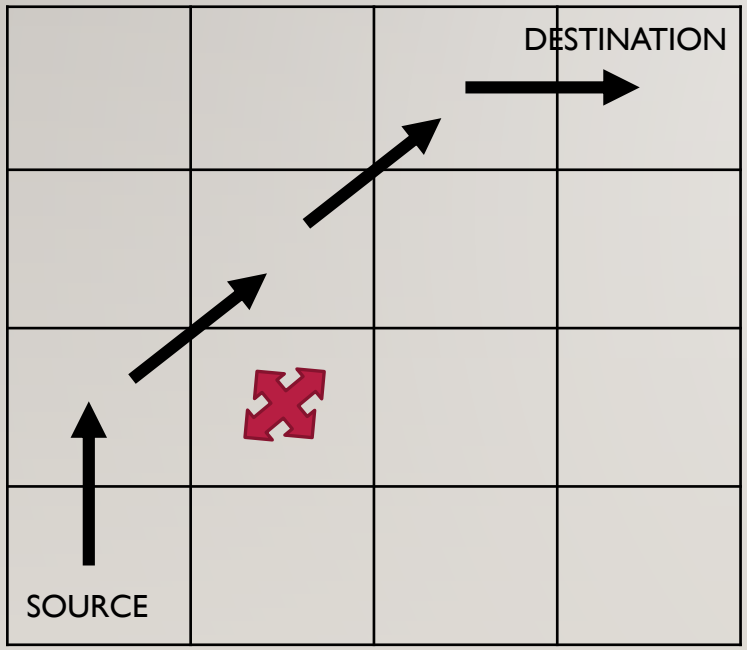
ASSUMPTIONS:

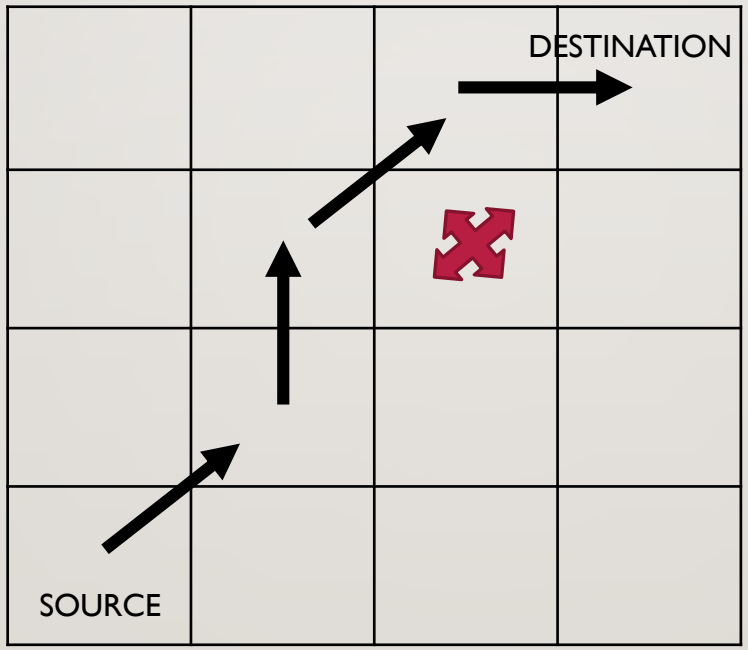
1. Robot is restricted from moving backwards
2. Obstacles are assumed to be at the centre of the grid

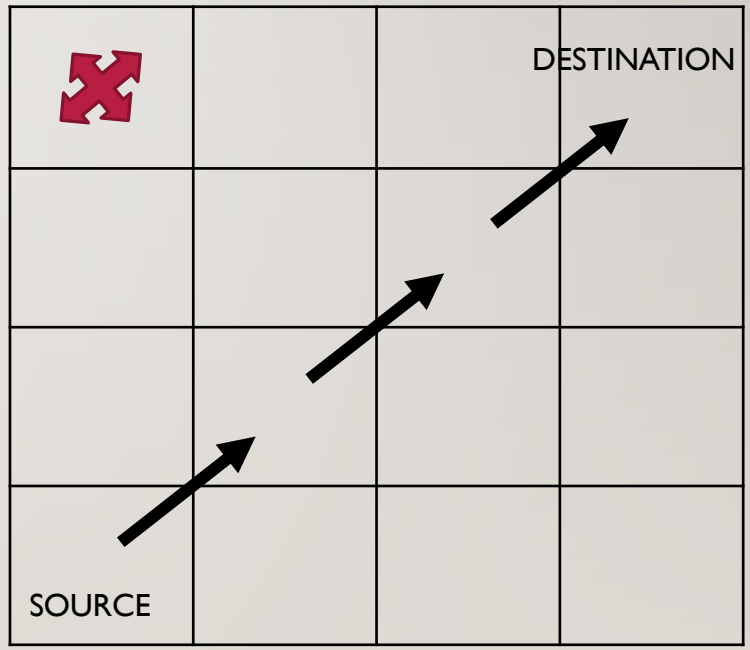
IMPLEMENTATION OF TD ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

d. Cases tried for TD algorithm in Hardware for 1-5 obstacles

Path Planning with one obstacle



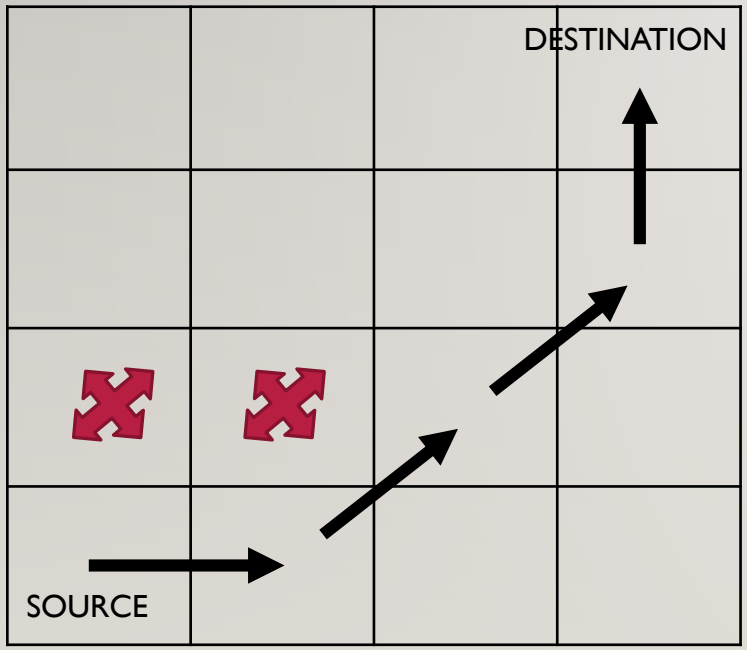


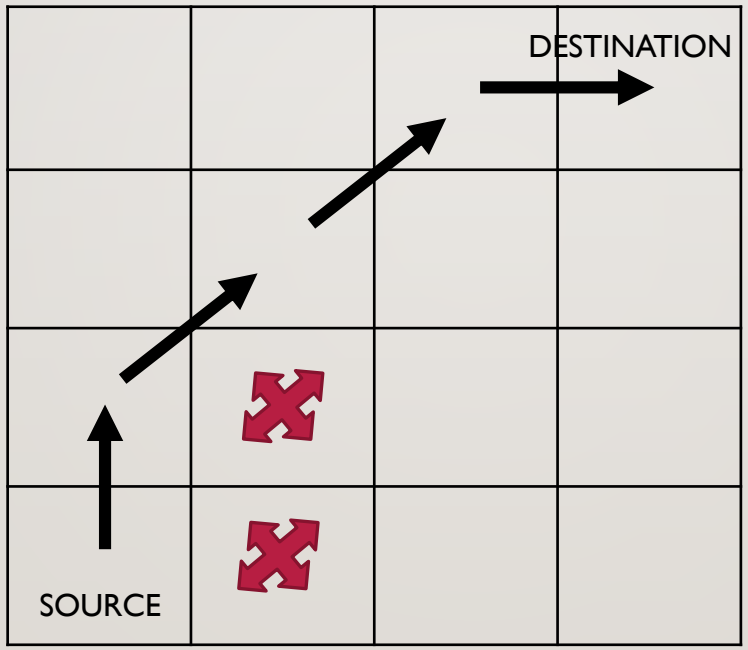


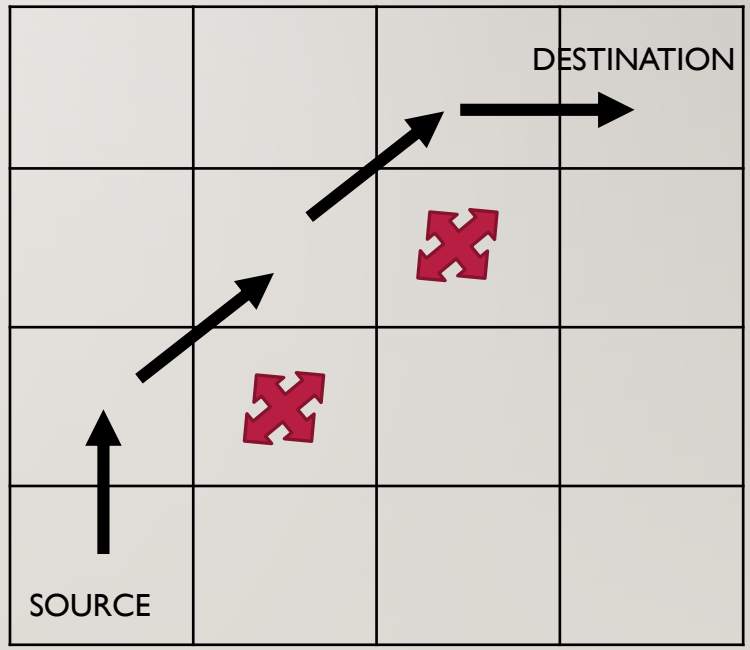
IMPLEMENTATION OF TD ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

d. Cases tried for TD algorithm in Hardware for 1-5 obstacles

Path Planning with two obstacle



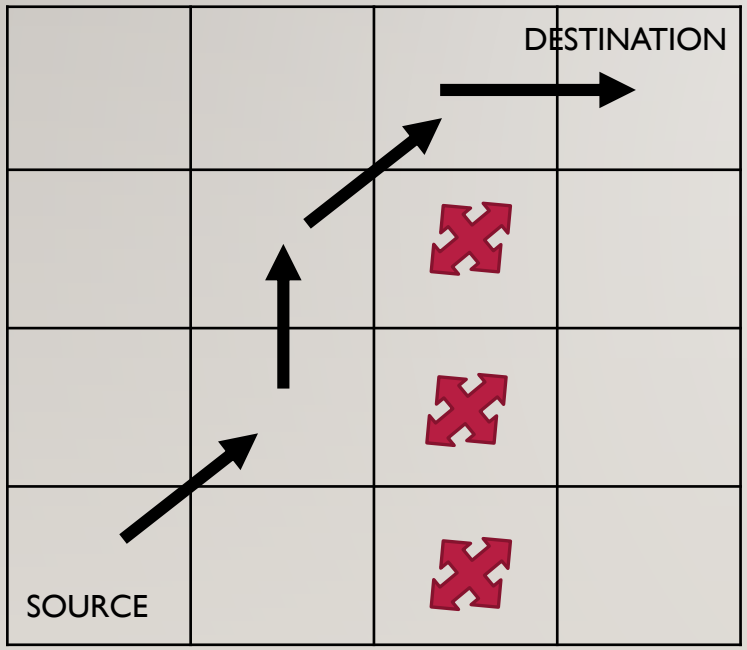


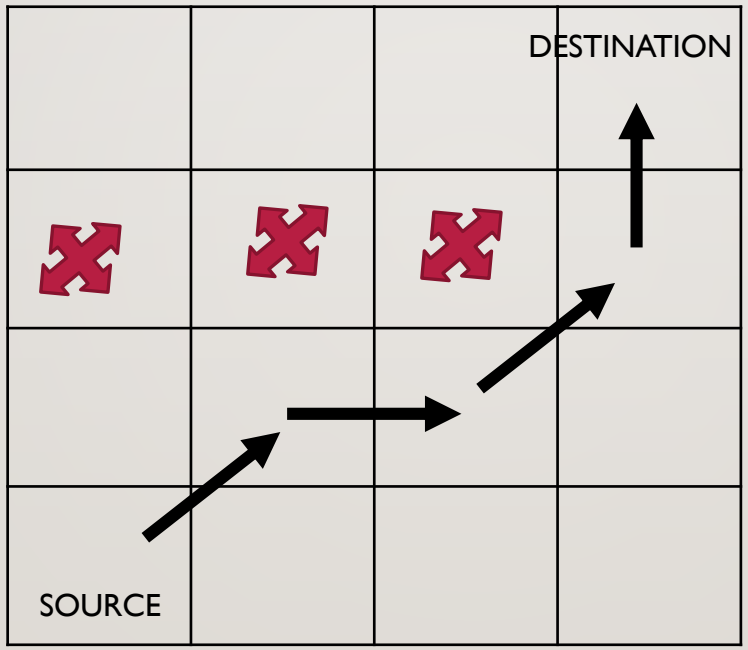


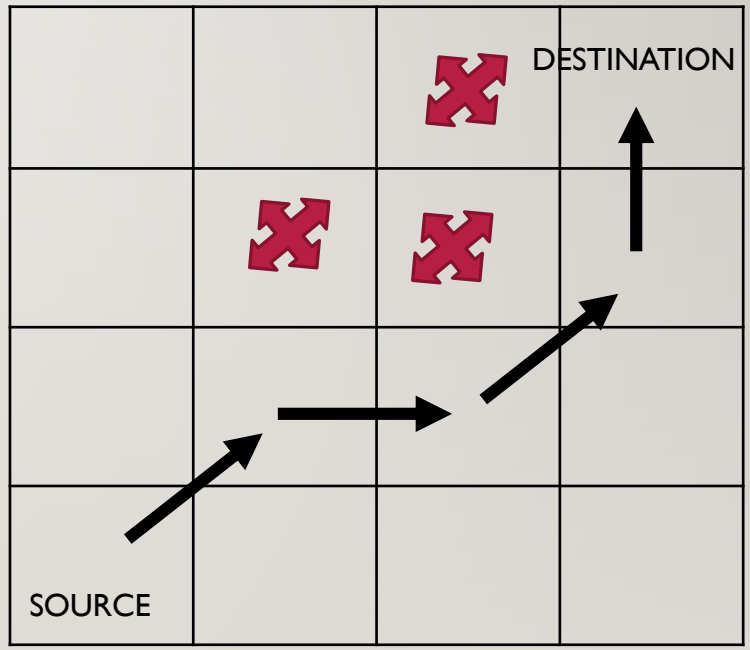
IMPLEMENTATION OF TD ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

d. Cases tried for TD algorithm in Hardware for 1-5 obstacles

Path Planning with three obstacle



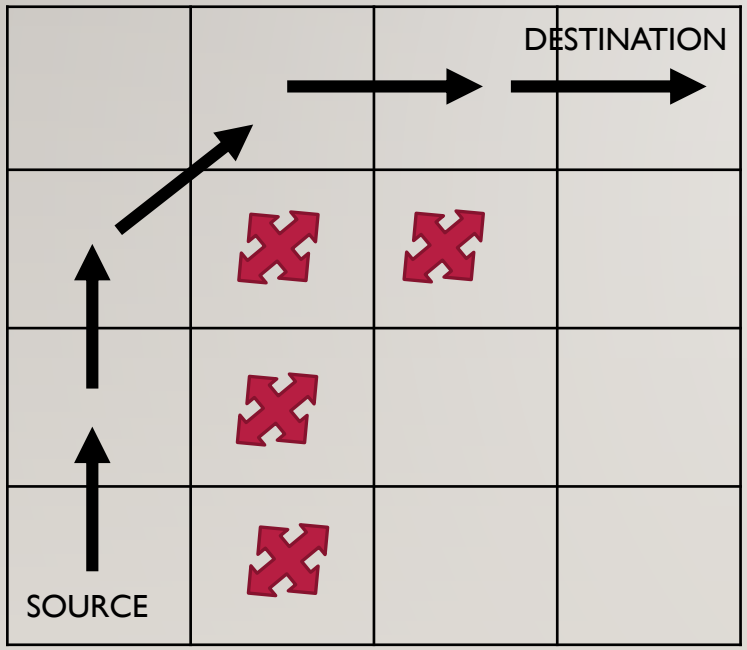


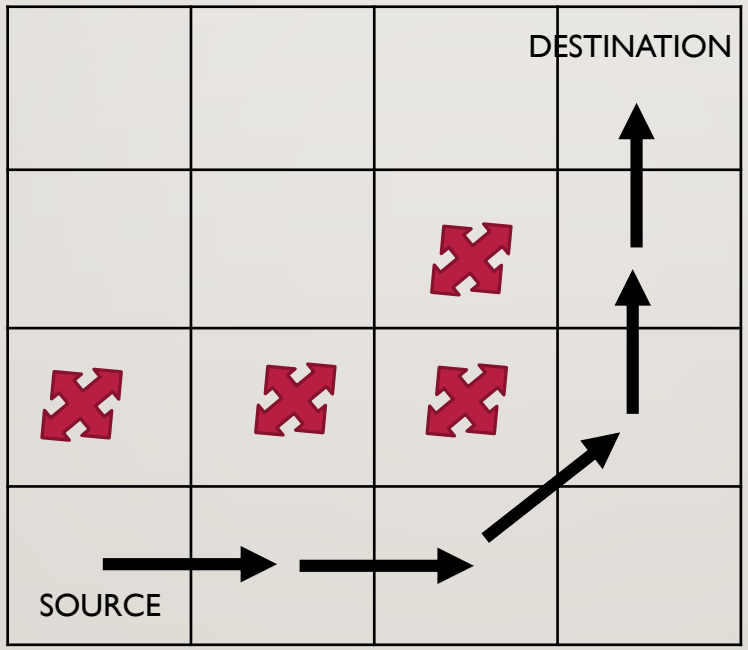


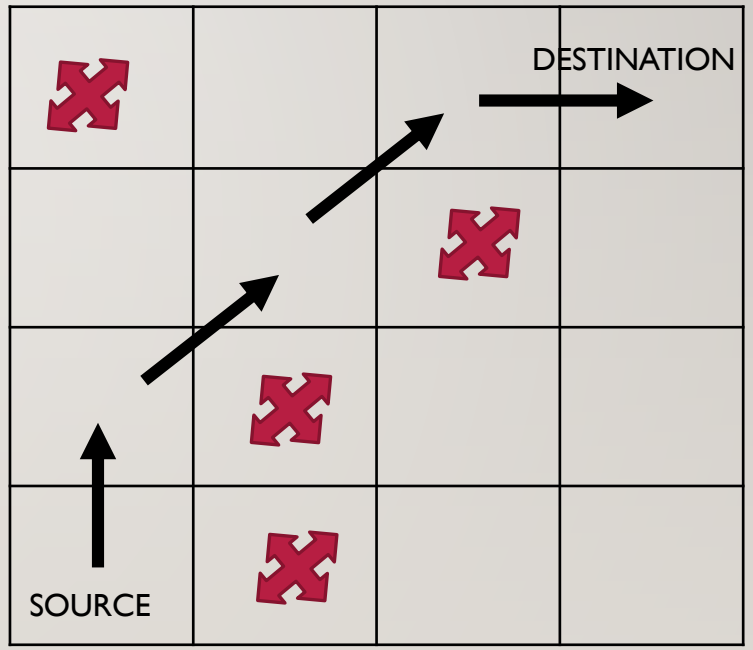
IMPLEMENTATION OF TD ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

d. Cases tried for TD algorithm in Hardware for 1-5 obstacles

Path Planning with four obstacle



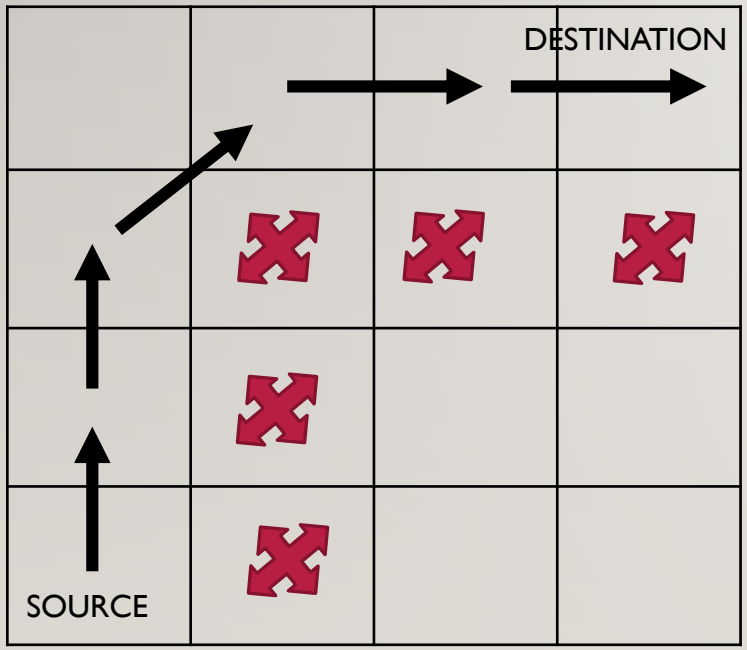


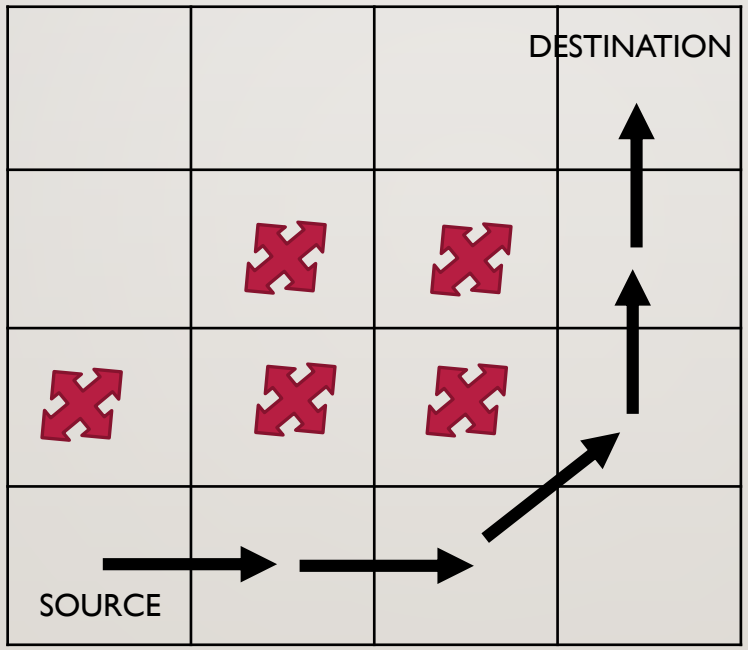


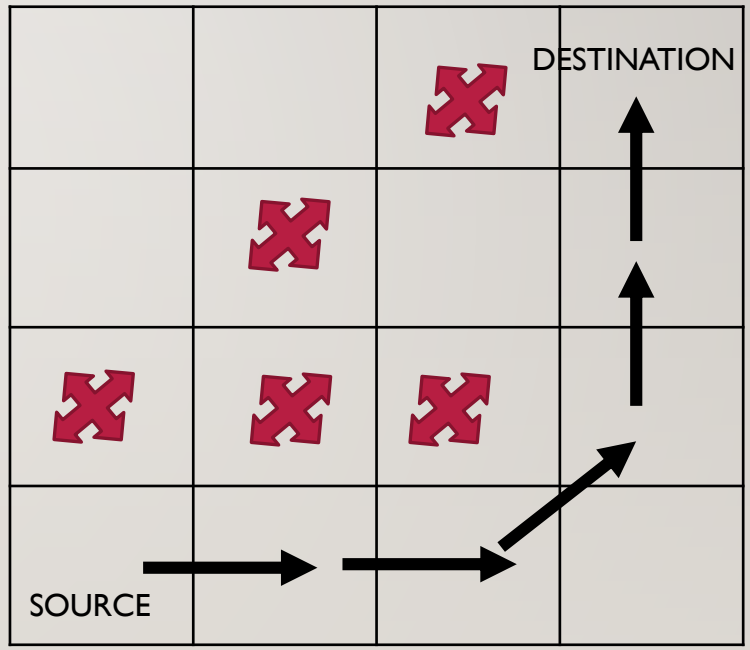
IMPLEMENTATION OF TD ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

d. Cases tried for TD algorithm in Hardware for 1-5 obstacles

Path Planning with five obstacle

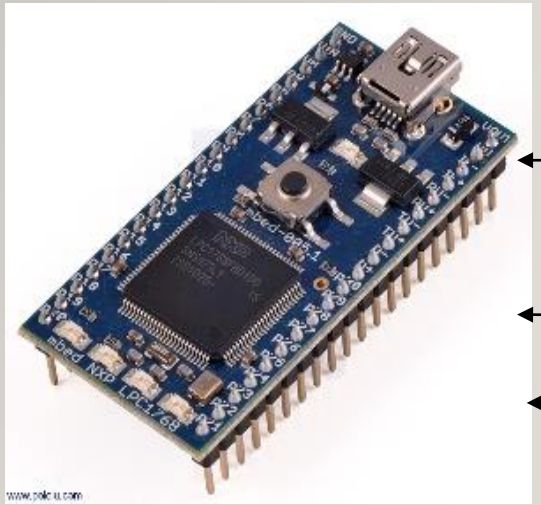






IMPLEMENTATION OF TD ALGORITHM FOR DYNAMIC OBSTACLES

a. System Layout



MBED Microcontroller

5V

GND

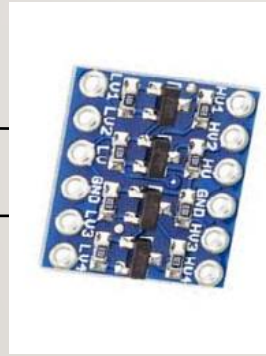
GND

p6

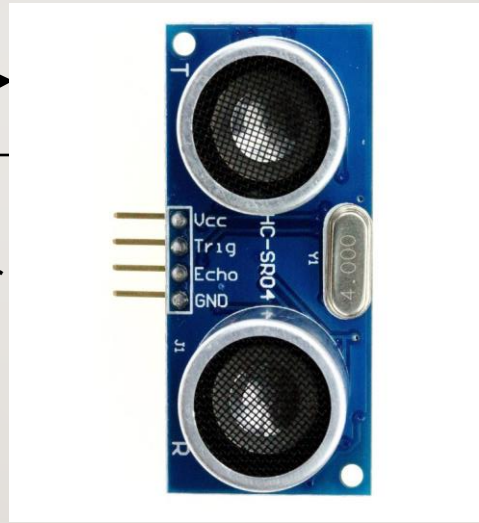
p7

Trigger

Echo



Level Shifters



Ultrasonic Sensor

b. Position of Ultrasonic Sensor in iRobot

iRobot Create

Ultrasonic Sensor

Ultrasonic sensor is placed in the front side of the robot so that it can detect the obstacles ahead of it

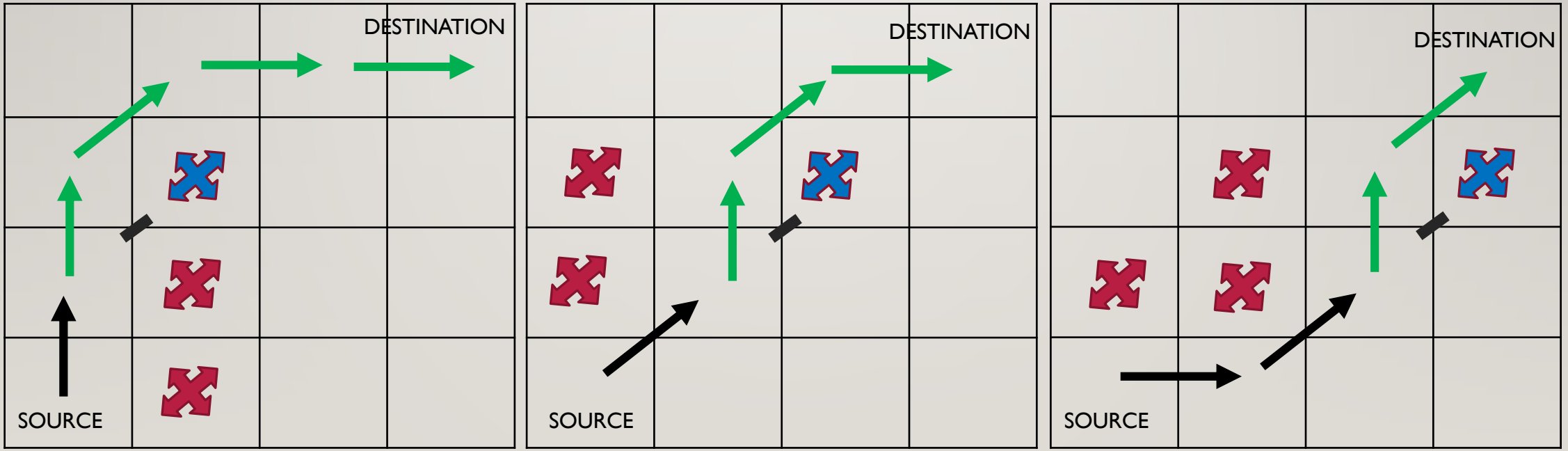
IMPLEMENTATION OF TD ALGORITHM FOR DYNAMIC OBSTACLES

c. Cases tried for TD algorithm with dynamic obstacles in Hardware

Path Planning with two or three static obstacle and one dynamic obstacle

: Path taken before the presence of dynamic obstacle, : Path taken before the presence of dynamic obstacle,

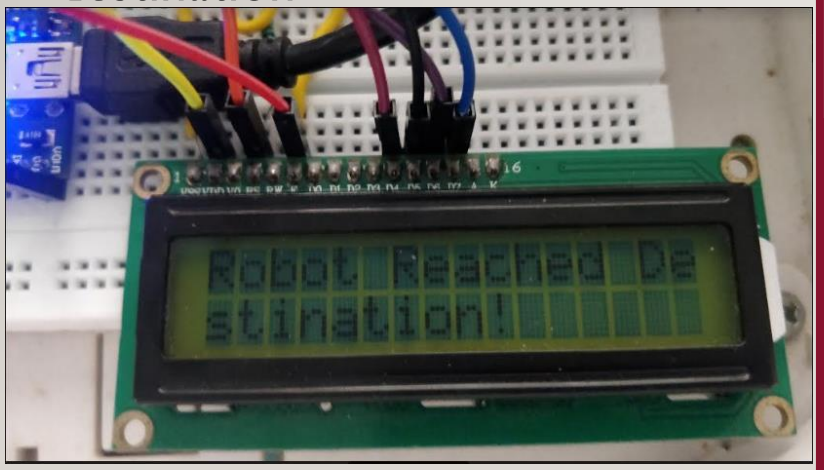
: Static Obstacles, : Dynamic Obstacles



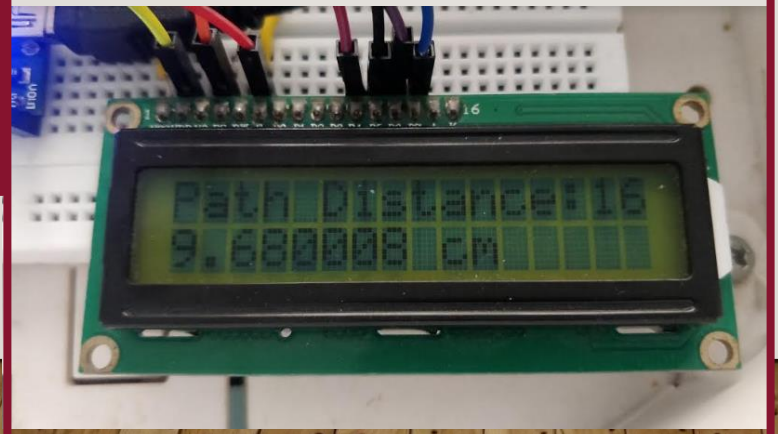
INTERFACING LCD TO MBED MICROCONTROLLER

a. Pictures of LCD outputs

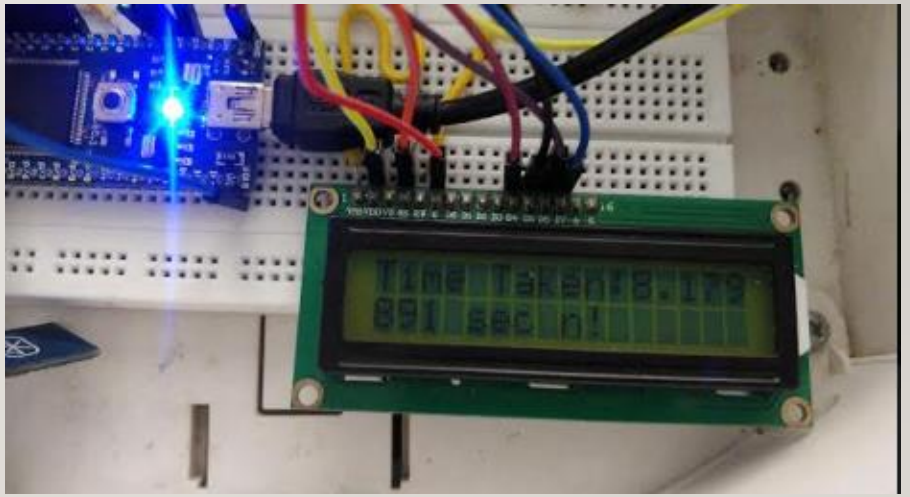
- Displays a message after reaching destination



- Displays the total distance travelled in cms.



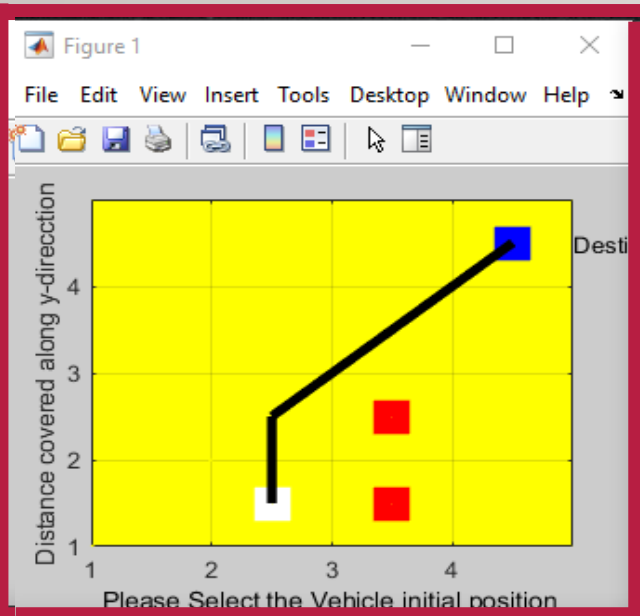
- Displays the time taken to travel

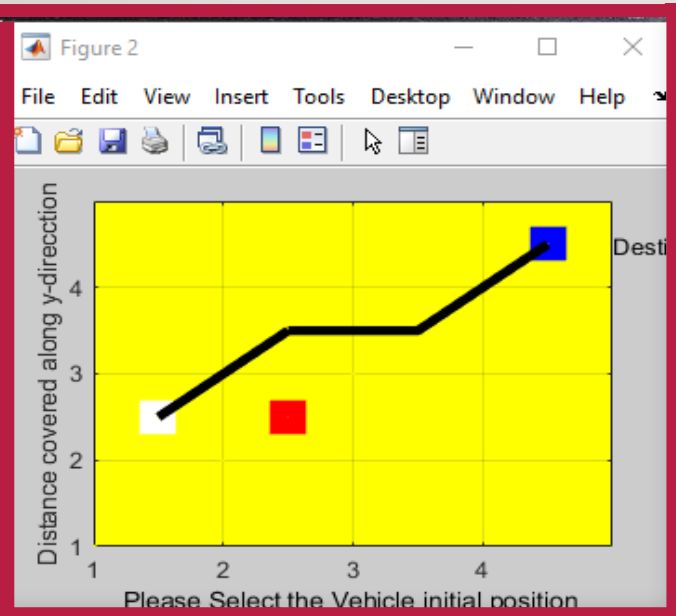


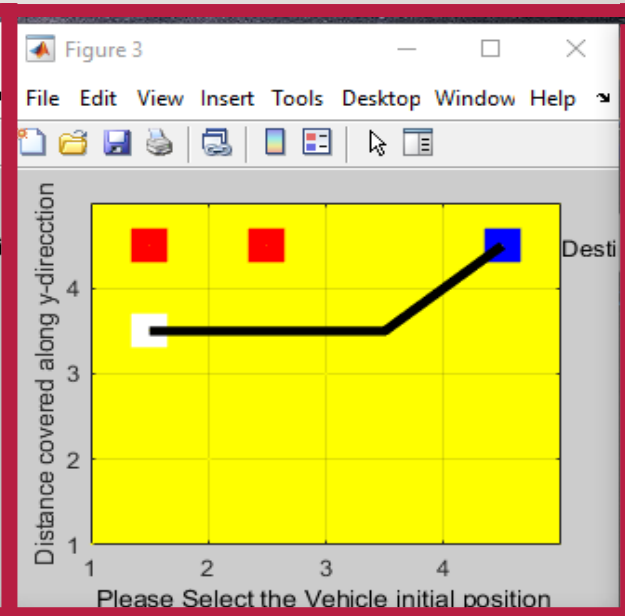
IMPLEMENTATION OF SARSA ALGORITHM FOR ALTERING DESTINATION AND SOURCE POINTS

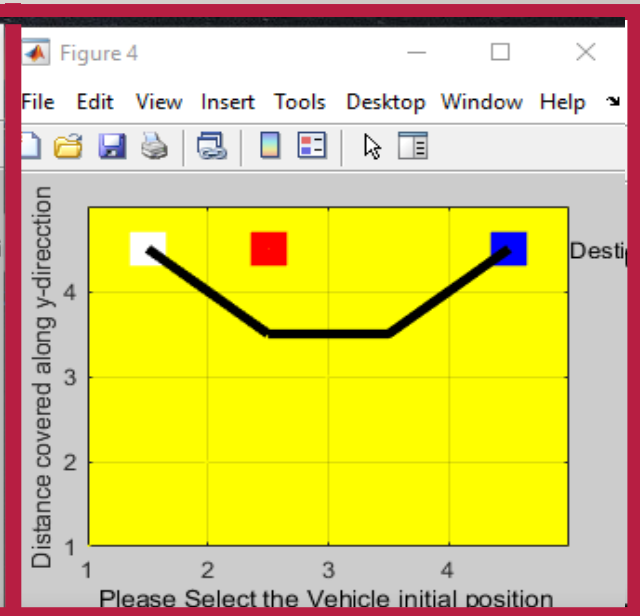
a. Altering Source Points

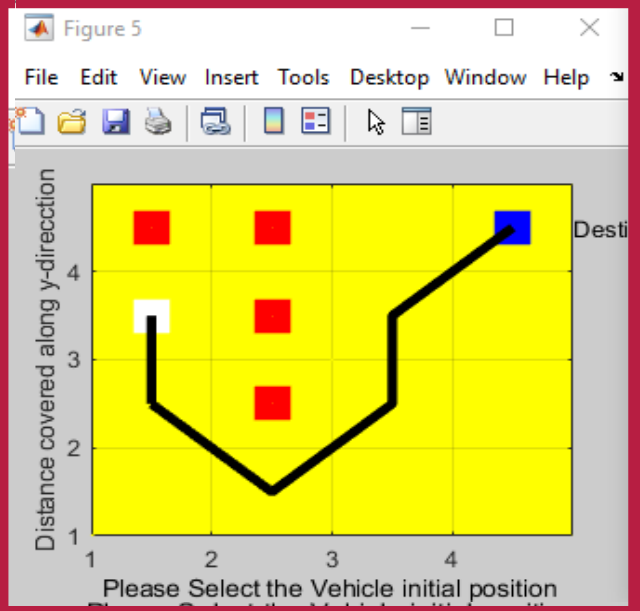
: Source : Destination : Obstacle

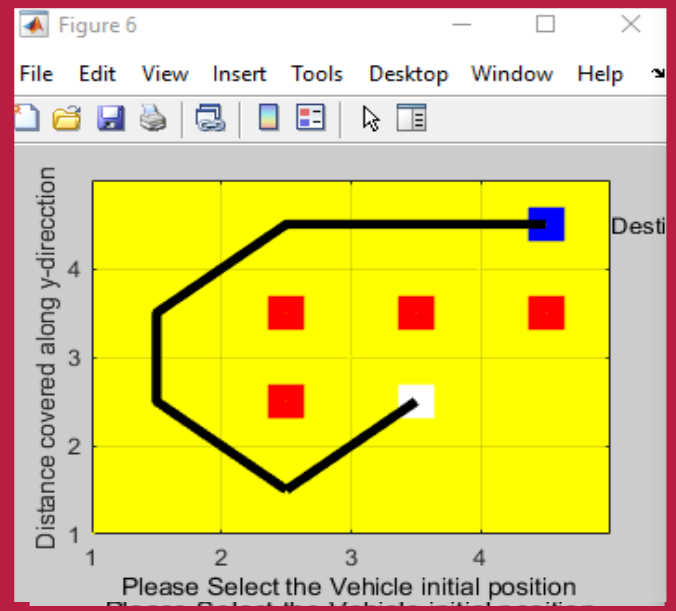


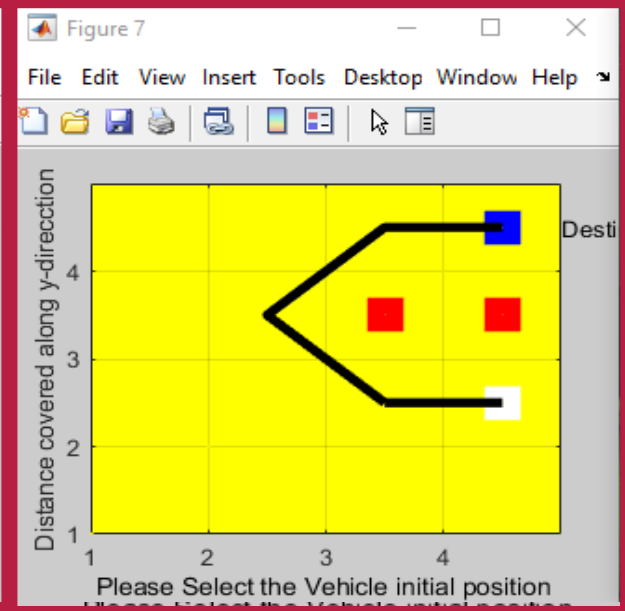


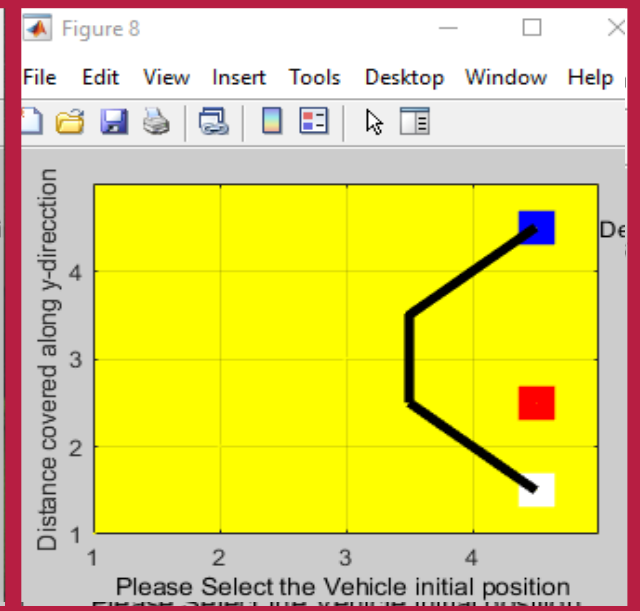








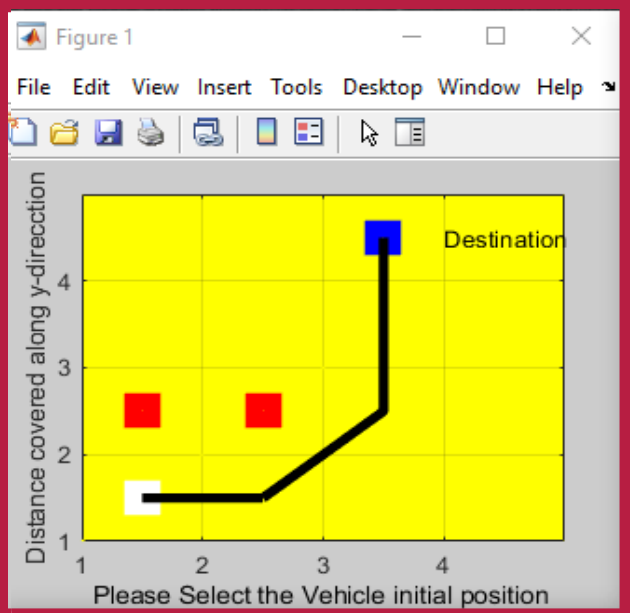


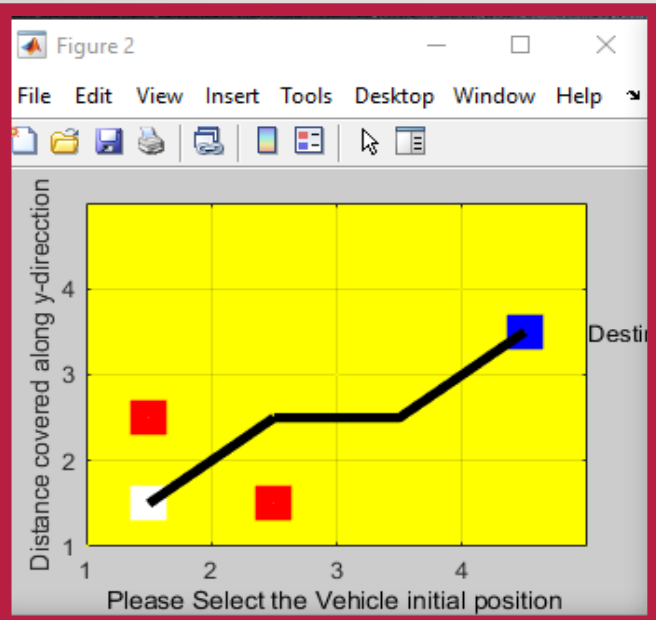


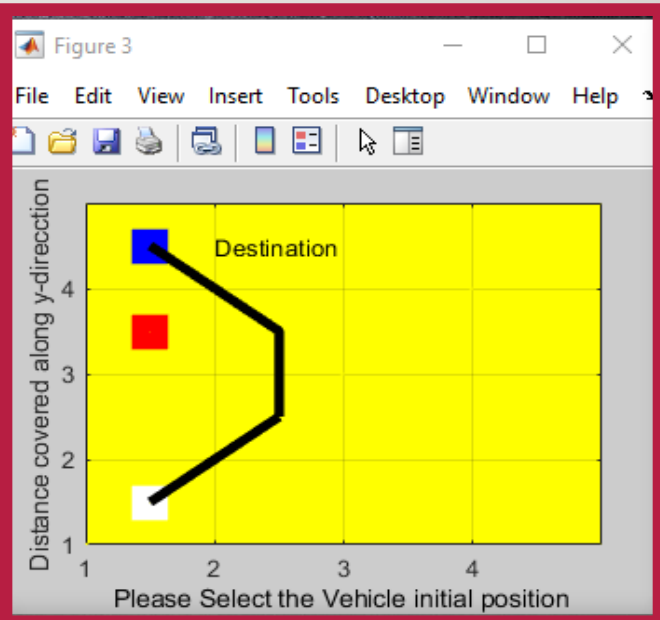
IMPLEMENTATION OF SARSA ALGORITHM FOR ALTERING DESTINATION AND SOURCE POINTS

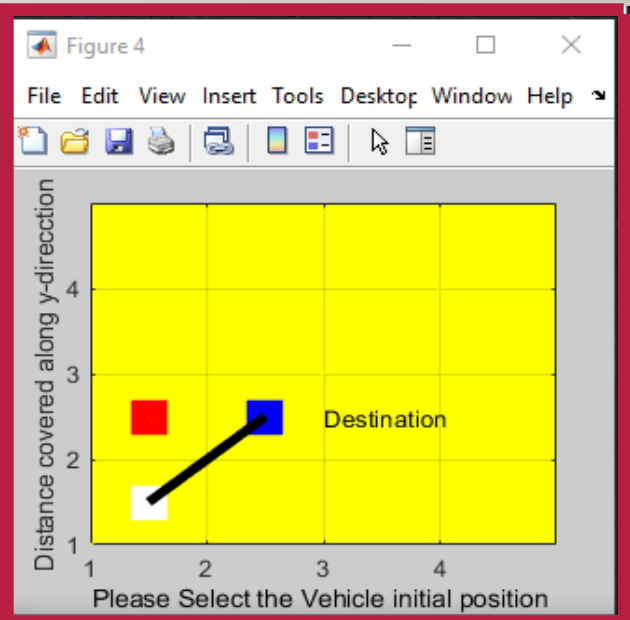
b. Altering Destination Points

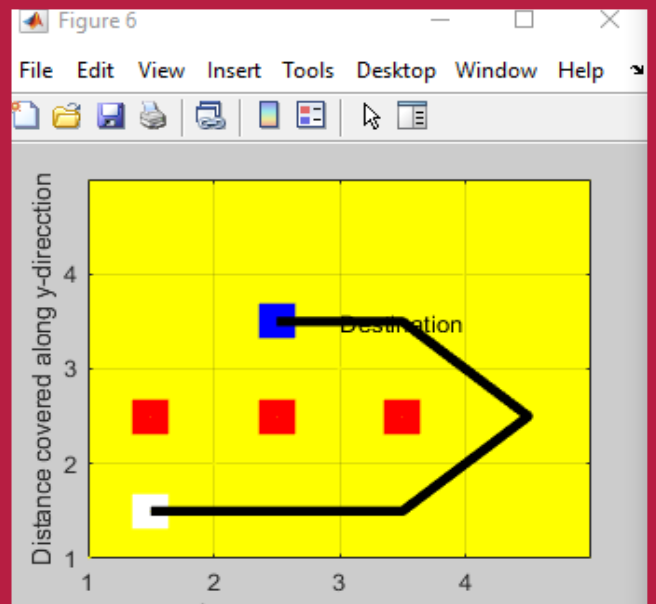
: Source : Destination : Obstacle

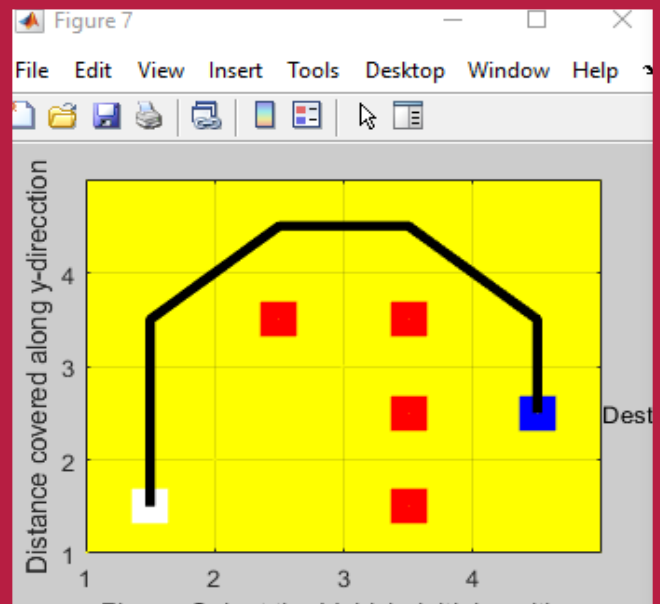


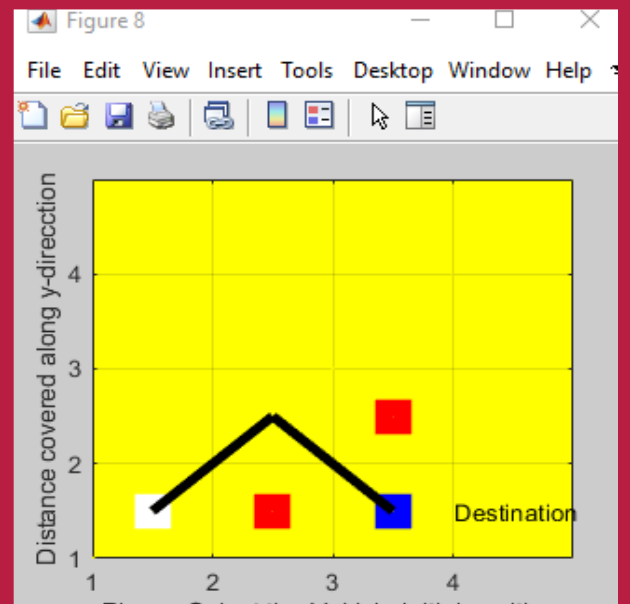








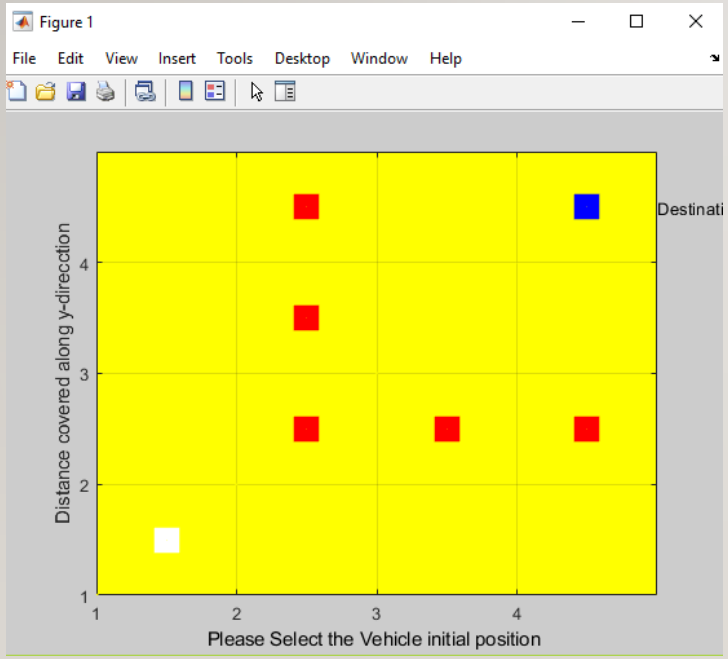




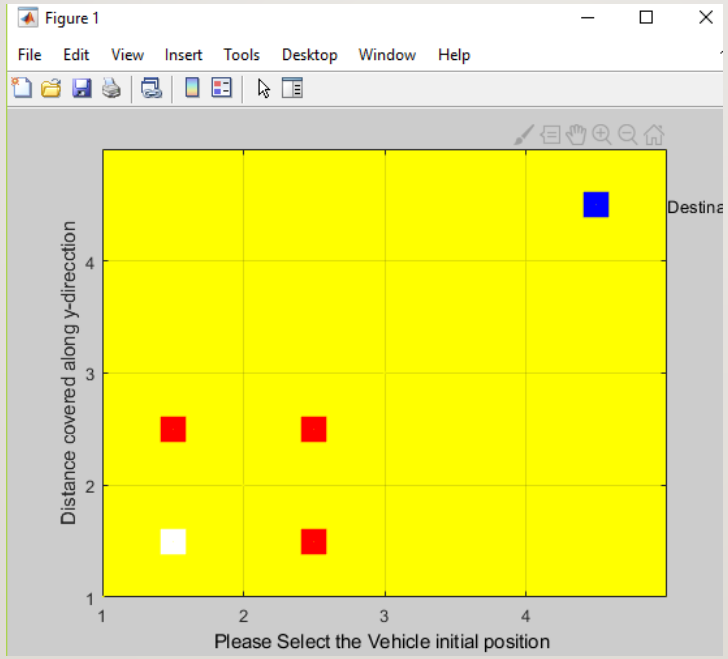
IMPLEMENTATION OF SARSA ALGORITHM FOR ALTERING DESTINATION AND SOURCE POINTS

c. Cases Where Algorithm Fails

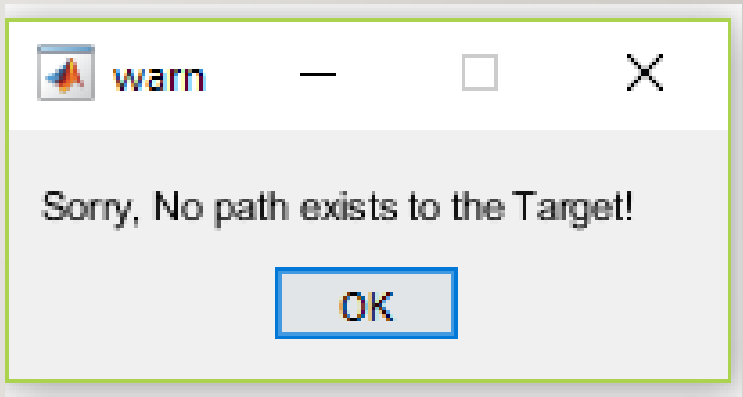
: Source : Destination : Obstacle



When destination point is covered with obstacles



When source point is covered with obstacles

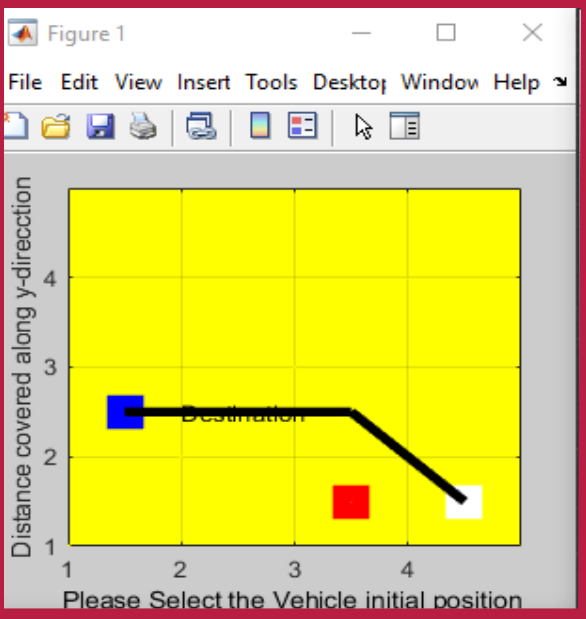


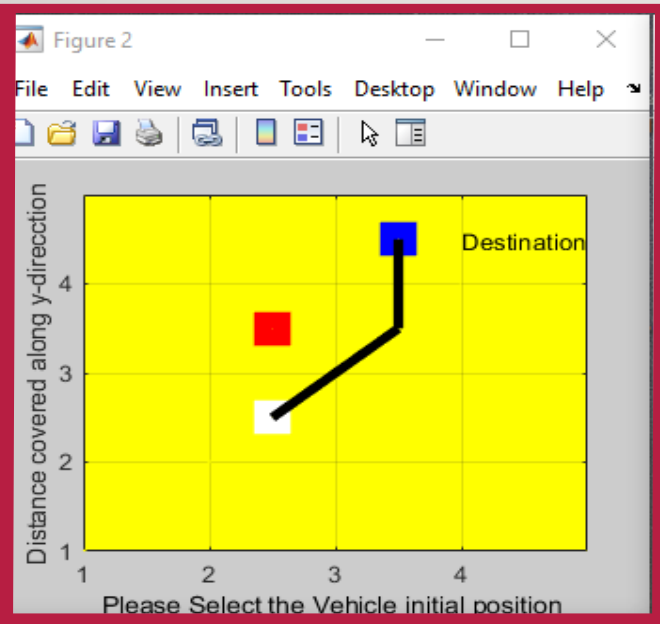
When there is no path between source and destination, then a warning message will be displayed stating that there is no path

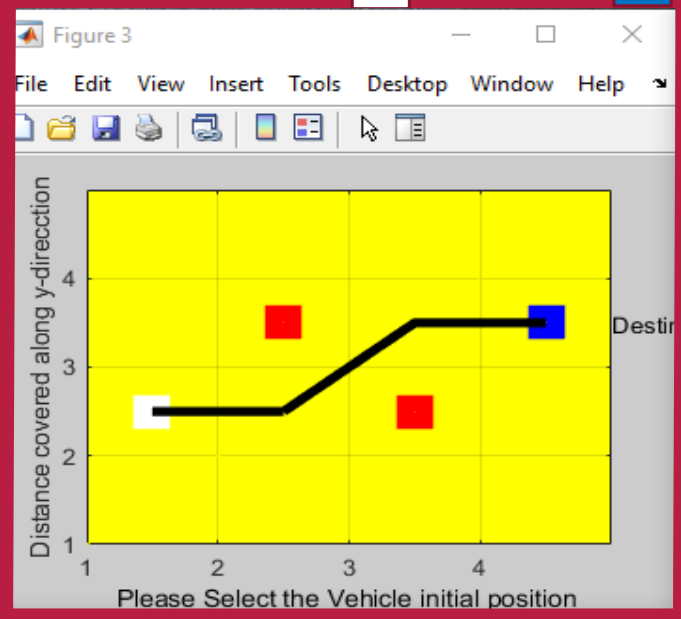
IMPLEMENTATION SARSA ALGORITHM FOR ALTERING DESTINATION AND SOURCE POINTS

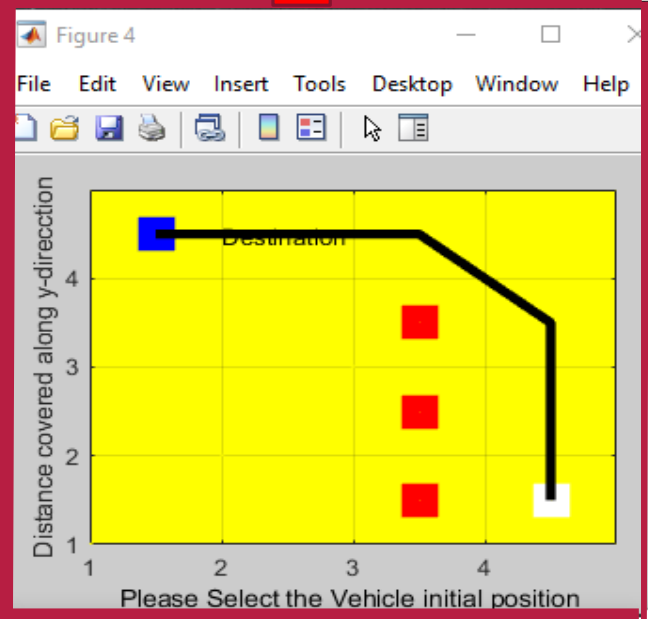
d. Altering Both Source And Destination Points

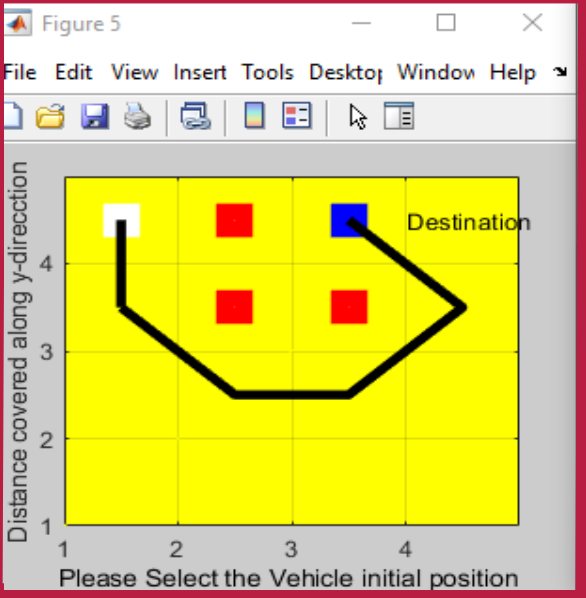
: Source : Destination : Obstacle

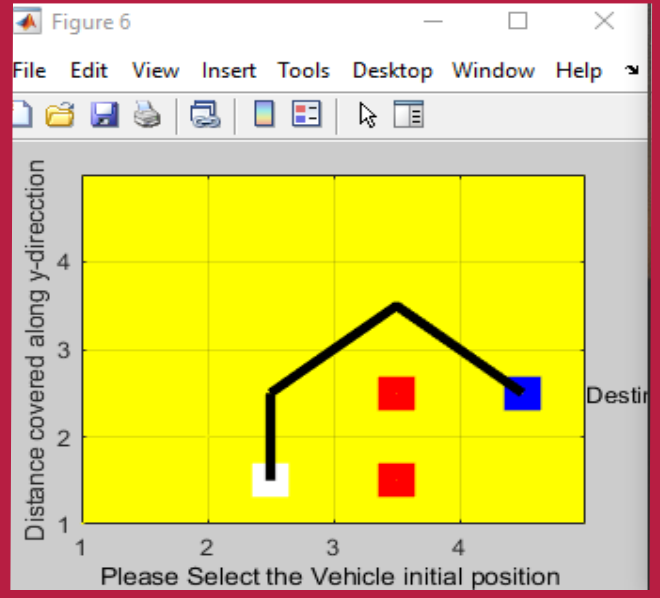


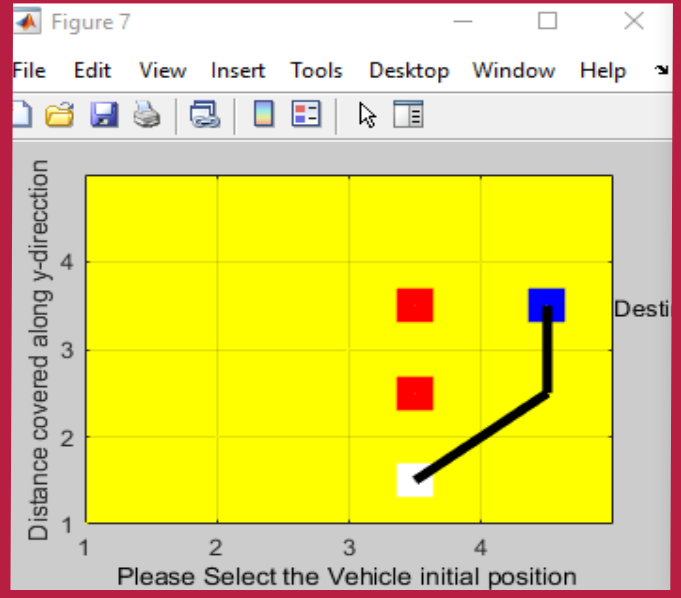


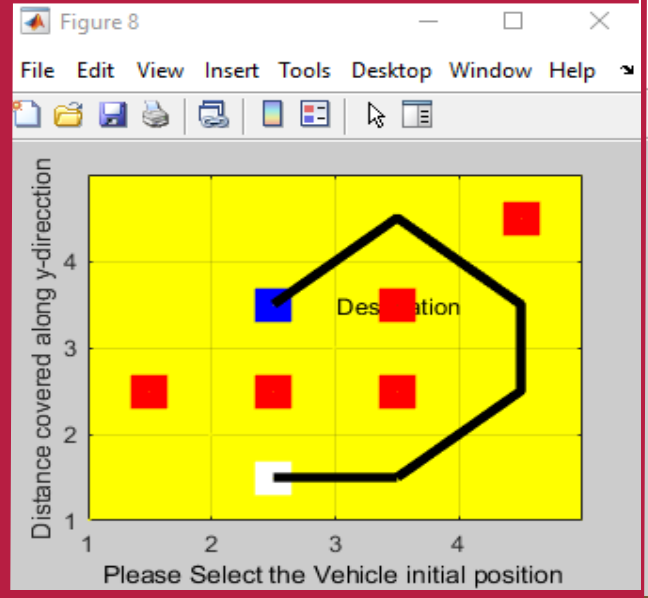






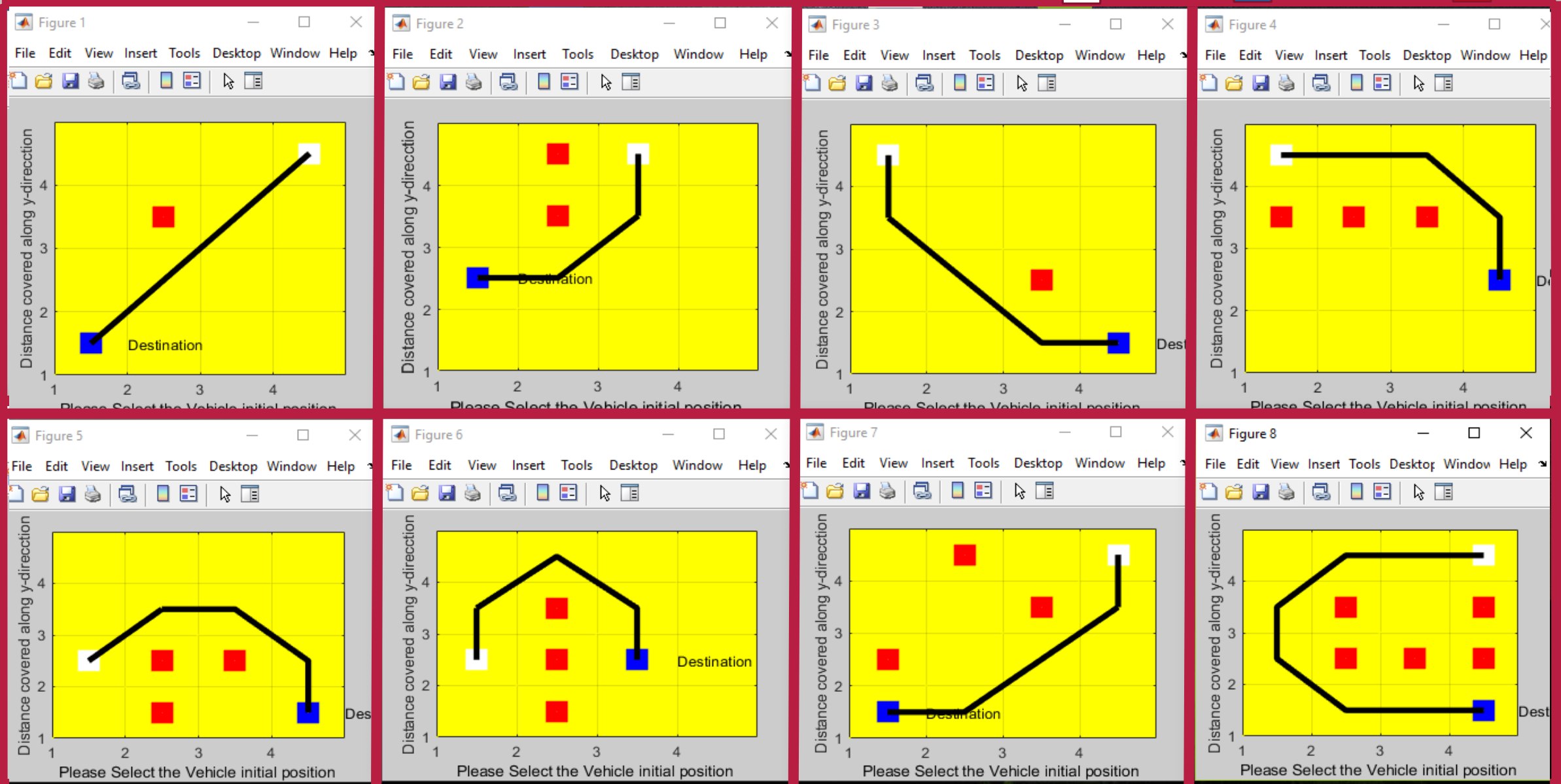






IMPLEMENTED SARSA ALGORITHM WHERE SOURCE POINT IN THE TOP PORTION OF THE GRID AND DESTINATION POINT IN THE LOWER SECTION OF THE GRID

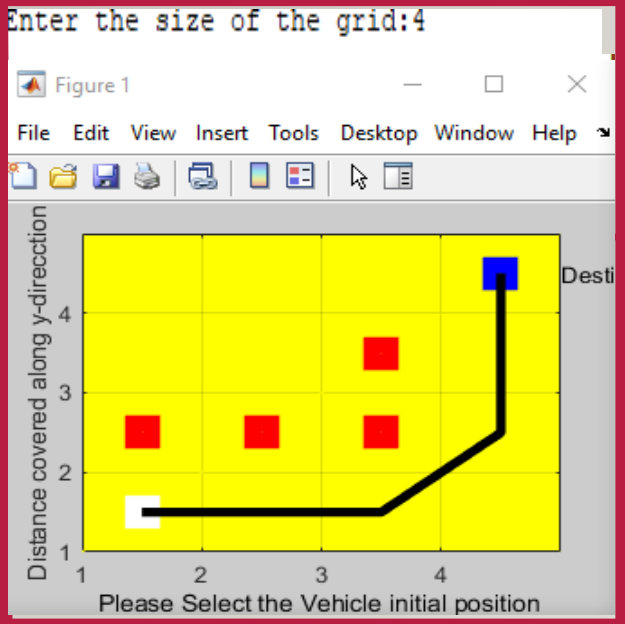
: Source : Destination : Obstacle



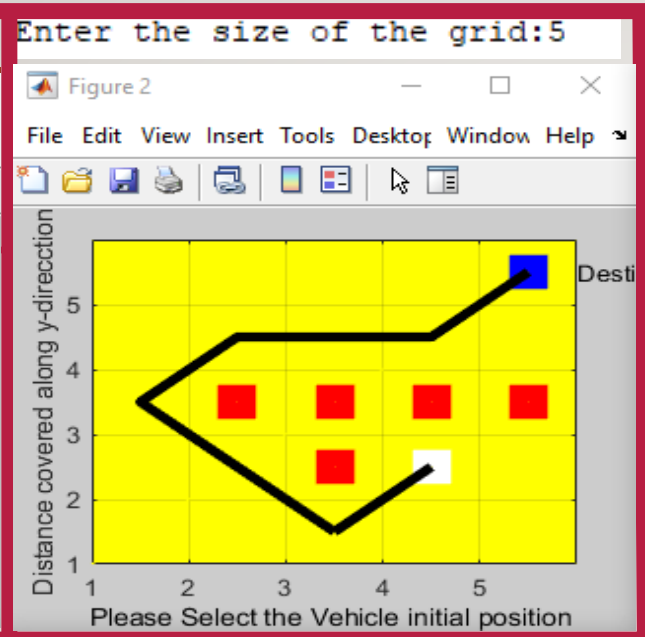
GENERALISED CODE FROM 4X4 GRID TO 20X20 GRID

: Source : Destination : Obstacle

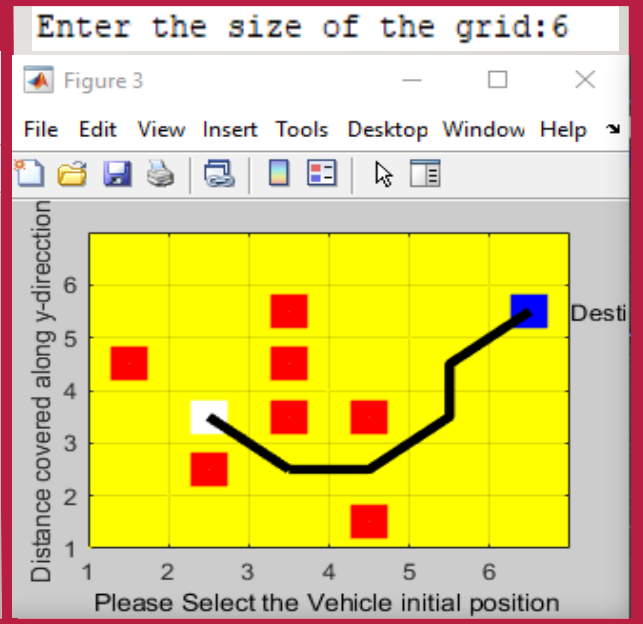
Enter the size of the grid:4



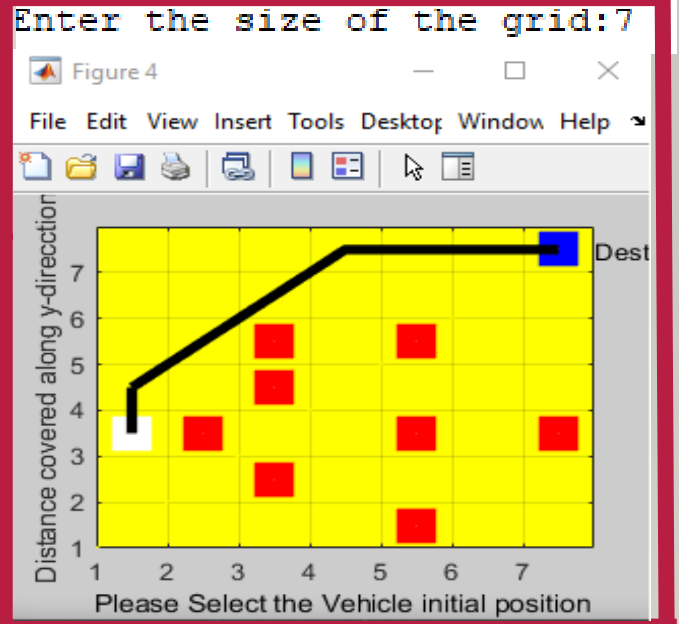
Enter the size of the grid:5



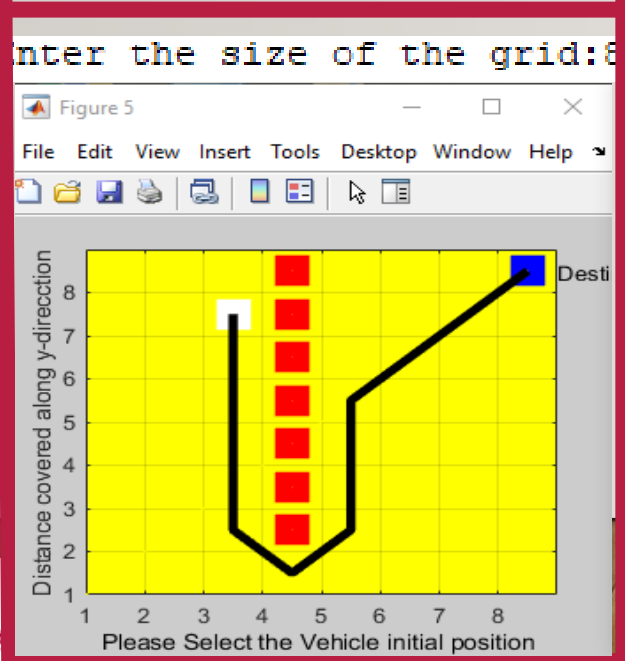
Enter the size of the grid:6



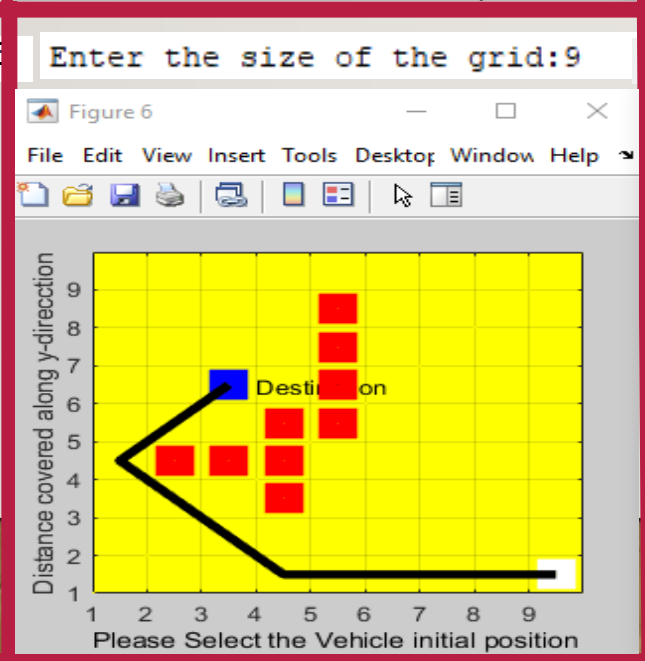
Enter the size of the grid:7



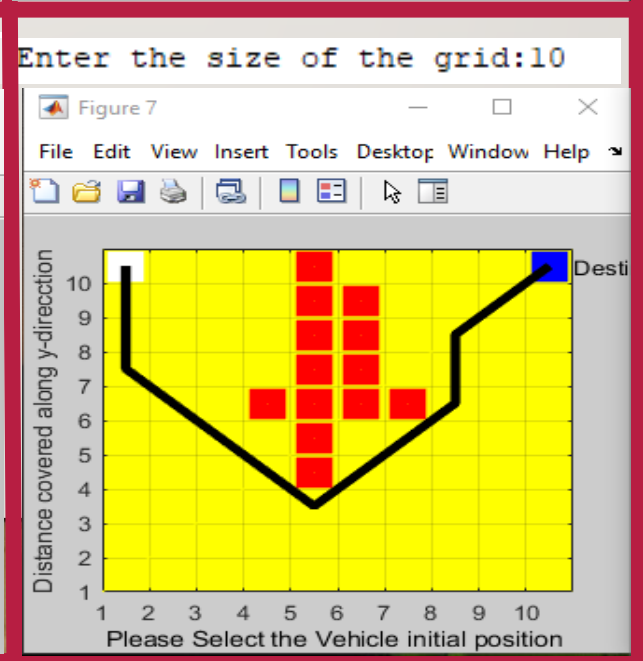
nter the size of the grid:8



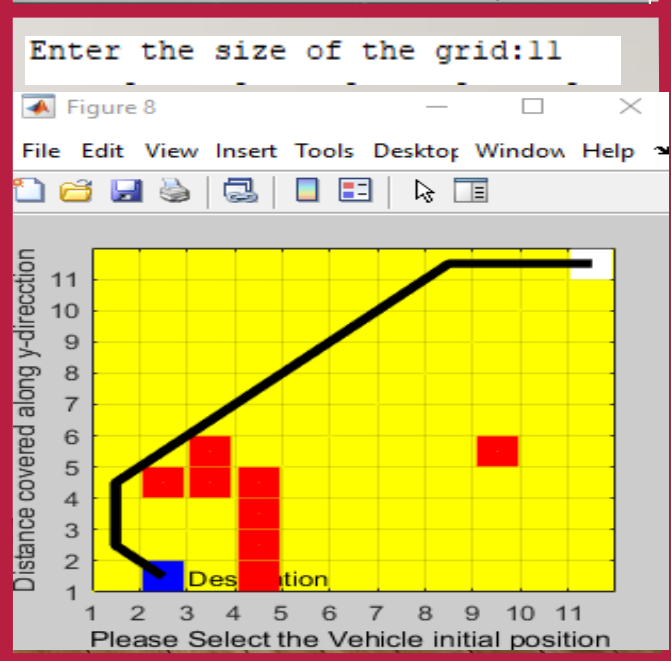
Enter the size of the grid:9



Enter the size of the grid:10



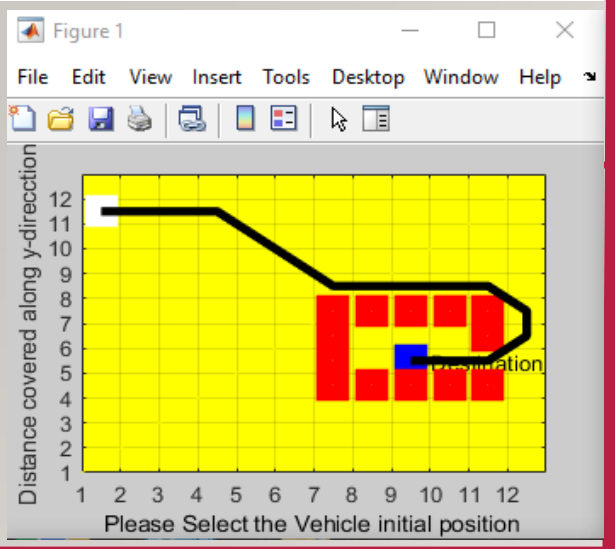
Enter the size of the grid:11



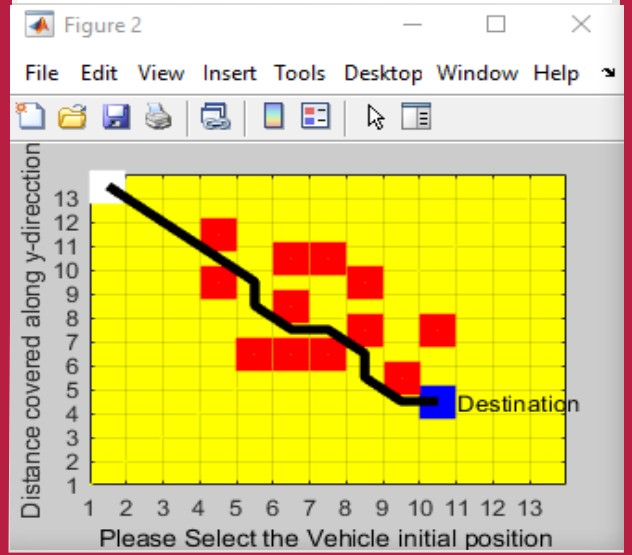
GENERALISED CODE FROM 4X4 GRID TO 20X20 GRID

: Source : Destination : Obstacle

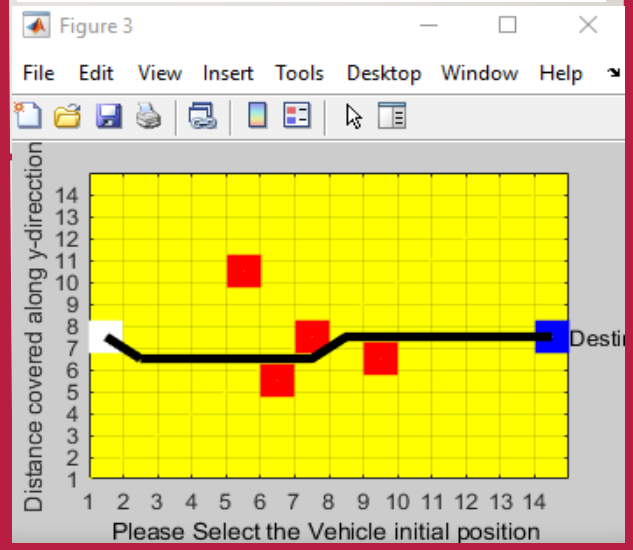
Enter the size of the grid:12



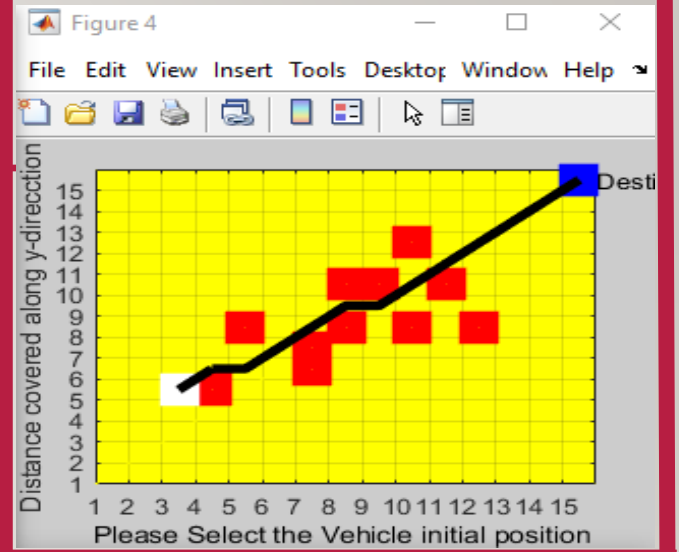
Enter the size of the grid:13



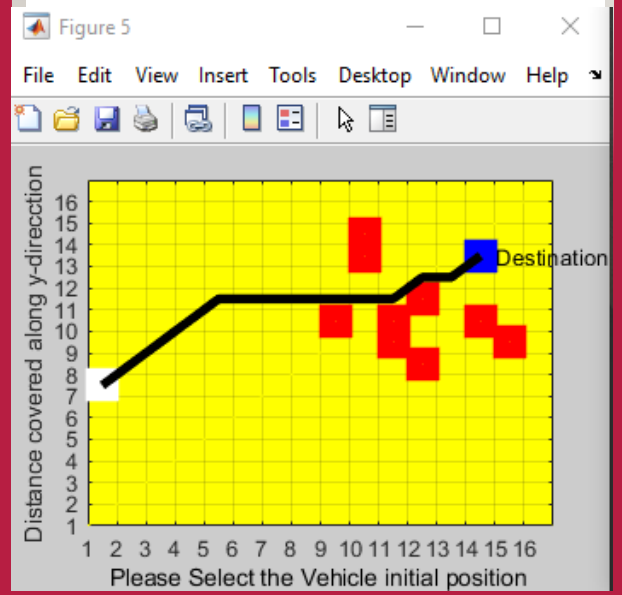
Enter the size of the grid:14



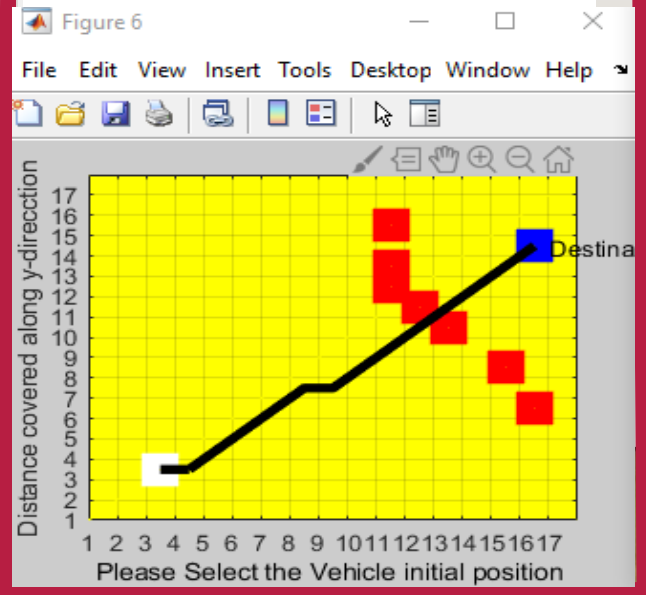
Enter the size of the grid:15



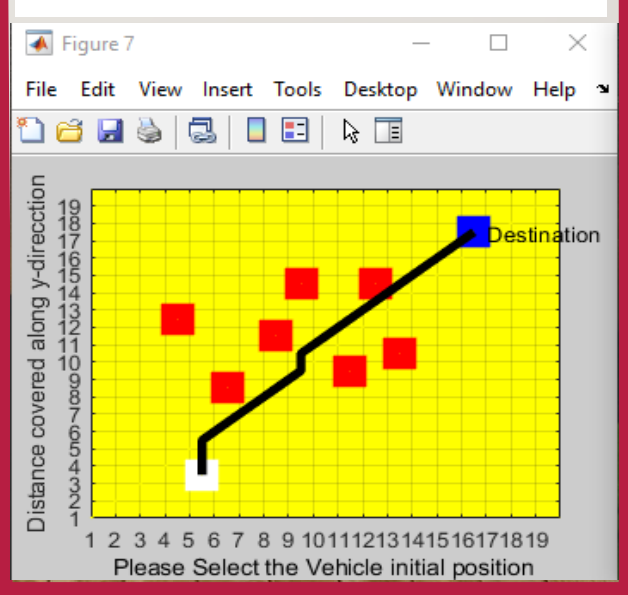
Enter the size of the grid:16



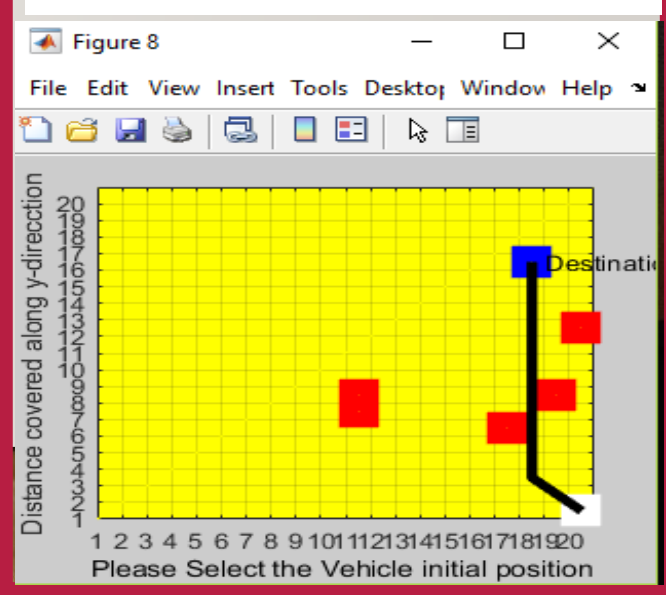
Enter the size of the grid:17



Enter the size of the grid:19



Enter the size of the grid:20

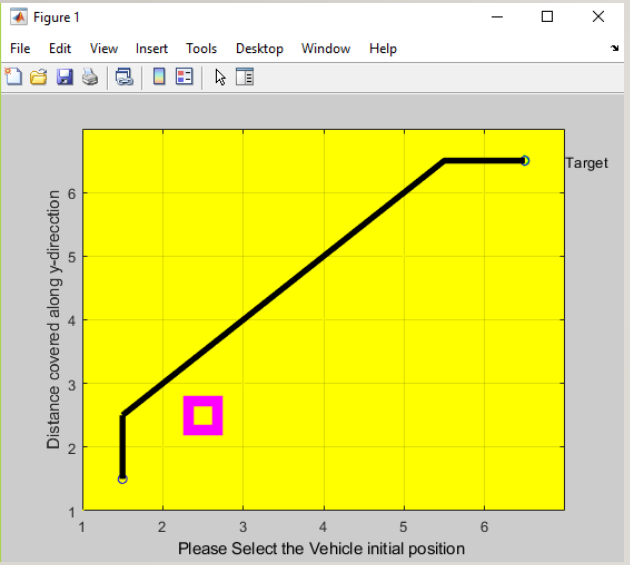


IMPLEMENTATION OF SARSA ALGORITHM FOR 6X6 GRID IN MATLAB

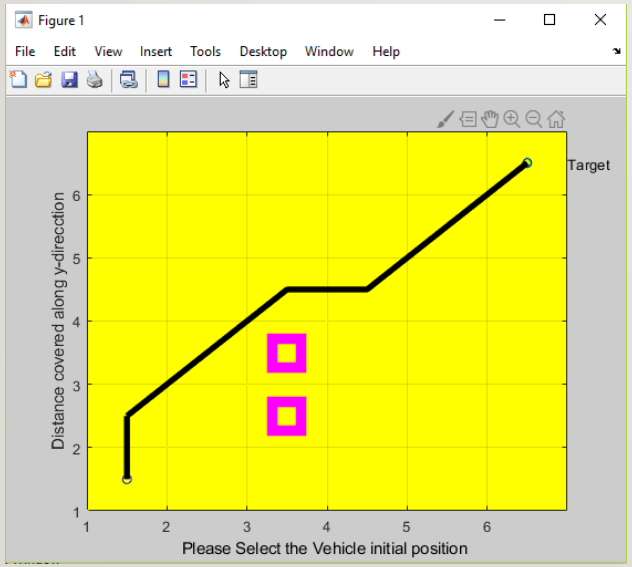
SIMULATION RESULTS WHERE SARSA ALGORITHM WORKS :

OPTIMIZATION CRITERIA: SHORTEST PATH BETWEEN SOURCE AND DESTINATION

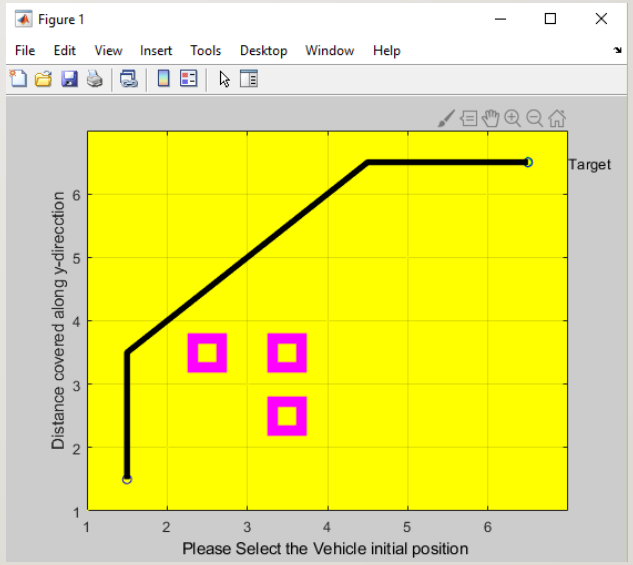
No of Obstacles: 1 Position of Obstacle:8



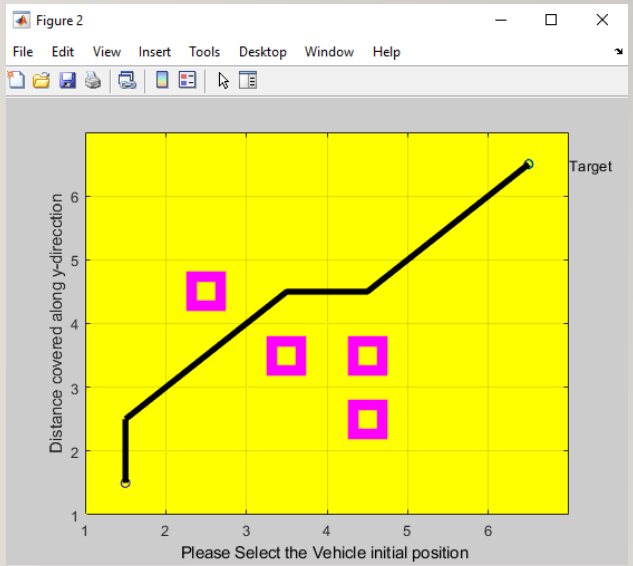
No of Obstacles: 2 Position of Obstacle:8,14



No of Obstacles: 3 Position of Obstacle:14,15,9



No of Obstacles: 4 Position of Obstacle:15,16,10,20

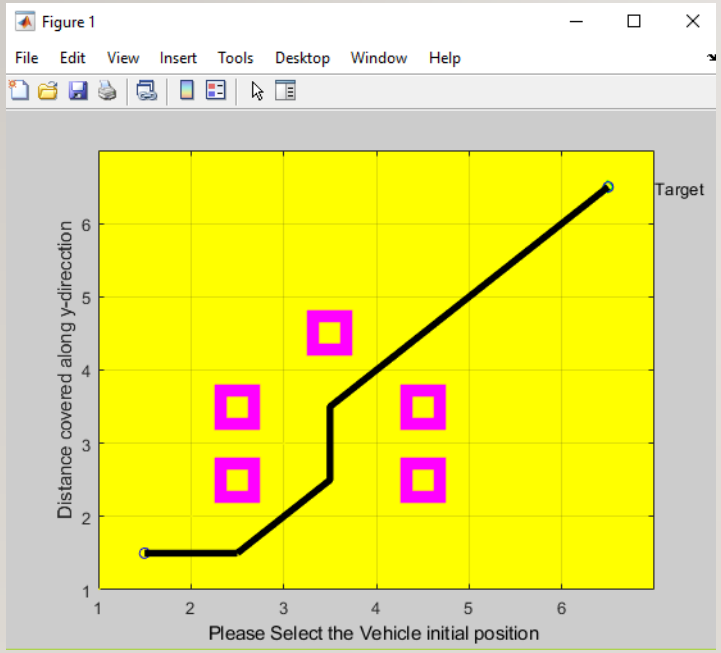


IMPLEMENTATION OF SARSA ALGORITHM FOR 6X6 GRID IN MATLAB

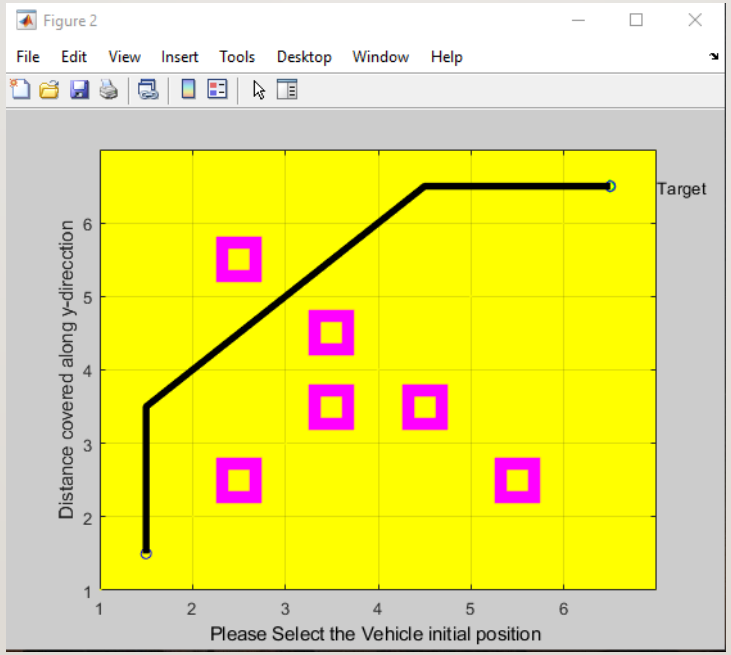
SIMULATION RESULTS WHERE SARSA ALGORITHM WORKS :

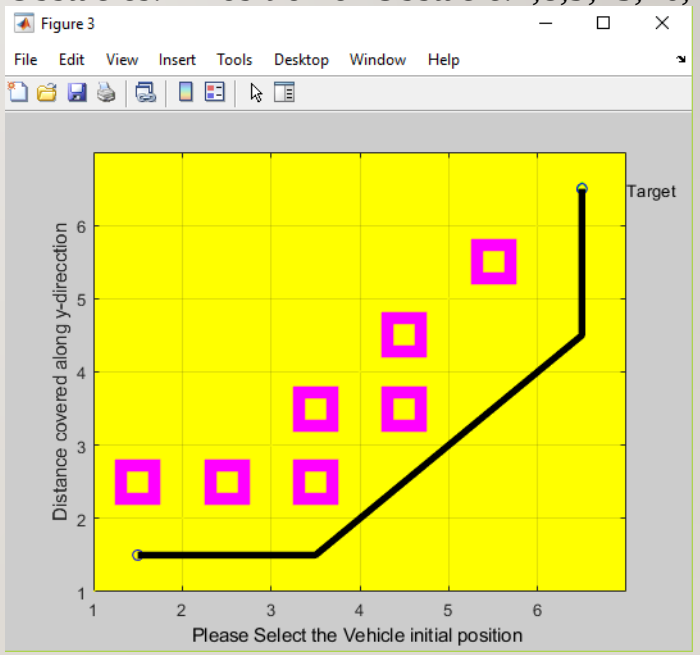
OPTIMIZATION CRITERIA: SHORTEST PATH BETWEEN SOURCE AND DESTINATIO

No of Obstacles: 5 Position of Obstacle:8,14,21,16,10



No of Obstacles: 7 Position of Obstacle:7,8,9,15,16,22,29

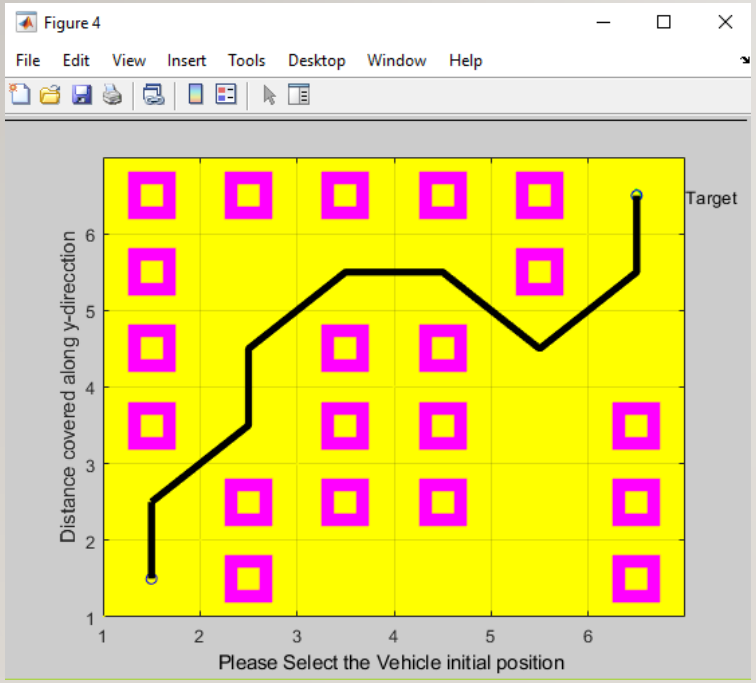




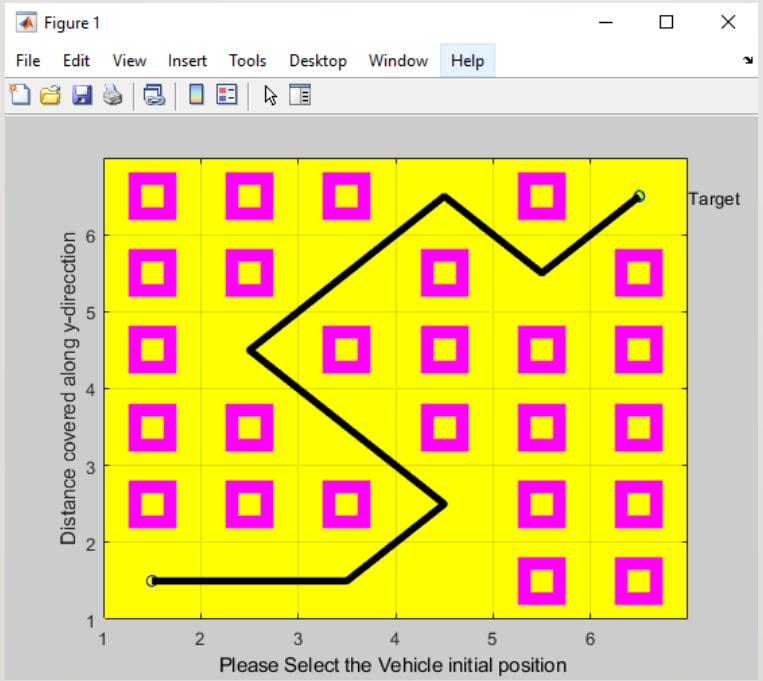
No of Obstacles: 6 Position of Obstacle:8,15,16,11,21,26

IMPLEMENTATION OF SARSA ALGORITHM TO SOLVE COMPLEX MAZES WITH MORE THAN 15 OBSTACLES

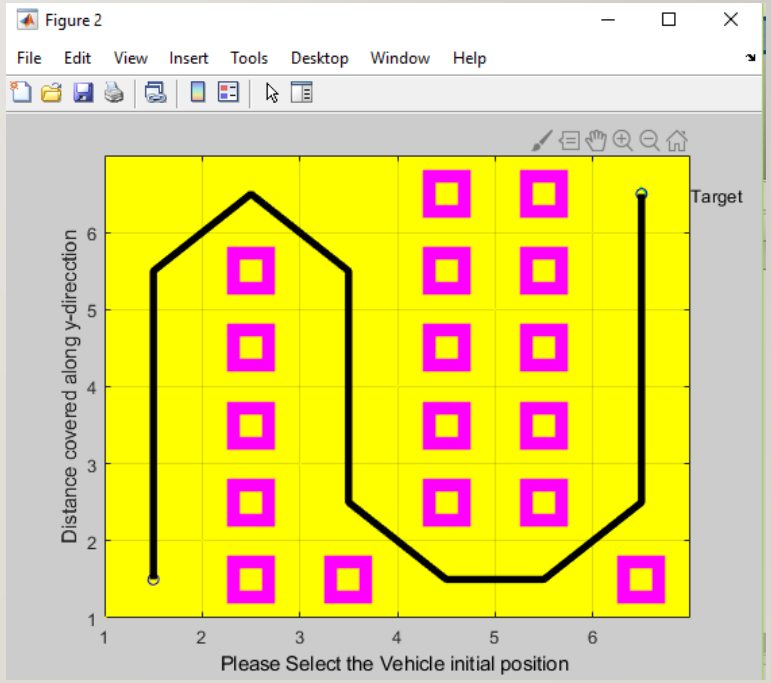
SIMULATION RESULTS WHERE SARSA ALGORITHM WORKS FOR COMPLEX MAZES :



No of Obstacles: 20



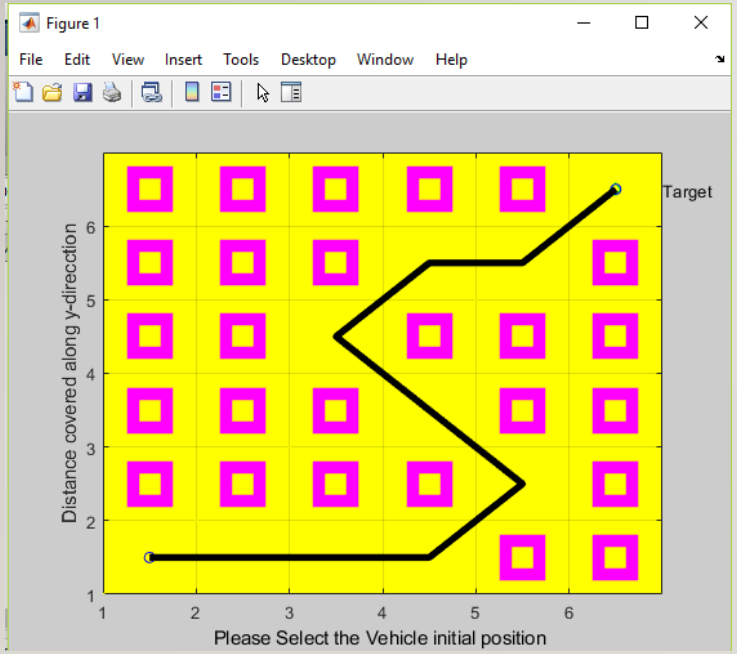
No of Obstacles: 25



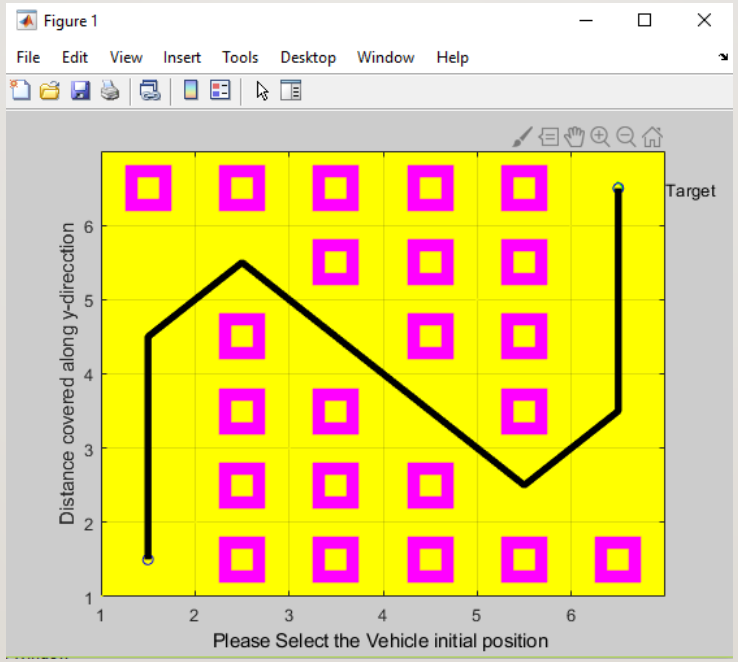
No of Obstacles: 17

IMPLEMENTATION OF SARSA ALGORITHM TO SOLVE COMPLEX MAZES WITH MORE THAN 15 OBSTACLES

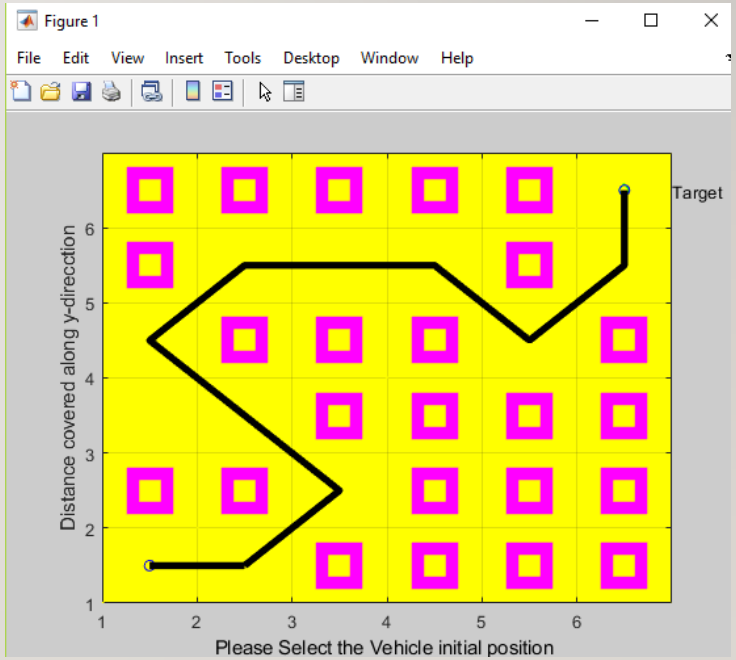
SIMULATION RESULTS WHERE SARSA ALGORITHM WORKS FOR COMPLEX MAZES :



No of Obstacles: 26



No of Obstacles: 22

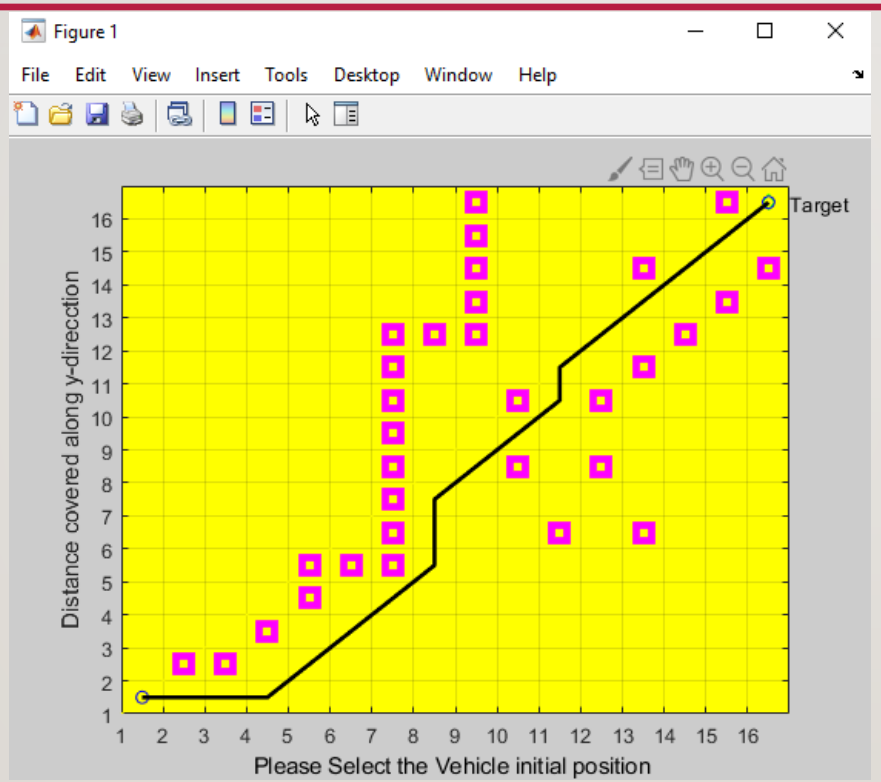


No of Obstacles: 24

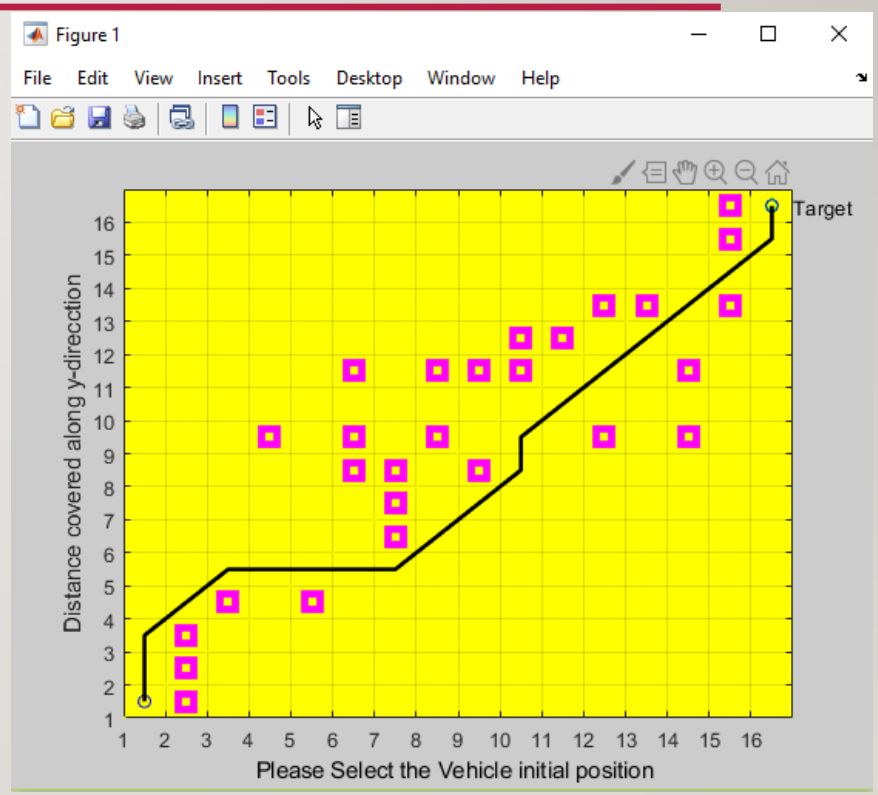
IMPLEMENTATION OF SARSA ALGORITHM FOR 16X16 GRID IN MATLAB

SIMULATION RESULTS WHERE SARSA ALGORITHM WORKS :

- 16x16 grid with 256 grids
- Up to 200 obstacles
- Works perfectly for almost all the cases



No of Obstacles: 31

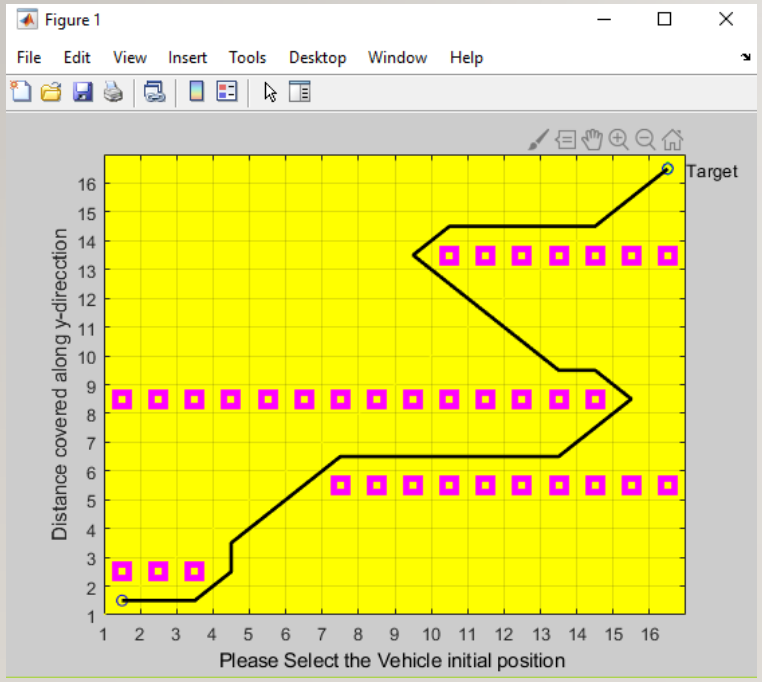


No of Obstacles: 27

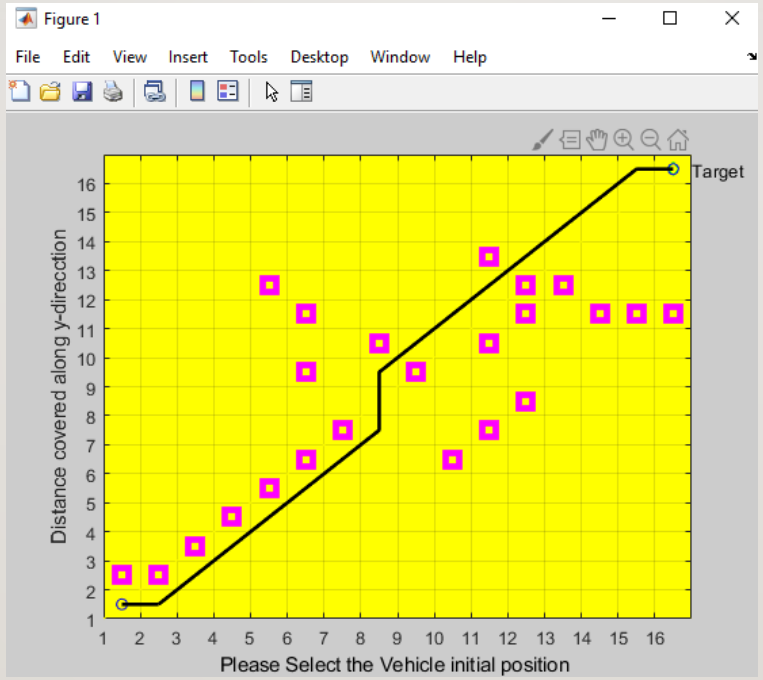
OPTIMIZATION CRITERIA: SHORTEST PATH BETWEEN SOURCE AND DESTINATION

IMPLEMENTATION OF SARSA ALGORITHM FOR 16X16 GRID IN MATLAB

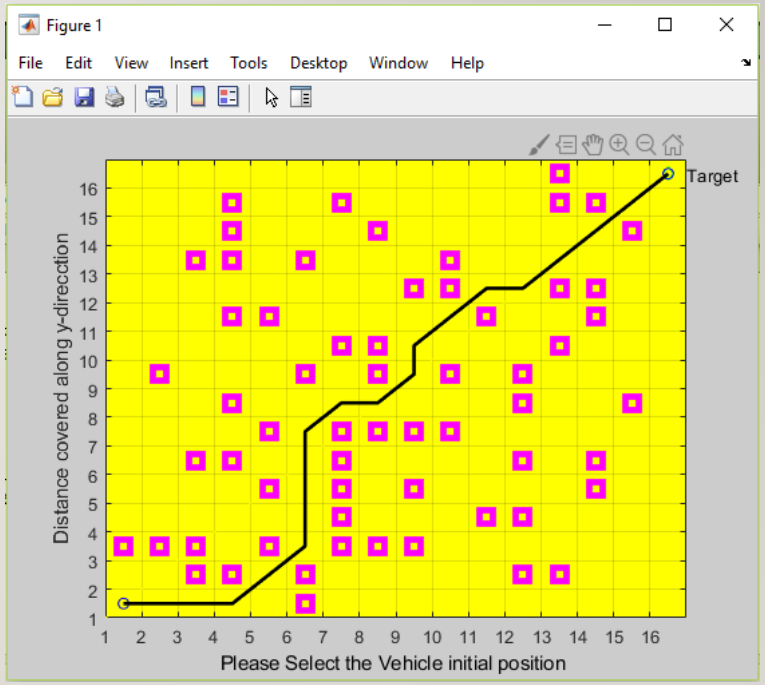
SIMULATION RESULTS WHERE SARSA ALGORITHM WORKS :



No of Obstacles: 34



No of Obstacles: 23

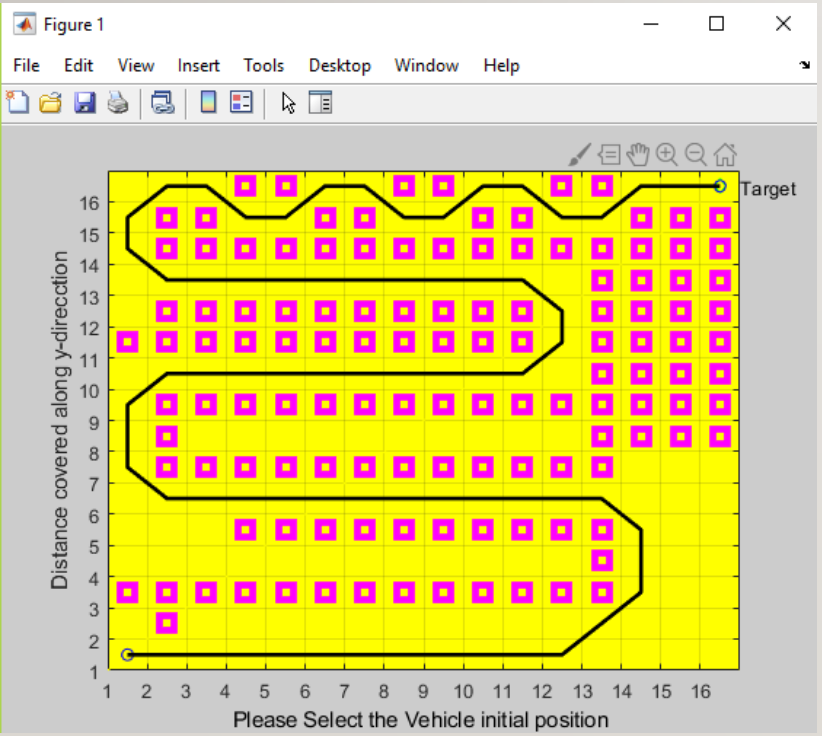


No of Obstacles: 60

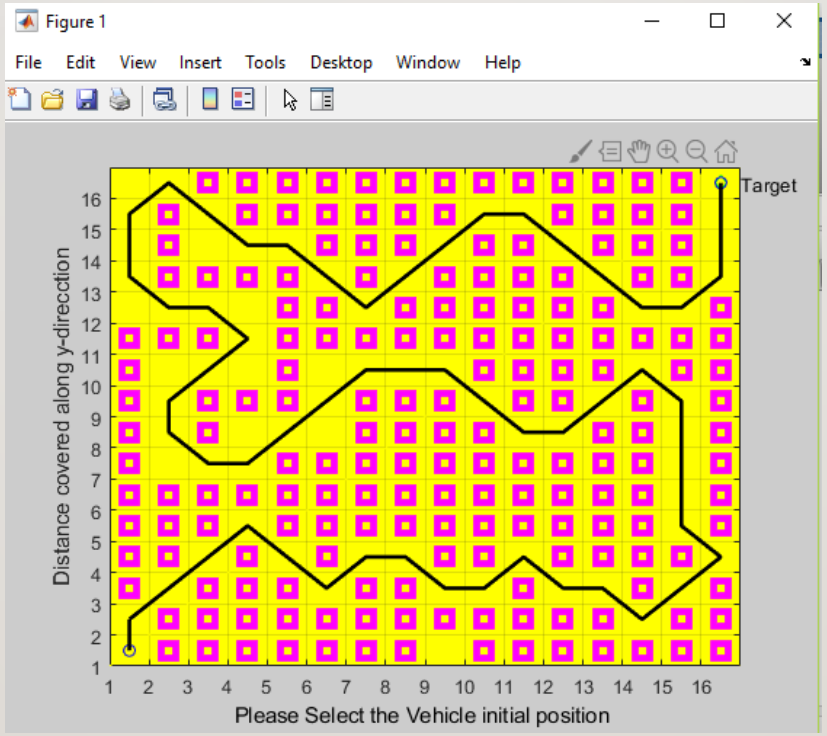
OPTIMIZATION CRITERIA: SHORTEST PATH BETWEEN SOURCE AND DESTINATION

IMPLEMENTATION OF SARSA ALGORITHM FOR 16X16 GRID IN MATLAB

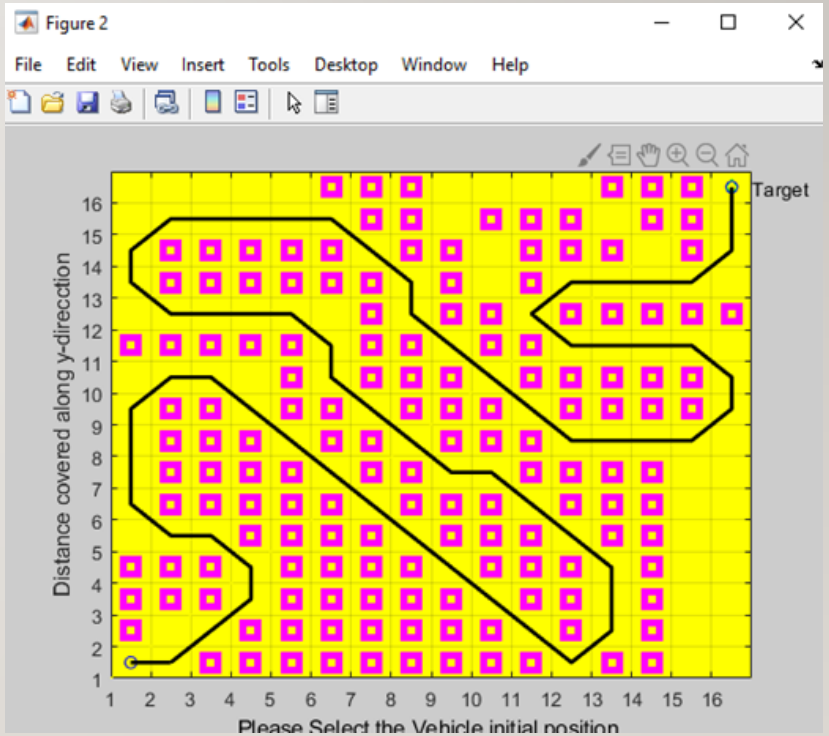
SIMULATION RESULTS WHERE SARSA ALGORITHM IS USED FOR SOLVING COMPLEX MAZES :



No of Obstacles: 123



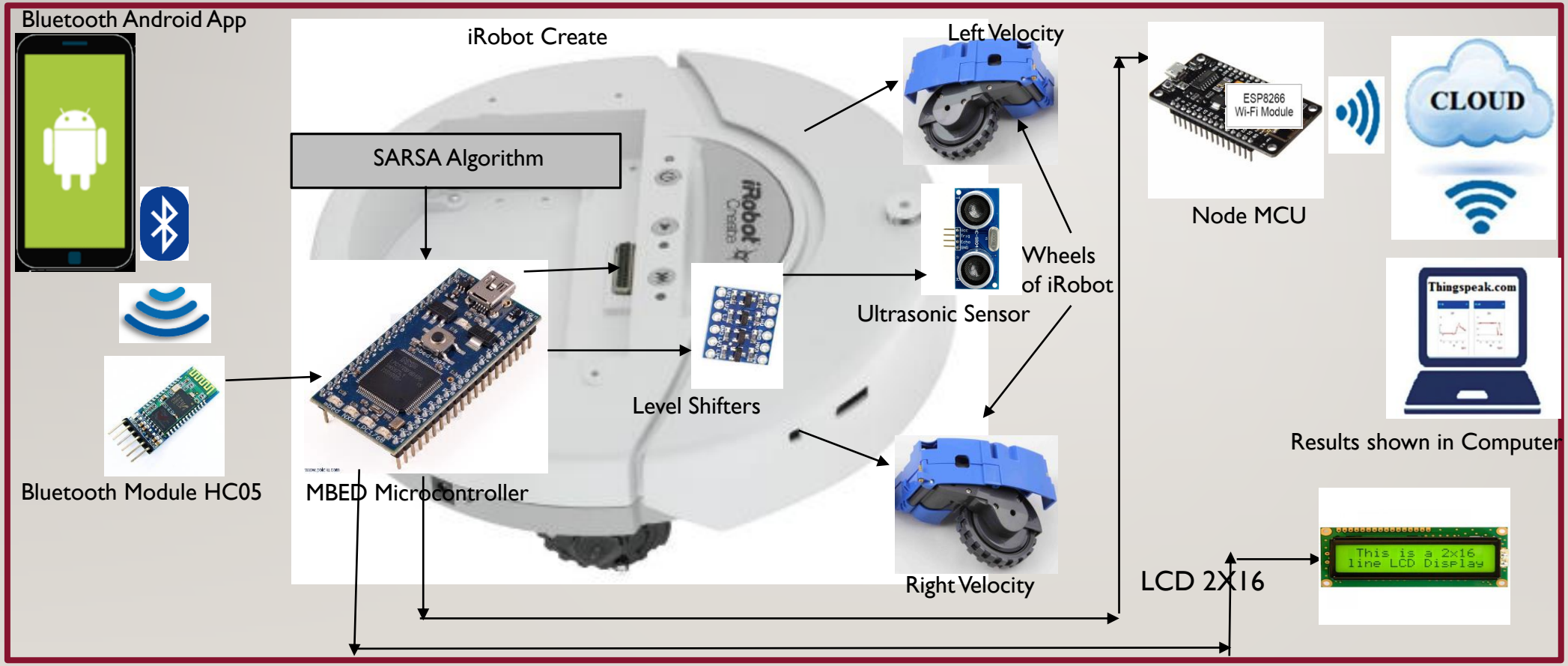
No of Obstacles: 185



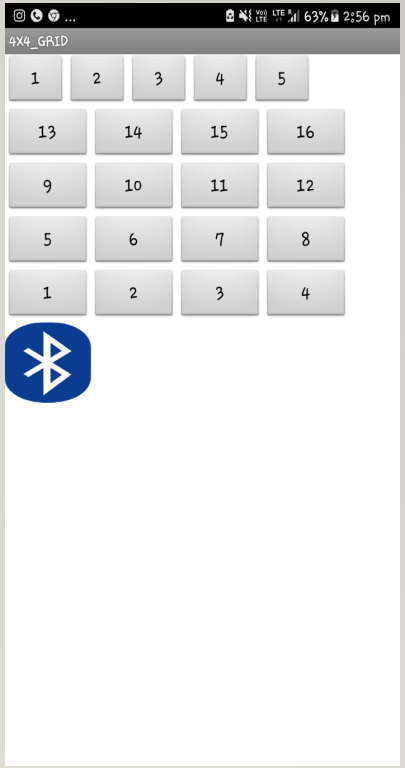
No of Obstacles: 148

BLOCK DIAGRAM OF SARSA ALGORITHM FOR A 4X4 GRID

a. System Layout



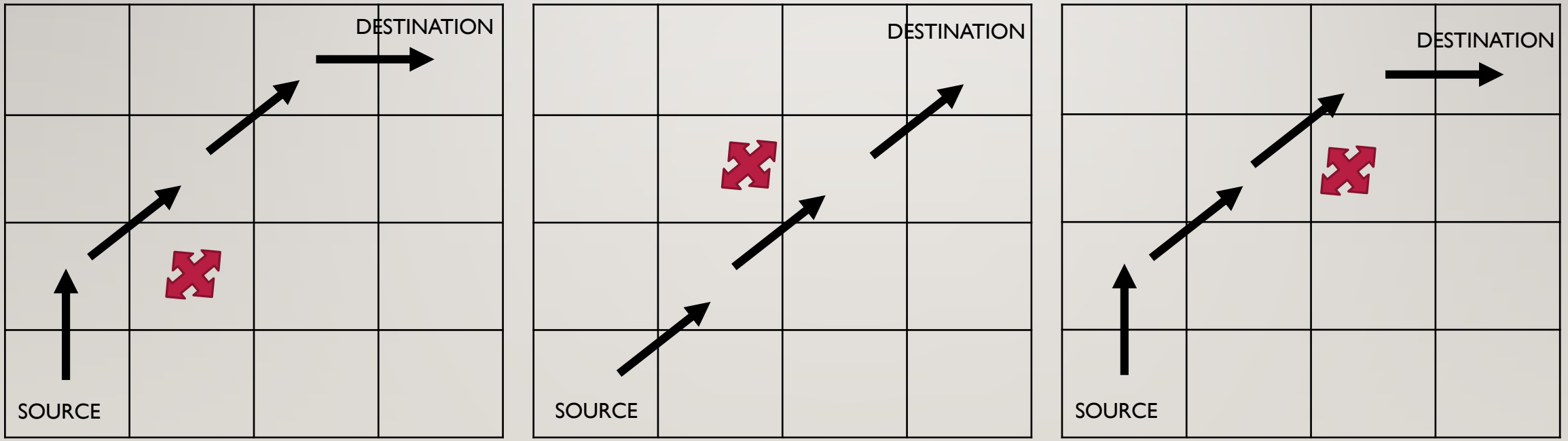
b. App Layout



IMPLEMENTATION OF SARSA ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

Cases tried for SARSA algorithm in Hardware for 1-5 obstacles

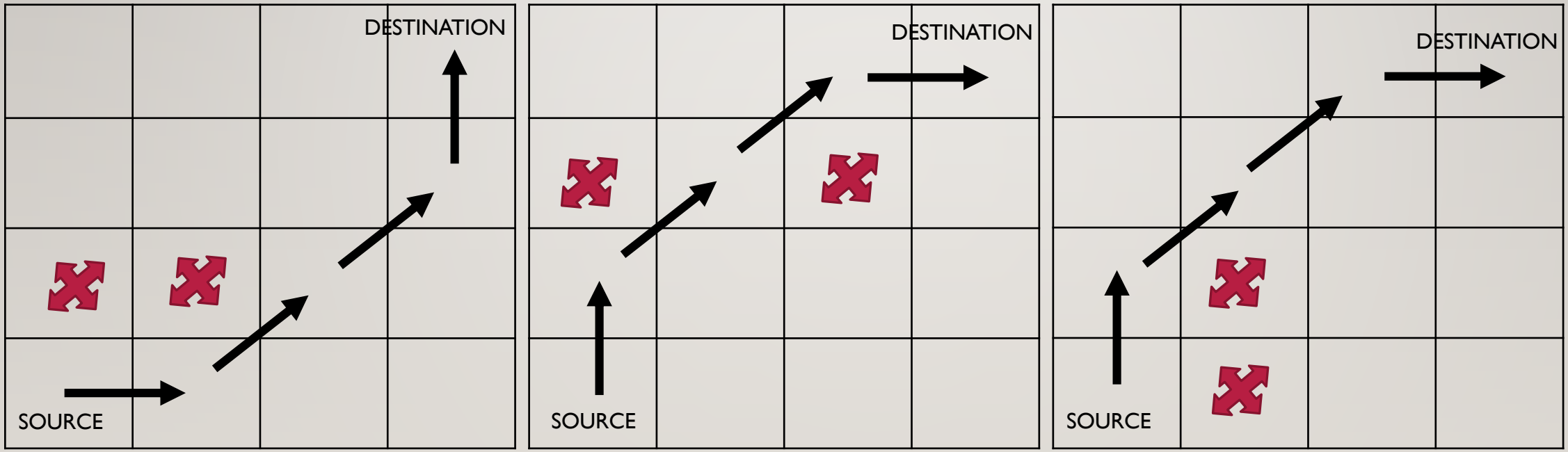
Path Planning with one obstacle



IMPLEMENTATION OF SARSA ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

Cases tried for SARSA algorithm in Hardware for 1-5 obstacles

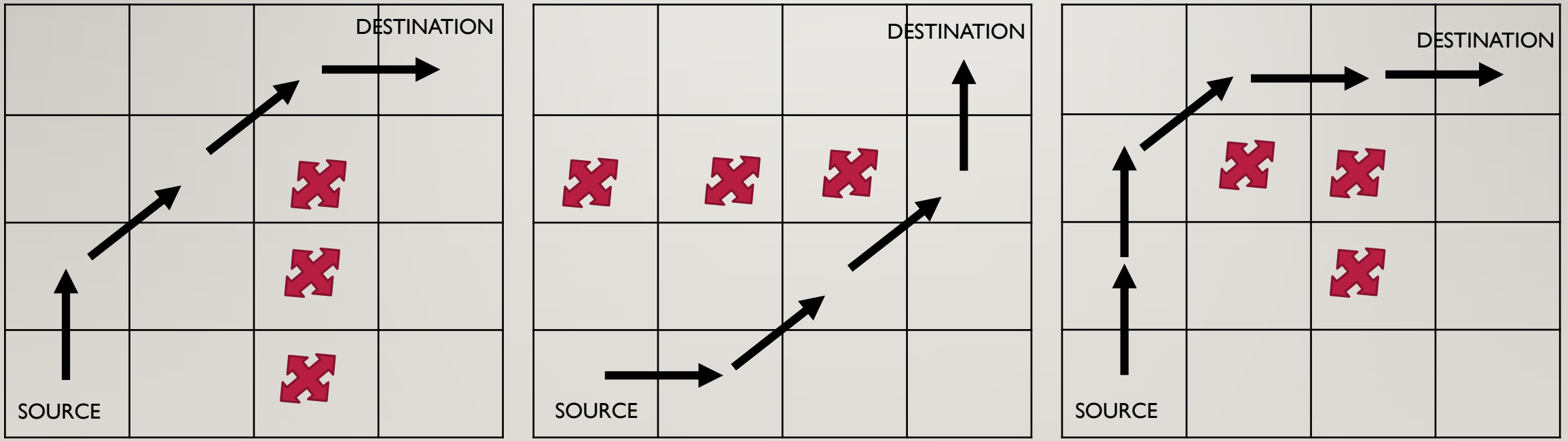
Path Planning with two obstacle



IMPLEMENTATION OF SARSA ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

Cases tried for SARSA algorithm in Hardware for 1-5 obstacles

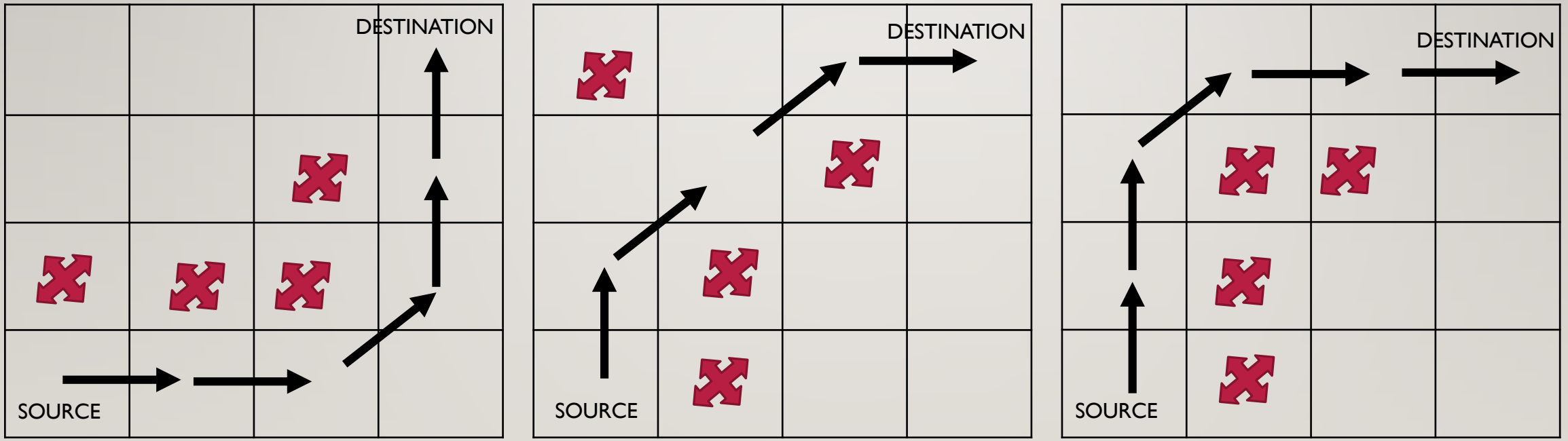
Path Planning with three obstacle



IMPLEMENTATION OF SARSA ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

Cases tried for SARSA algorithm in Hardware for 1-5 obstacles

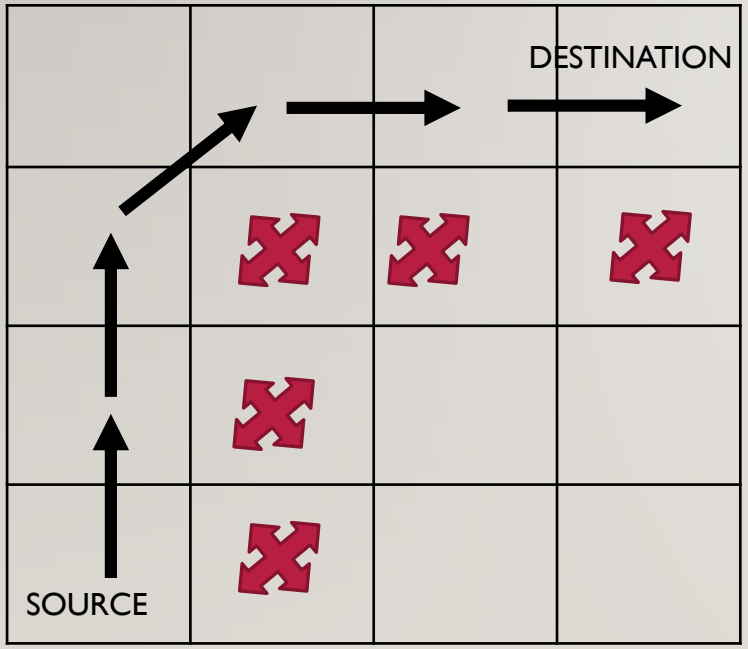
Path Planning with four obstacle

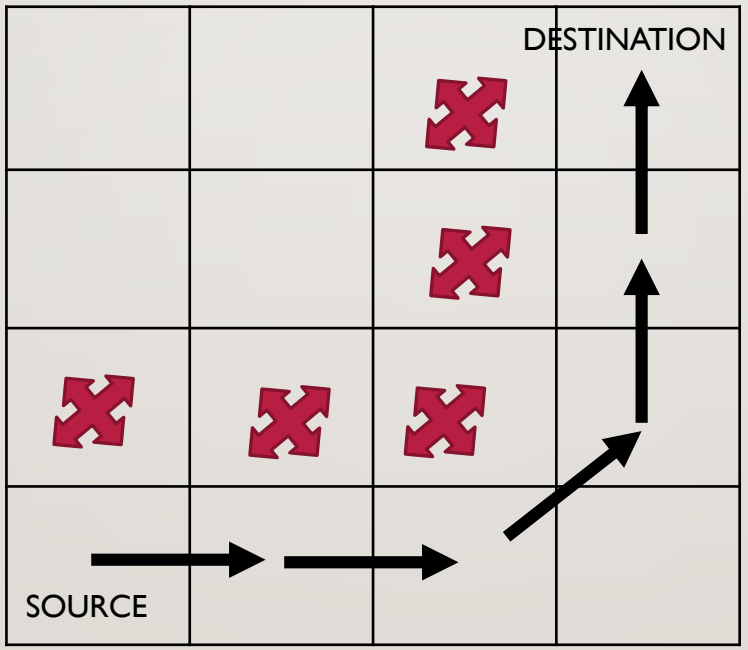


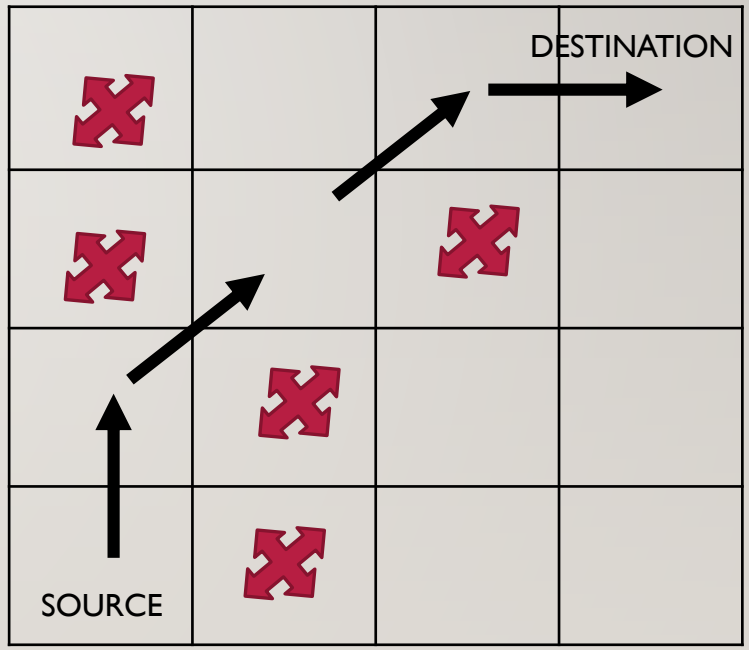
IMPLEMENTATION OF SARSA ALGORITHM IN MBED MICROCONTROLLER FOR A 4X4 GRID

Cases tried for SARSA algorithm in Hardware for 1-5 obstacles

Path Planning with five obstacle







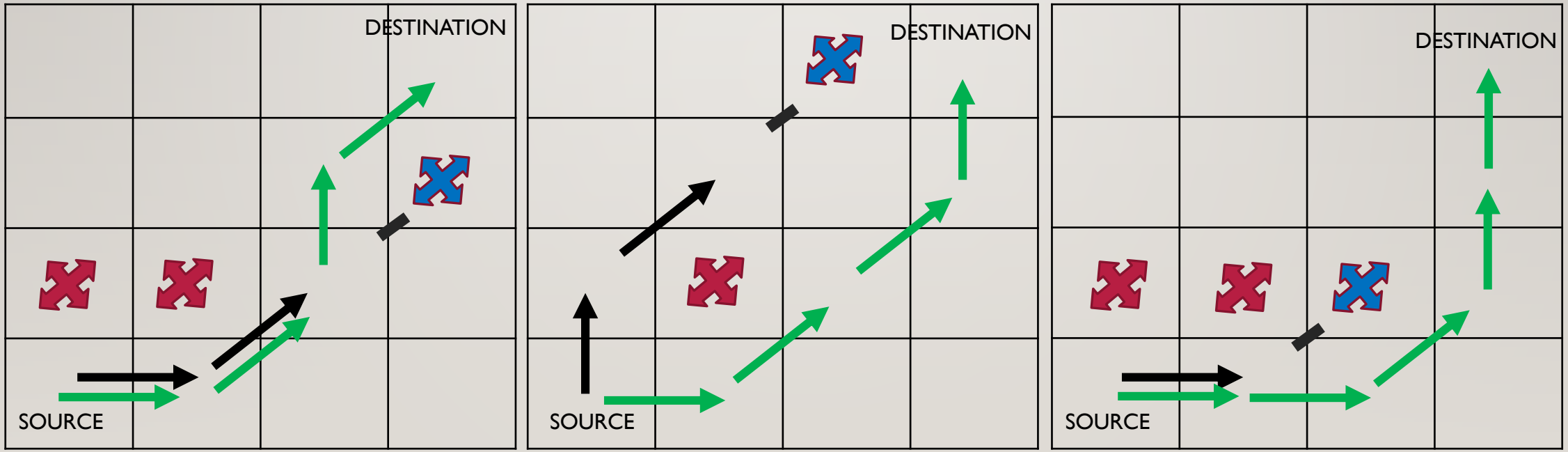
IMPLEMENTATION OF SARSA ALGORITHM FOR A 4X4 GRID WITH DYNAMIC OBSTCALES

Cases tried for TD algorithm with dynamic obstacles in Hardware

Path Planning with one or two static obstacle and one dynamic obstacle

ULTRASONIC SENSOR

: Path taken before the presence of dynamic obstacle, : Path taken before the presence of dynamic obstacle, : Static Obstacles, : Dynamic Obstacles

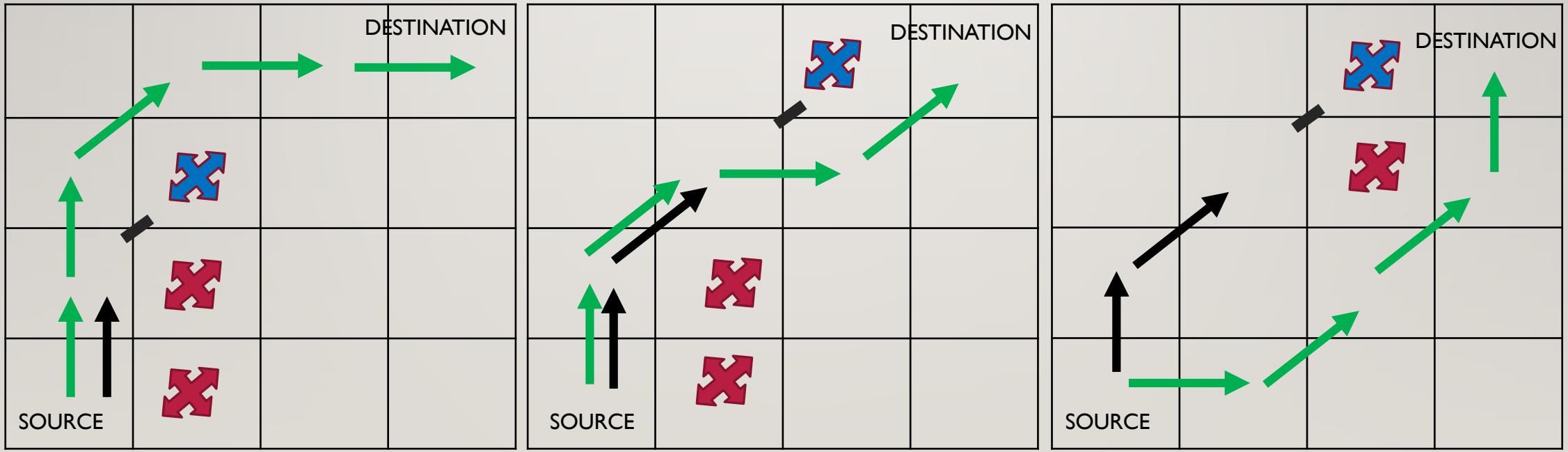


IMPLEMENTATION OF SARSA ALGORITHM FOR A 4X4 GRID WITH DYNAMIC OBSTCALES

Cases tried for TD algorithm with dynamic obstacles in Hardware

Path Planning with one or two static obstacle and one dynamic obstacle

: Path taken before the presence of dynamic obstacle, : Path taken before the presence of dynamic obstacle, : Static Obstacles, : Dynamic Obstacles

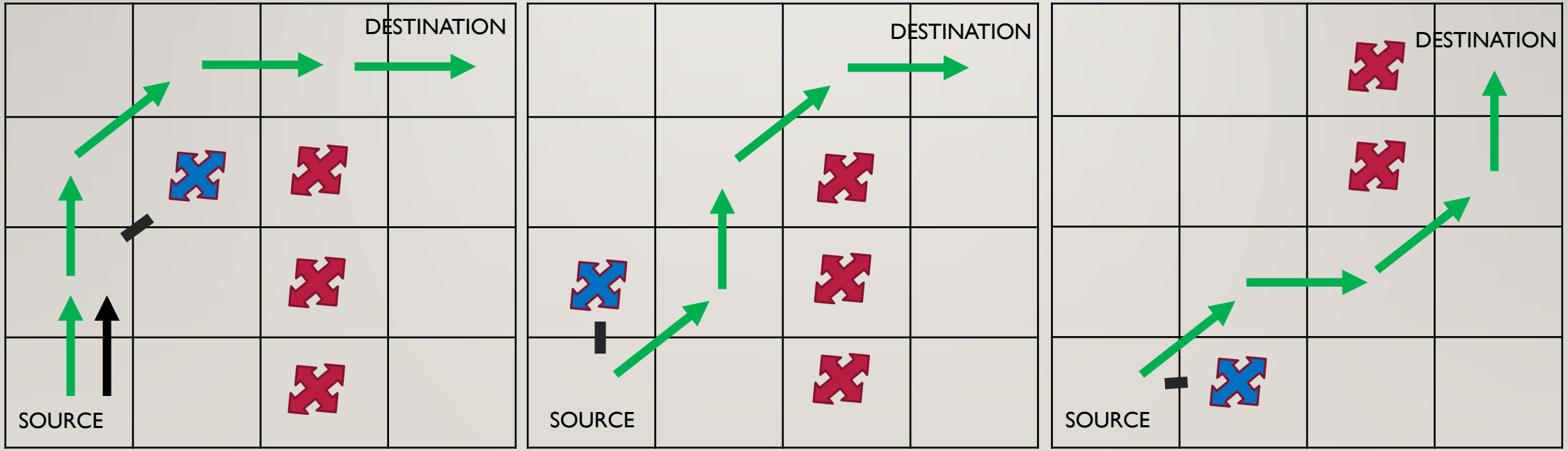


IMPLEMENTATION OF SARSA ALGORITHM FOR A 4X4 GRID WITH DYNAMIC OBSTCALES

Cases tried for TD algorithm with dynamic obstacles in Hardware

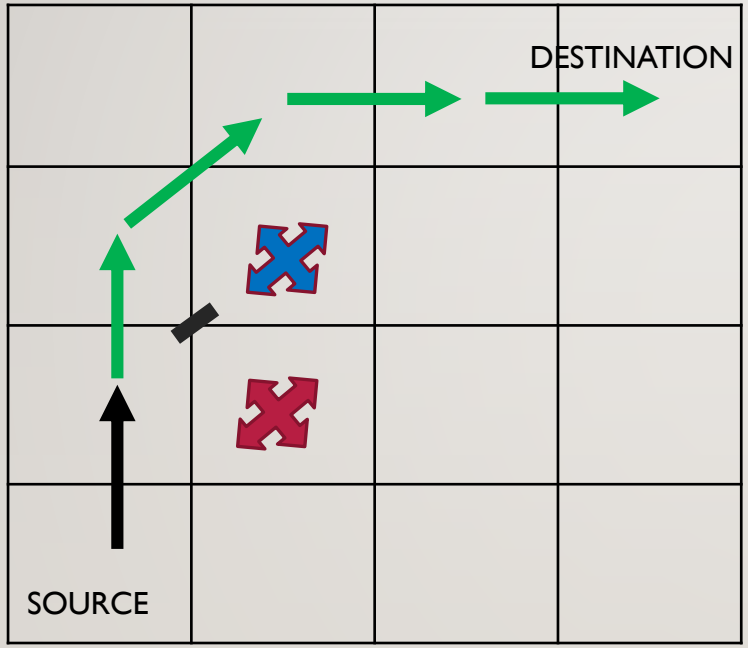
Path Planning with three or two static obstacle and one dynamic obstacle

: Path taken before the presence of dynamic obstacle, : Path taken before the presence of dynamic obstacle, : Static Obstacles, : Dynamic Obstacles



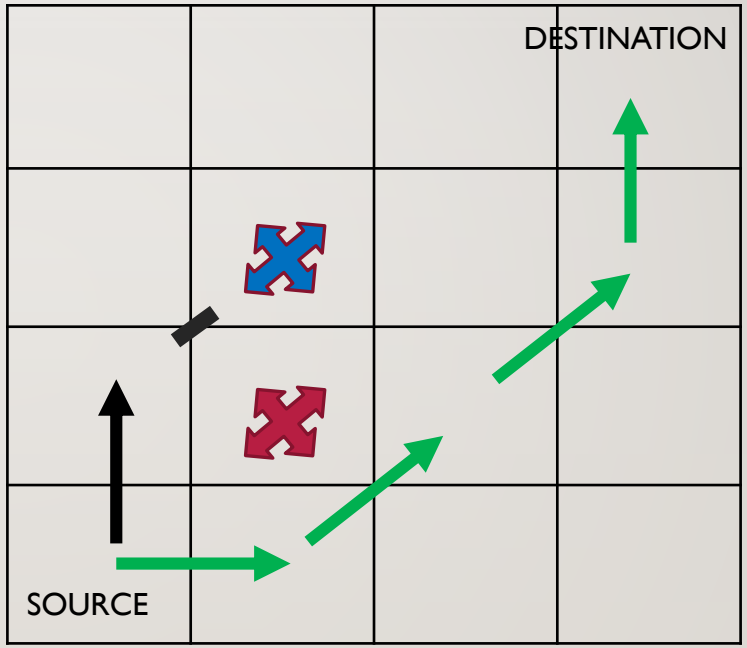
COMPARISON BETWEEN SARSA AND TD WITH DYNAMIC OBSTACLES BASED ON THE DISTANCE TRAVELLED FOR A 4X4 GRID IN HARDWARE

Case where SARSA takes the optimal path and TD doesn't



TD: No of Steps: 5

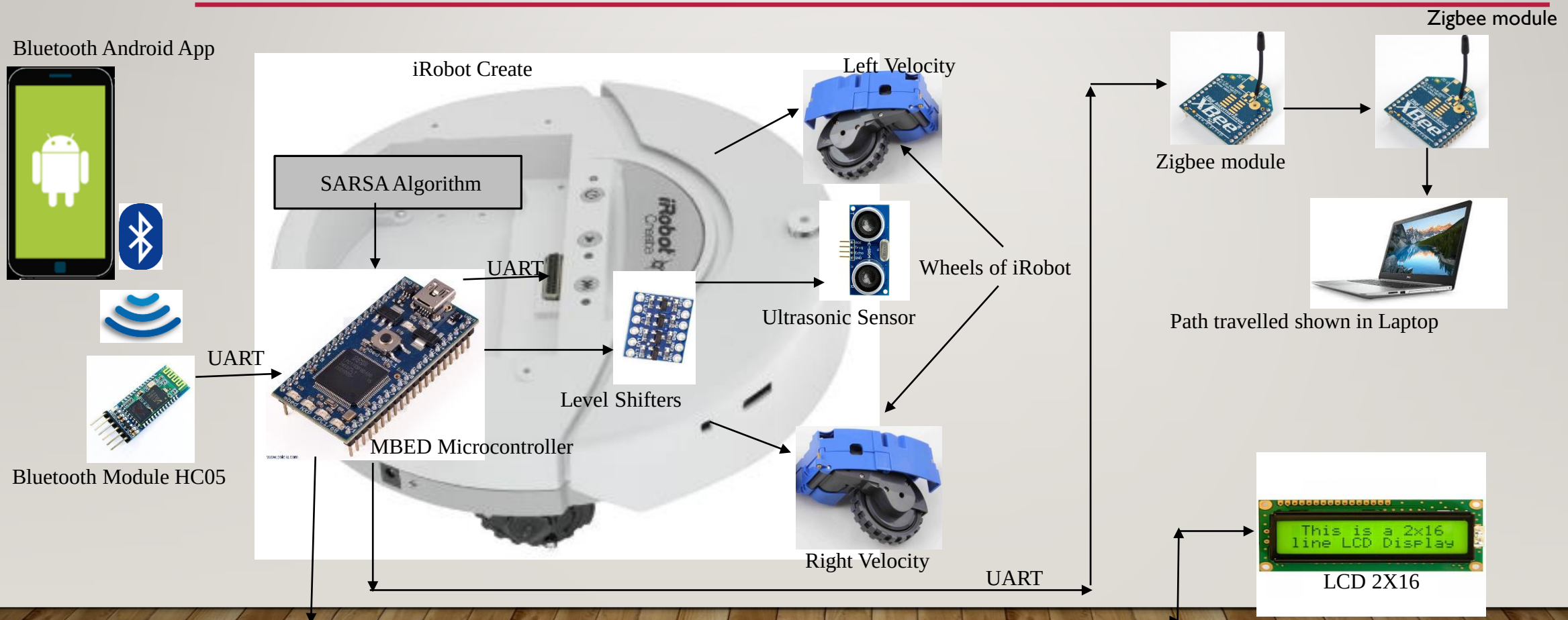
Distance Travelled: 216.559cm



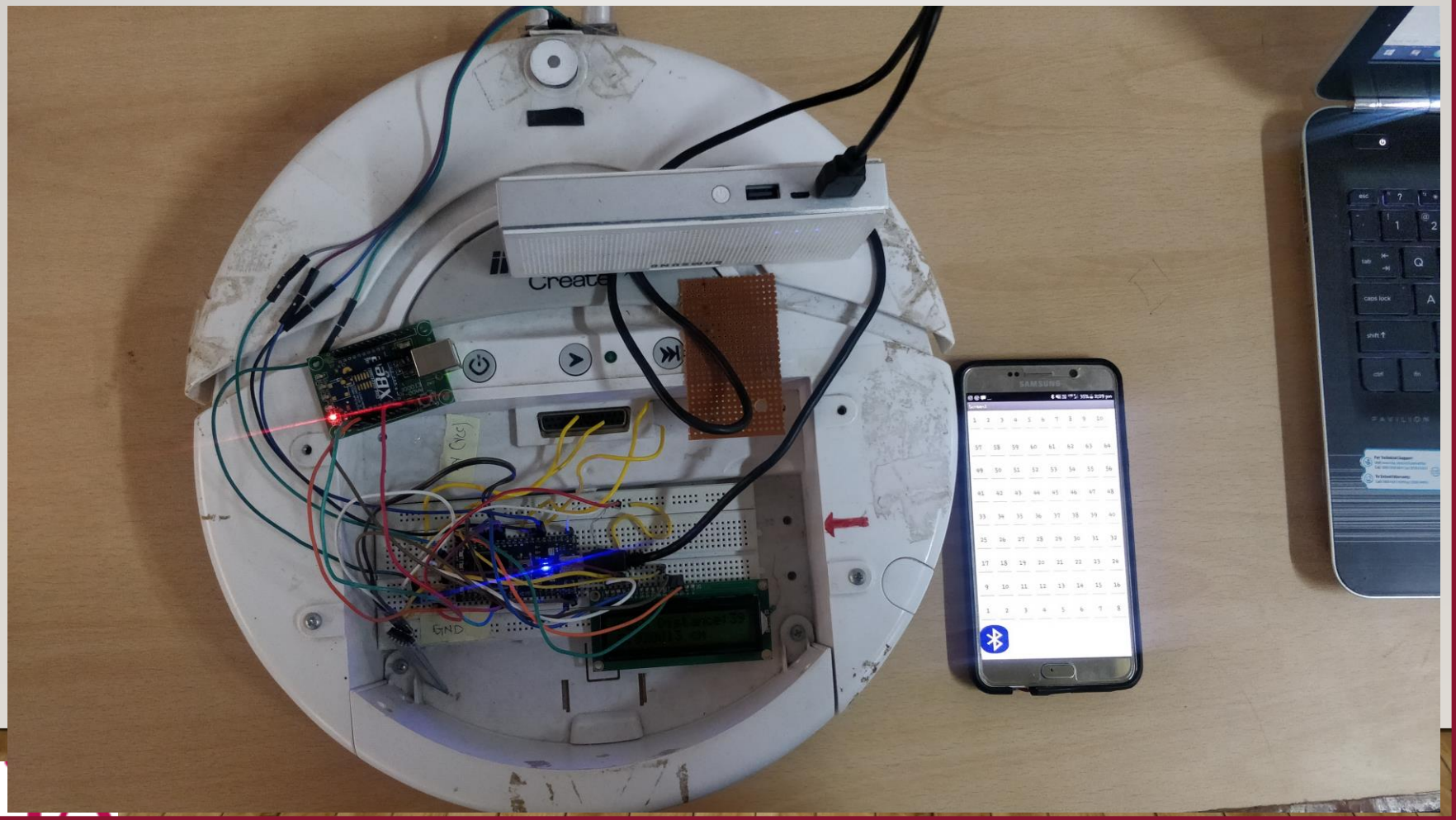
SARSA: No of Steps: 4

Distance Travelled: 193.119cm

BLOCK DIAGRAM OF SARSA ALGORITHM FOR AN 8X8 GRID IN HARDWARE



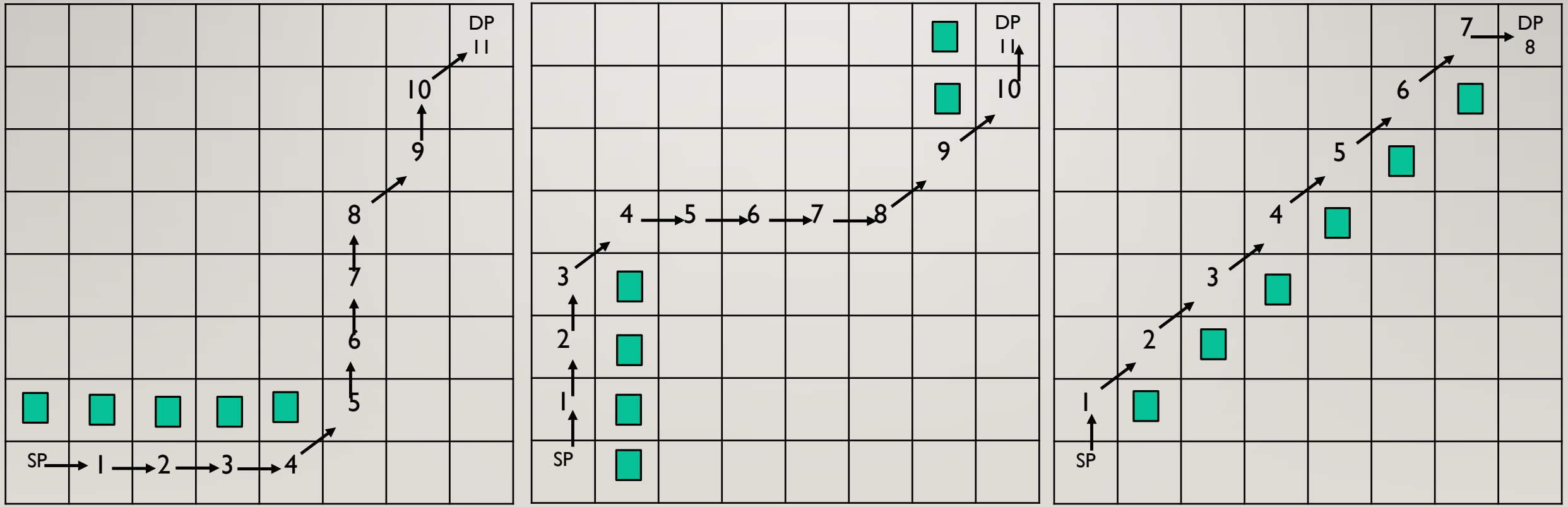
HARDWARE IMAGES FOR SARSA ALGORITHM FOR AN 8X8 GRID IN HARDWARE



IMPLEMENTATION OF SARSA ALGORITHM FOR 8X8 GRID IN HARDWARE

a. Cases tested for SARSA algorithm with 5 and 6 obstacles

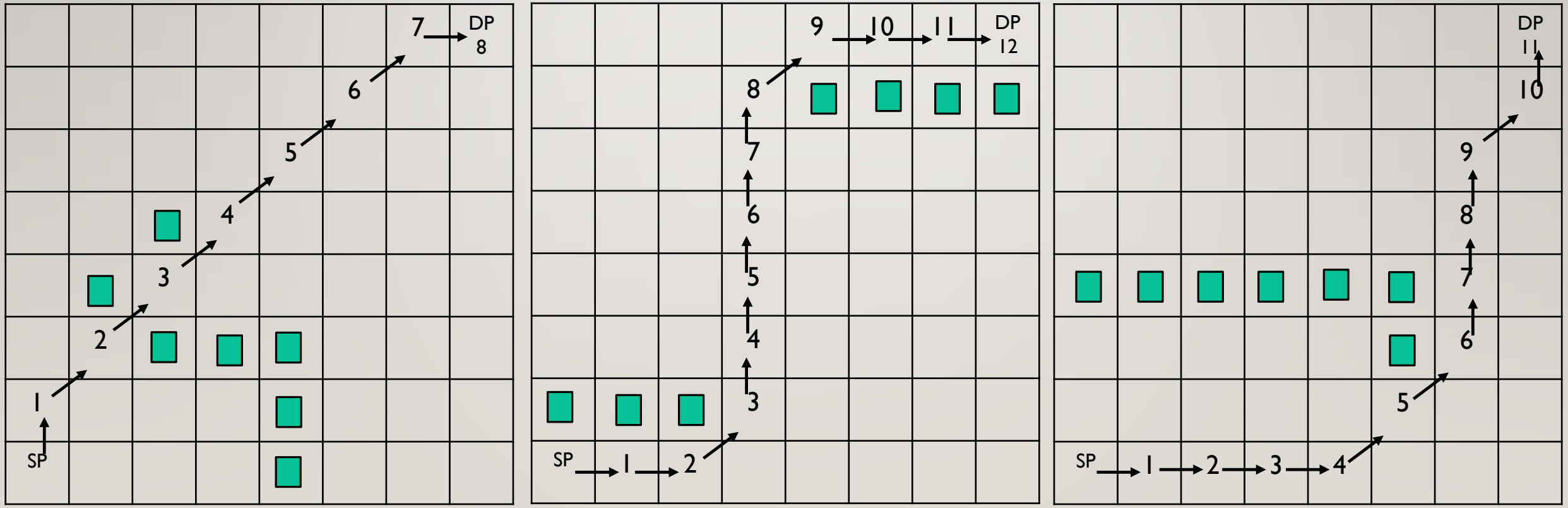
SP: Source Point, DP: Destination Point, ■: Static Obstacle, →: Indicates the path taken



IMPLEMENTATION OF SARSA ALGORITHM FOR 8X8 GRID IN HARDWARE

b. Cases tested for SARSA algorithm with 7 obstacles

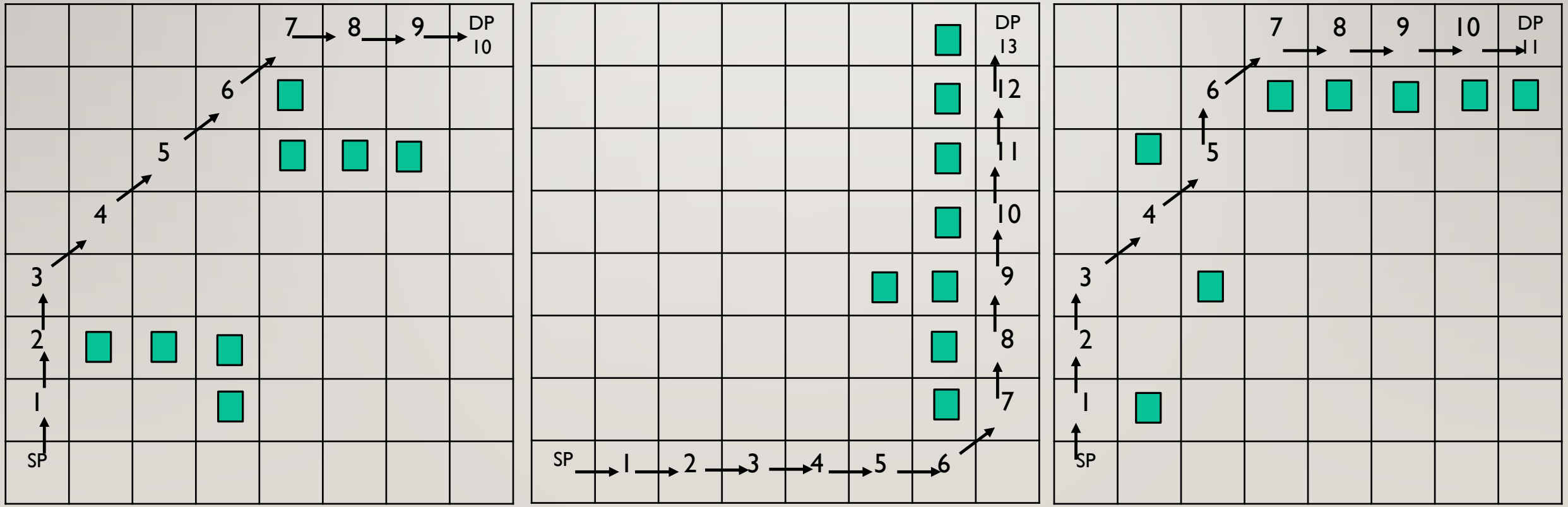
SP: Source Point, DP: Destination Point, ■: Static Obstacle, →: Indicates the path taken



IMPLEMENTATION OF SARSA ALGORITHM FOR 8X8 GRID IN HARDWARE

c. Cases tested for SARSA algorithm with 8 obstacles

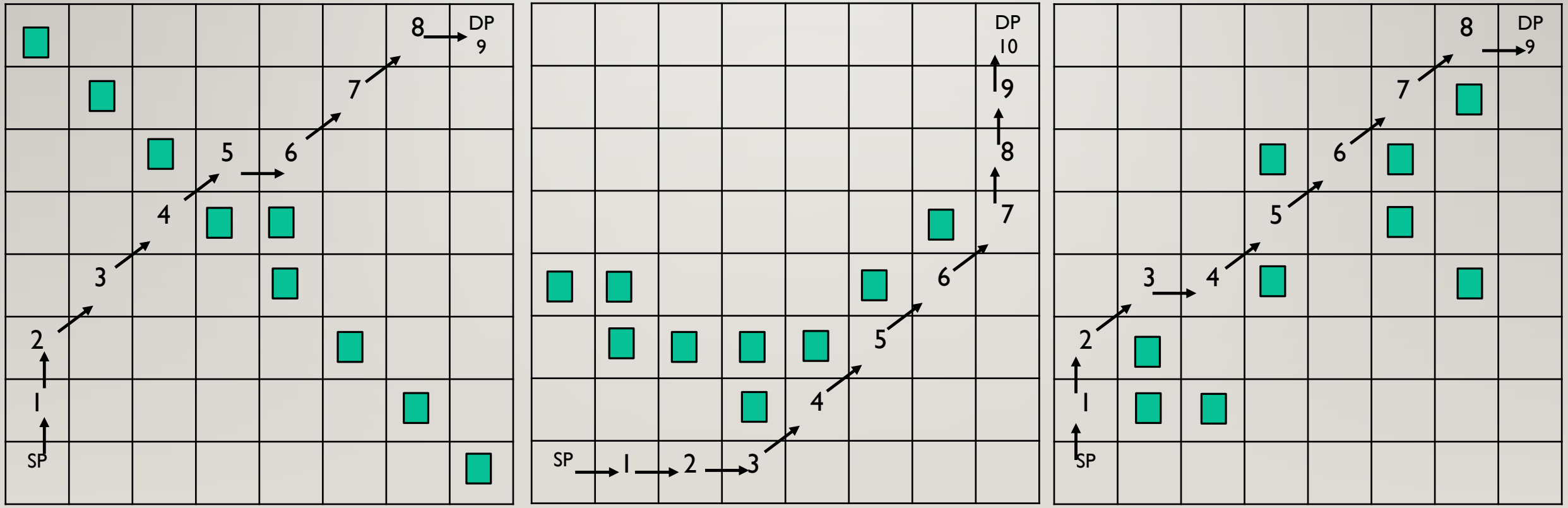
SP: Source Point, DP: Destination Point, ■: Static Obstacle, ⟶: Indicates the path taken



IMPLEMENTATION OF SARSA ALGORITHM FOR 8X8 GRID IN HARDWARE

d. Cases tested for SARSA algorithm with 9 obstacles

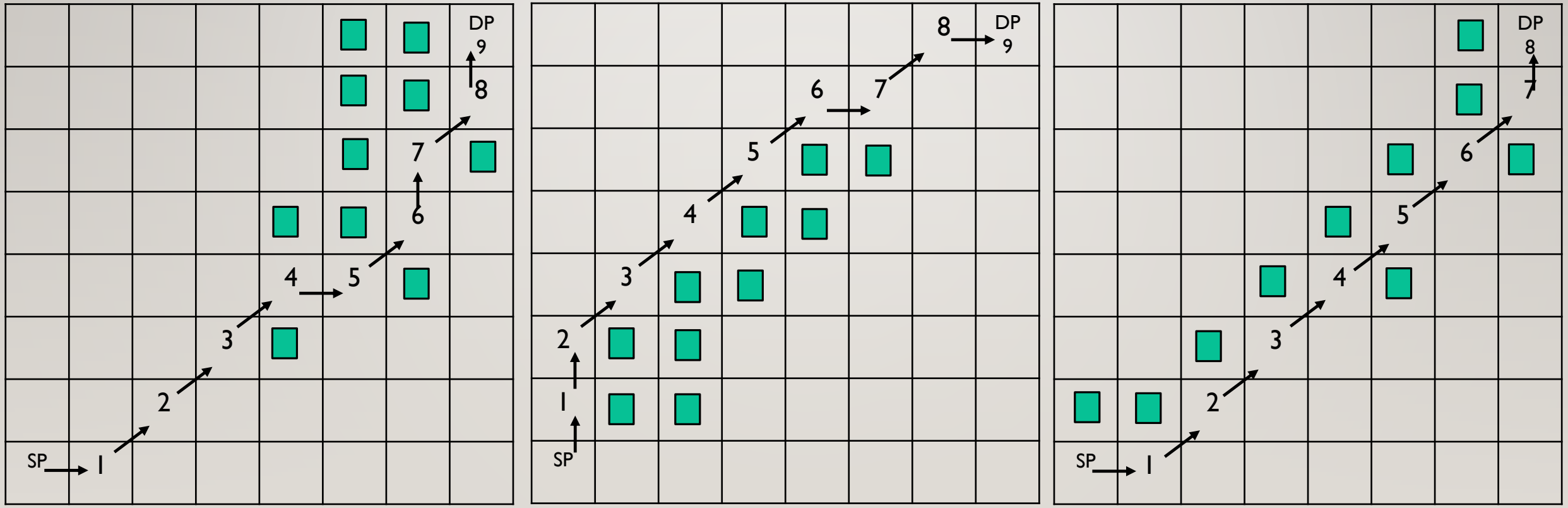
SP: Source Point, DP: Destination Point, ■: Static Obstacle, ⟶: Indicates the path taken



IMPLEMENTATION OF SARSA ALGORITHM FOR 8X8 GRID IN HARDWARE

e. Cases tested for SARSA algorithm with 10 obstacles

SP: Source Point, DP: Destination Point, ■: Static Obstacle, →: Indicates the path taken



IMPLEMENTATION OF SARSA ALGORITHM FOR AN 8X8 GRID IN HARDWARE

f. Analysis and Inference

SL NO	No of Obstacles	Position of Obstacles	Time Taken (in sec)
1	0	-	38.95
2	1	(1.5,1.5)	45.31
3	2	(1.5,1.5),(6.5,6.5)	45.61
4	3	(1.5,1.5),(1.5,2.5),(1.5,3.5)	50.58
5	4	(4.5,2.5),(4.5,3.5),(4.5,4.5),(4.5,5.5)	50.98
6	5	(1.5,0.5),(1.5,1.5),(1.5,2.5),(1.5,3.5),(1.5,4.5)	55.09
7	6	(1.5,0.5),(1.5,1.5),(1.5,2.5),(1.5,3.5),(6.5,7.5),(6.5,6.5)	55.06
8	7	(0.5,1.5),(1.5,1.5),(2.5,1.5),(7.5,6.5),(6.5,6.5),(5.5,6.5),(4.5,6.5)	57.66
9	8	(1.5,2.5),(2.5,2.5),(3.5,2.5),(3.5,1.5),(6.5,5.5),(5.5,5.5),(4.5,5.5),(4.5,6.5)	58.22
10	9	(0.5,3.5),(1.5,3.5),(1.5,2.5),(2.5,2.5),(3.5,2.5),(3.5,1.5),(4.5,2.5),(5.5,3.5),(6.5,4.5)	56.39
11	10	(1.5,0.5),(1.5,1.5),(1.5,2.5),(1.5,3.5),(1.5,4.5),(1.5,5.5),(1.5,6.5),(2.5,6.5),(3.5,6.5),(4.5,6.5)	62.57

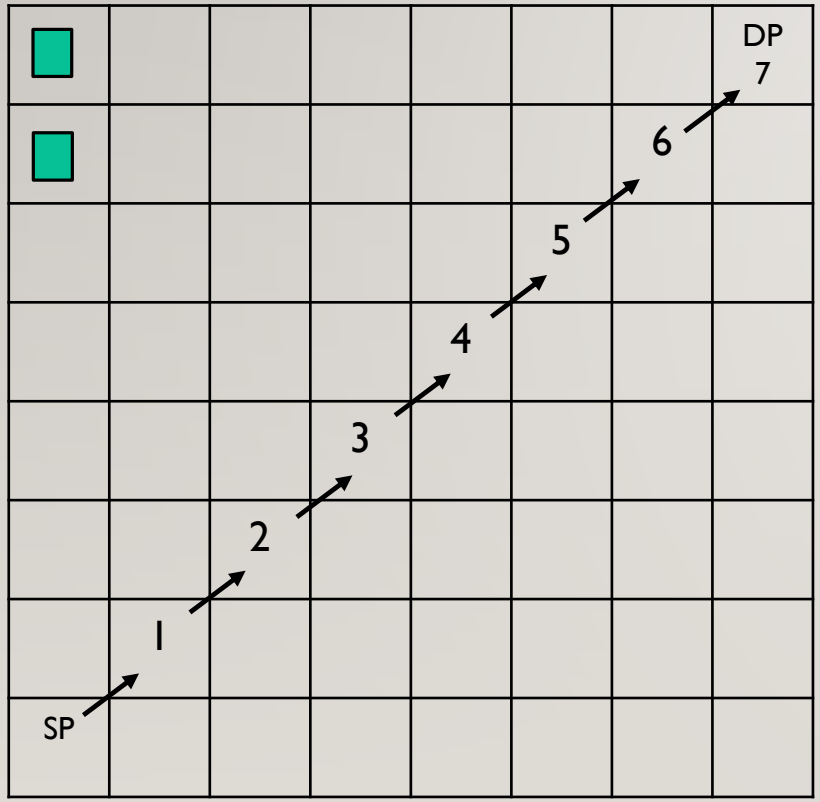
Inference: As the number of obstacles increase, the time of computation and time to reach the destination also increases. Time increases wrt the complexity of path planning, number of obstacles and according to the number of steps taken.

IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

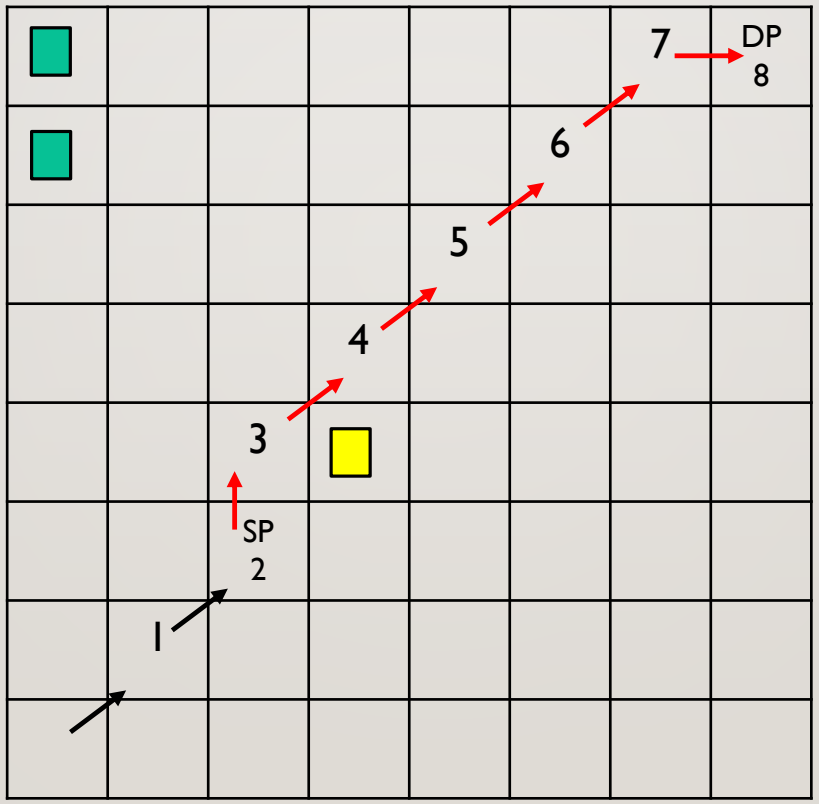
a. Case 1 with two static obstacles and three dynamic obstacles:

SP: Source Point, DP: Destination Point, : Static Obstacle, ⟶: Indicates the path taken, : Dynamic Obstacle,

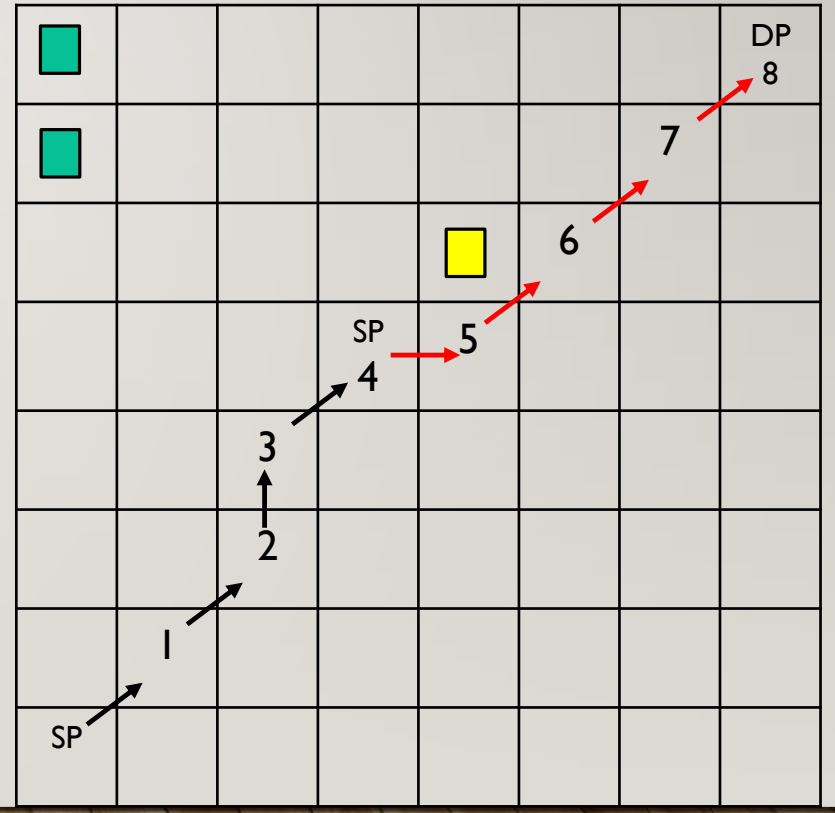
Path planning with no dynamic obstacles



Path planning with 1 dynamic obstacle



Path planning with 2 dynamic obstacles

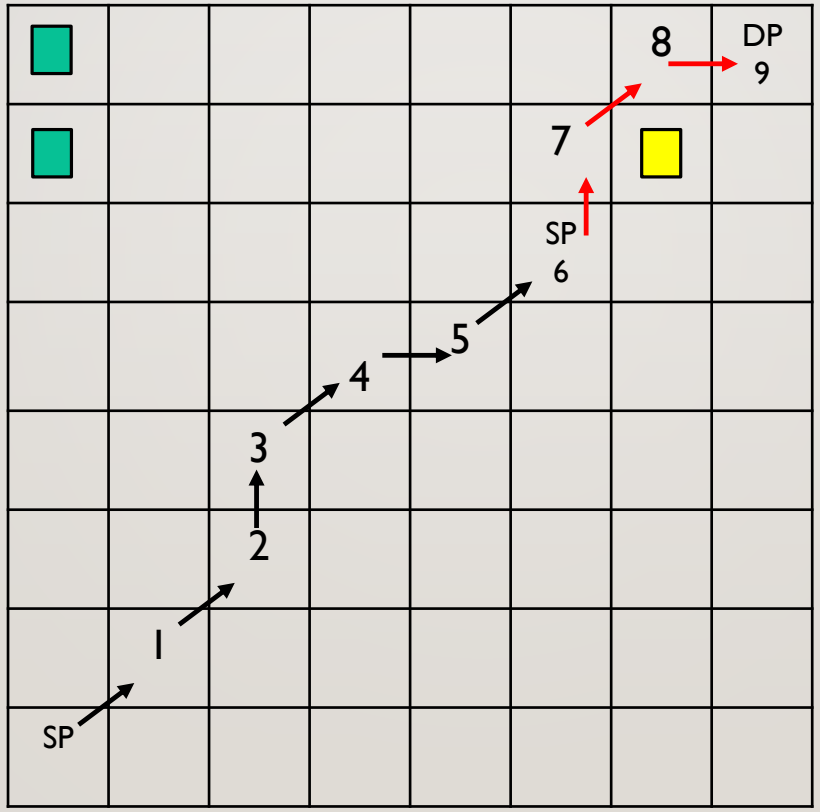


IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

a. Case 1 with two static obstacles and three dynamic obstacles: (continuation)

SP: Source Point, DP: Destination Point, : Static Obstacle, : Indicates the path taken, : Dynamic Obstacle,

Path planning with 3 dynamic obstacles

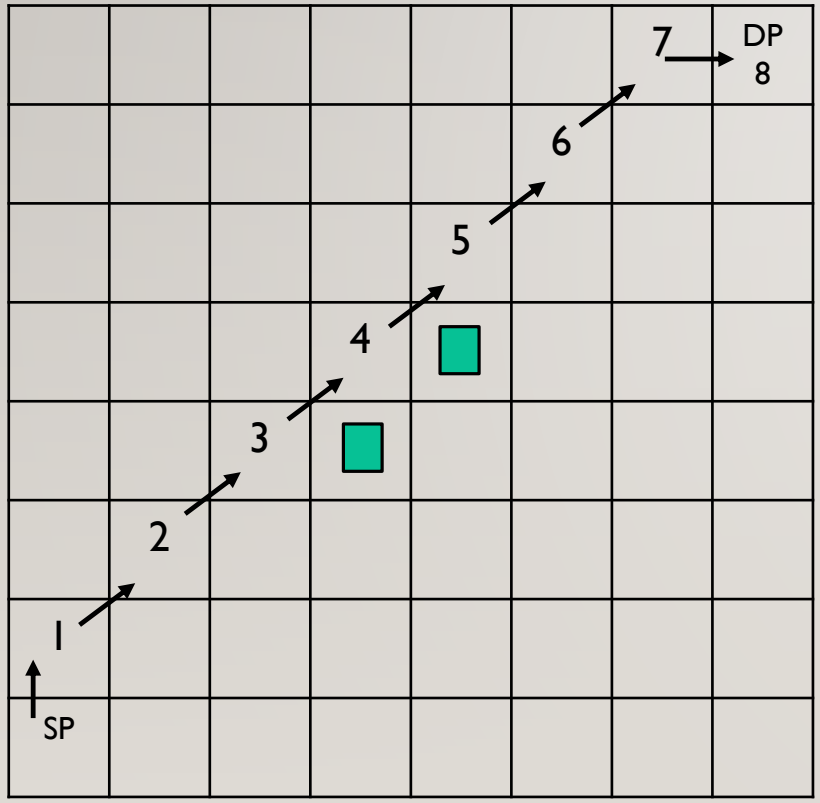


IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

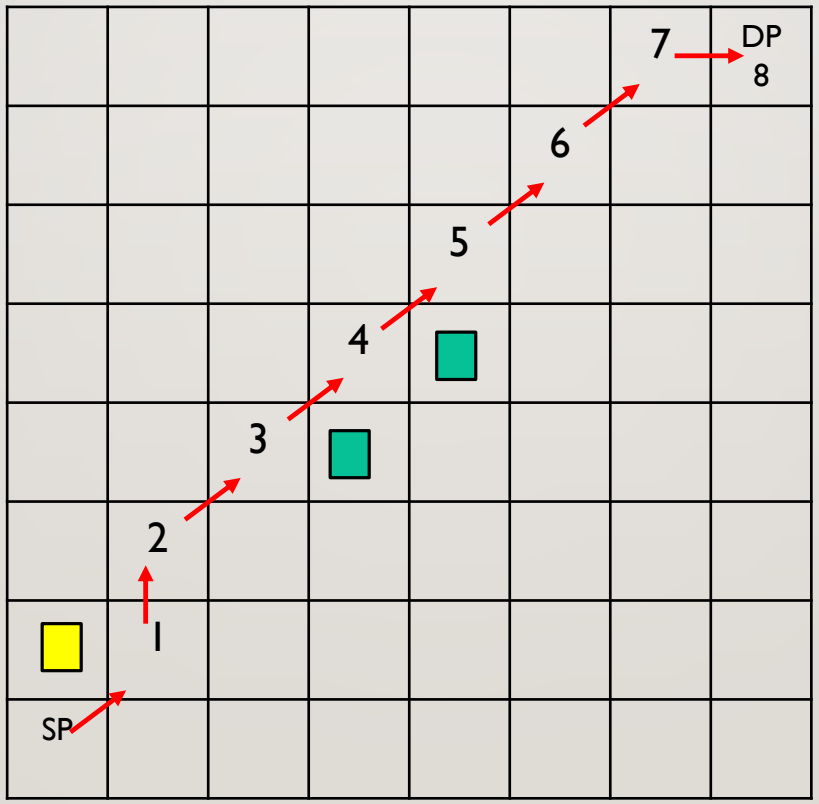
b. Case 2 with two static obstacles and five dynamic obstacles:

SP: Source Point, DP: Destination Point, : Static Obstacle, : Indicates the path taken, : Dynamic Obstacle,

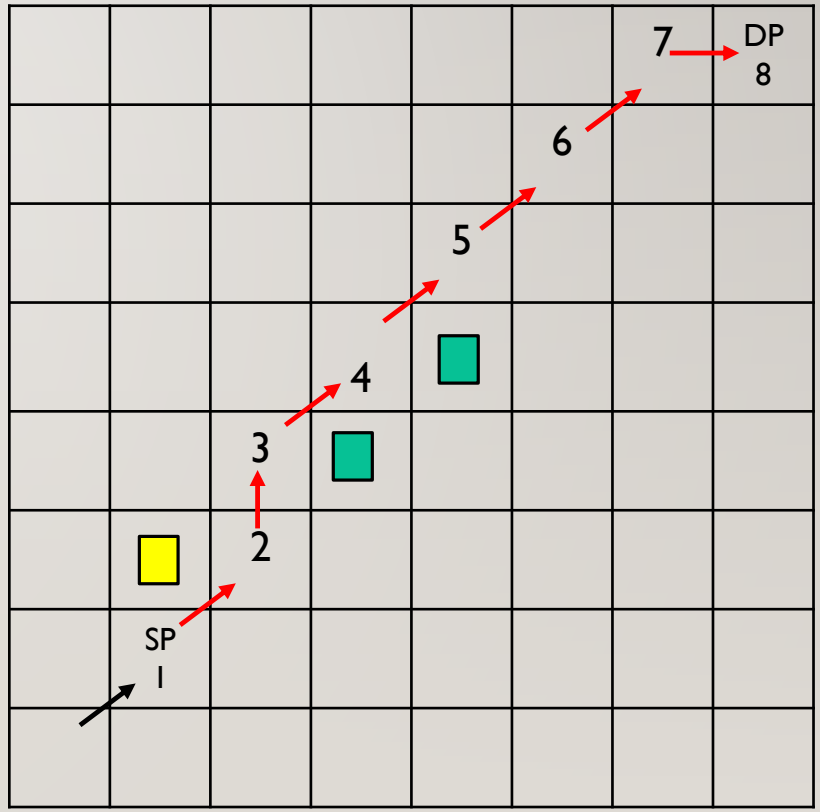
Path planning with no dynamic obstacles



Path planning with 1 dynamic obstacle



Path planning with 2 dynamic obstacles

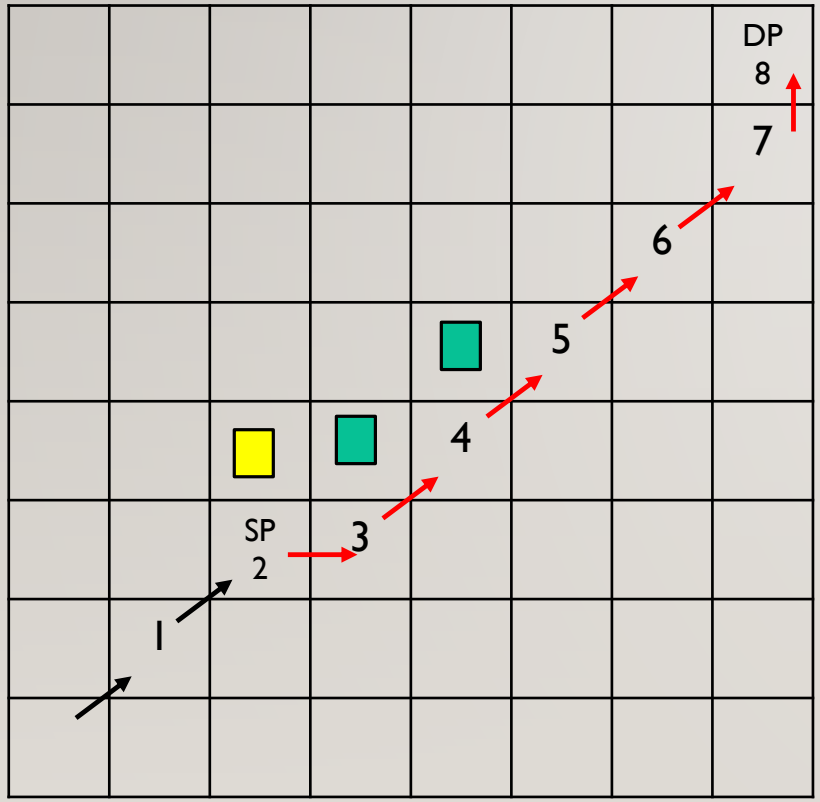


IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

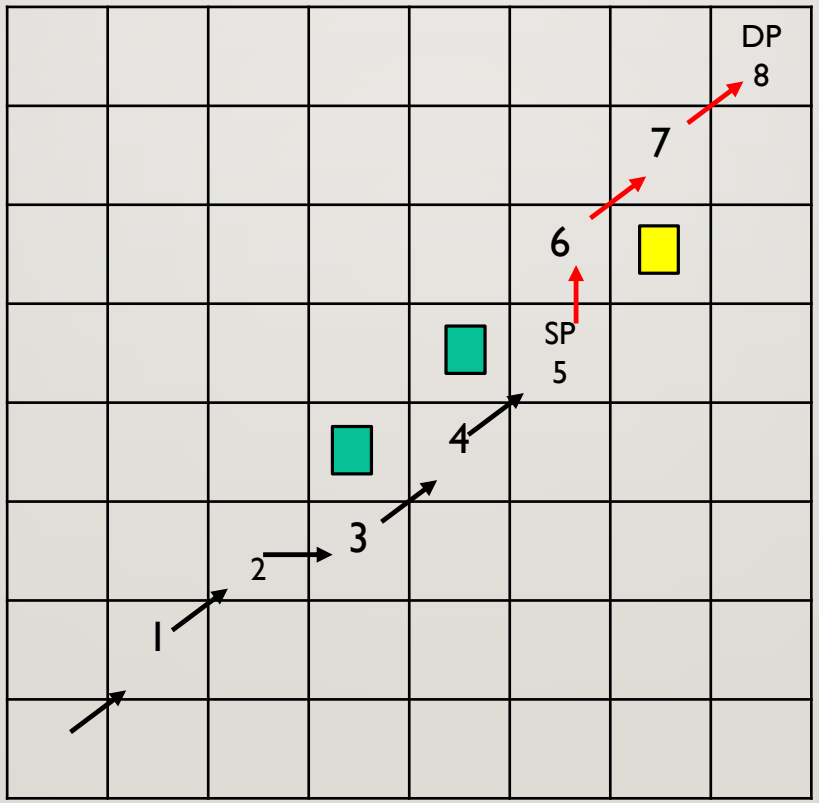
b. Case 2 with two static obstacles and five dynamic obstacles: (continuation)

SP: Source Point, DP: Destination Point, : Static Obstacle, ⟶: Indicates the path taken, : Dynamic Obstacle,

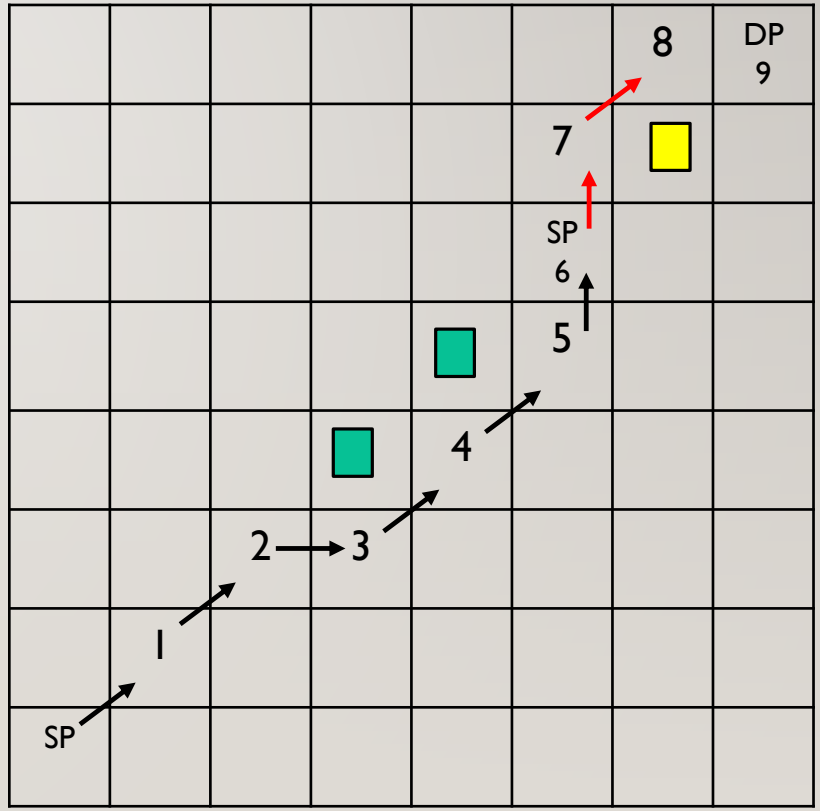
Path planning with 3 dynamic obstacles



Path planning with 4 dynamic obstacles



Path planning with 5 dynamic obstacles

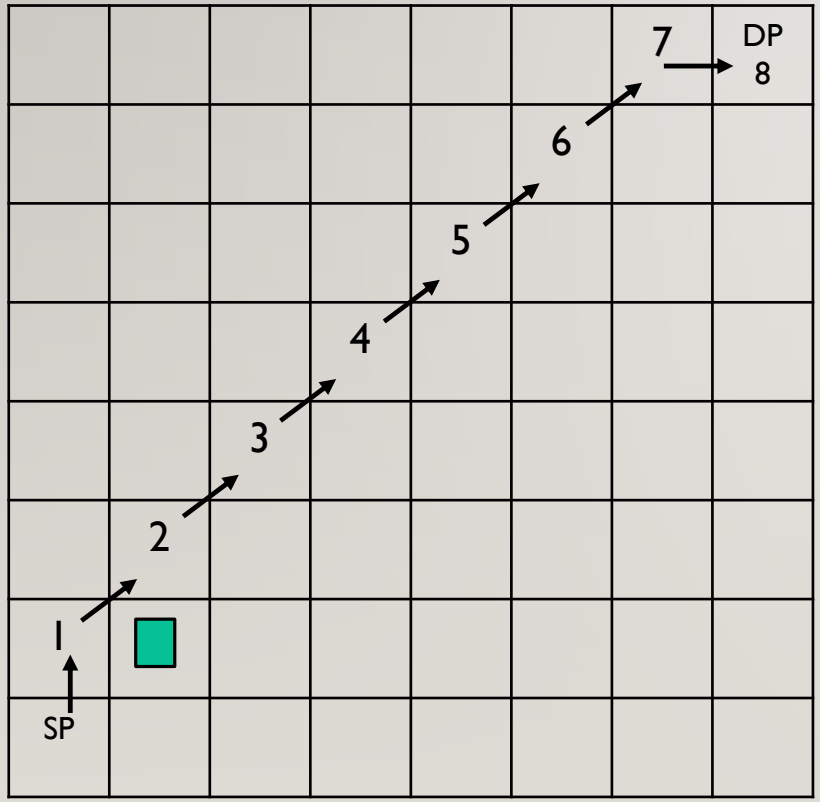


IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

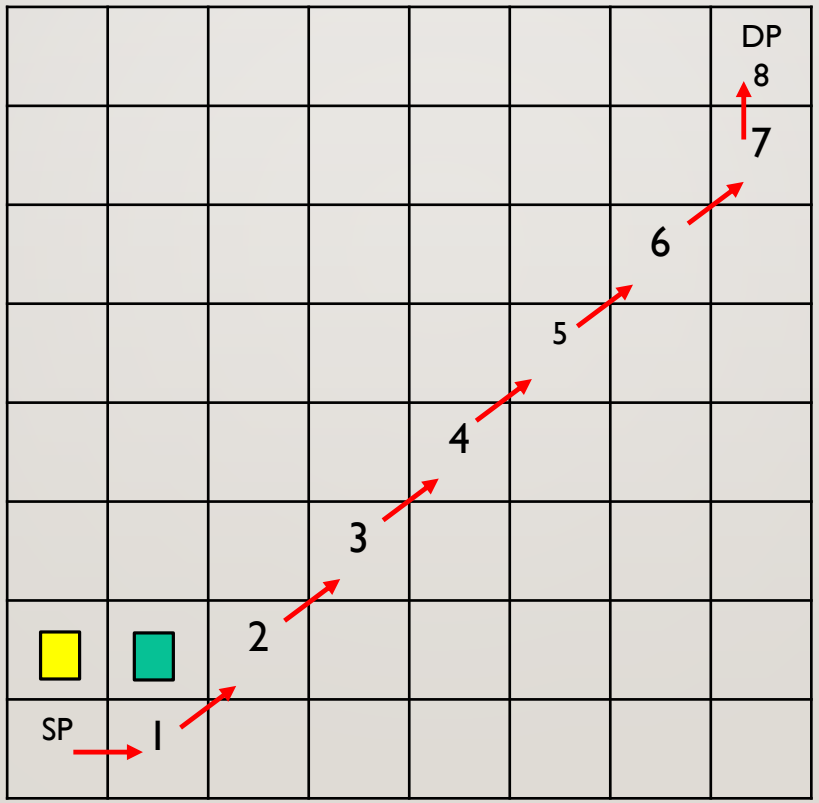
c. Case 1with one static obstacle and eight dynamic obstacles:

SP: Source Point, DP: Destination Point, : Static Obstacle, : Indicates the path taken, : Dynamic Obstacle,

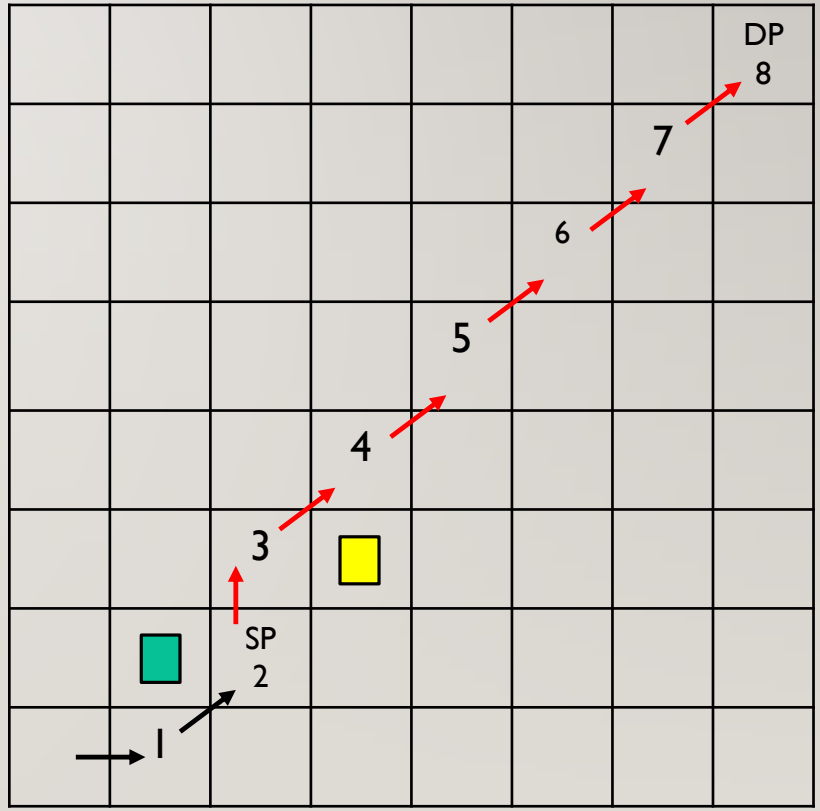
Path planning with no dynamic obstacles



Path planning with 1 dynamic obstacles



Path planning with 2 dynamic obstacles

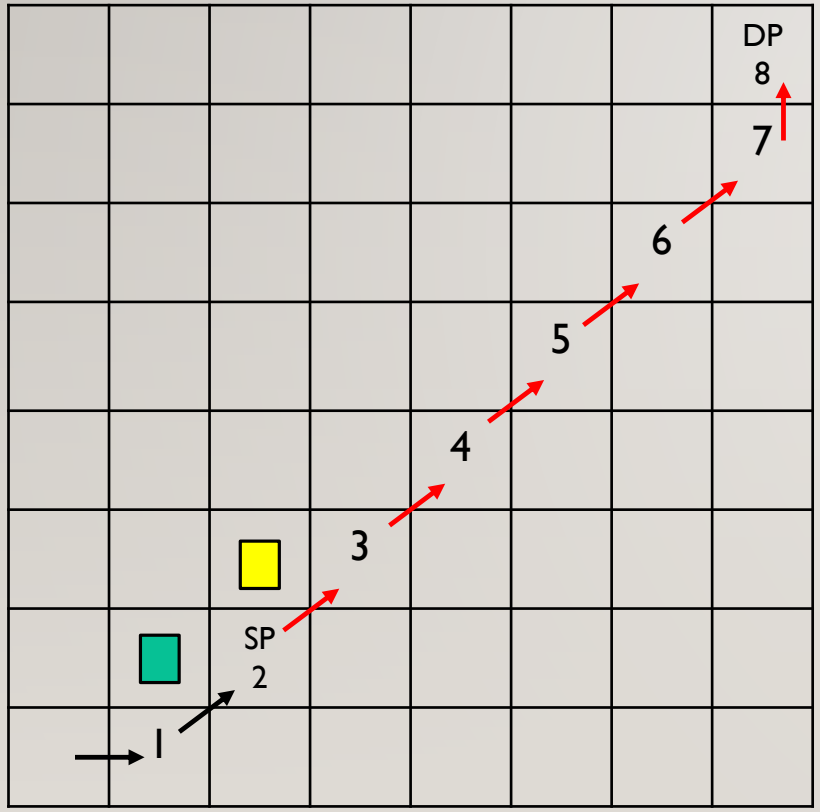


IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

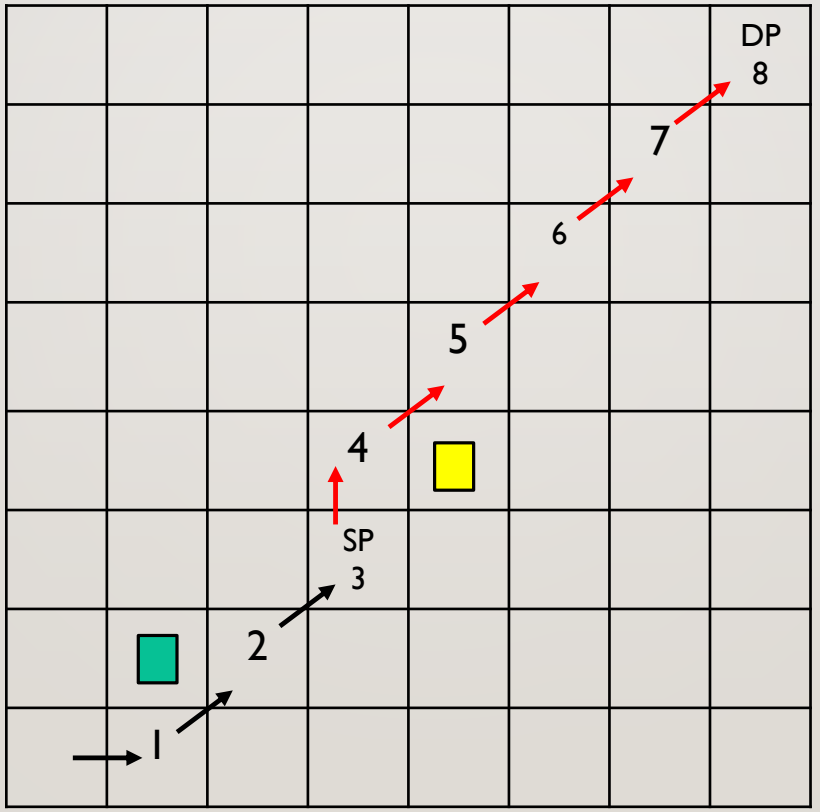
c. Case 1with one static obstacle and eight dynamic obstacles: (continuation)

SP: Source Point, DP: Destination Point, : Static Obstacle, : Indicates the path taken, : Dynamic Obstacle,

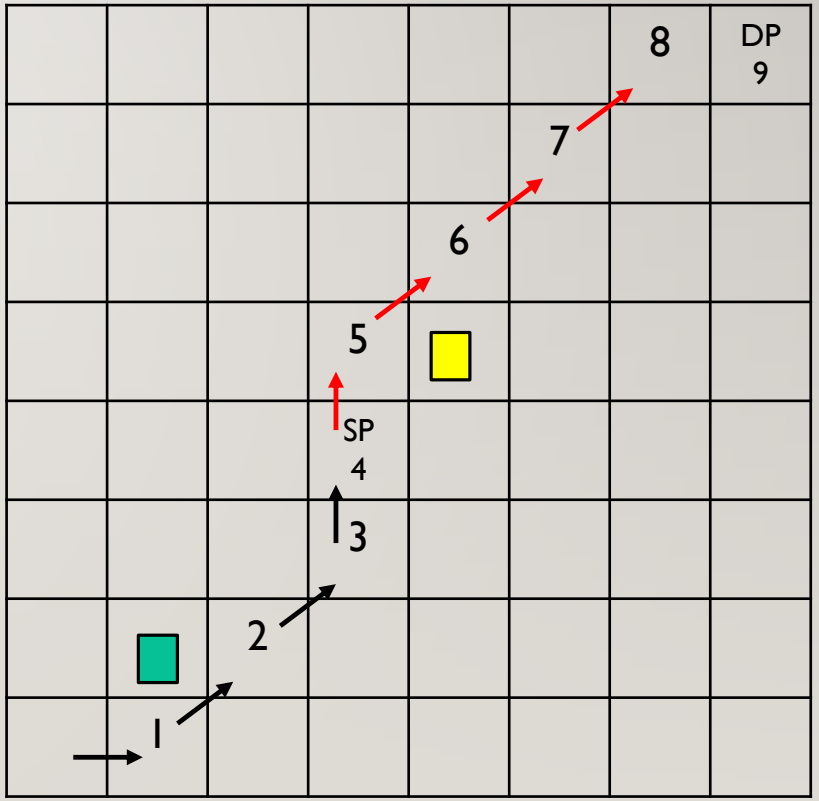
Path planning with 3 dynamic obstacles



Path planning with 4 dynamic obstacles



Path planning with 5 dynamic obstacles

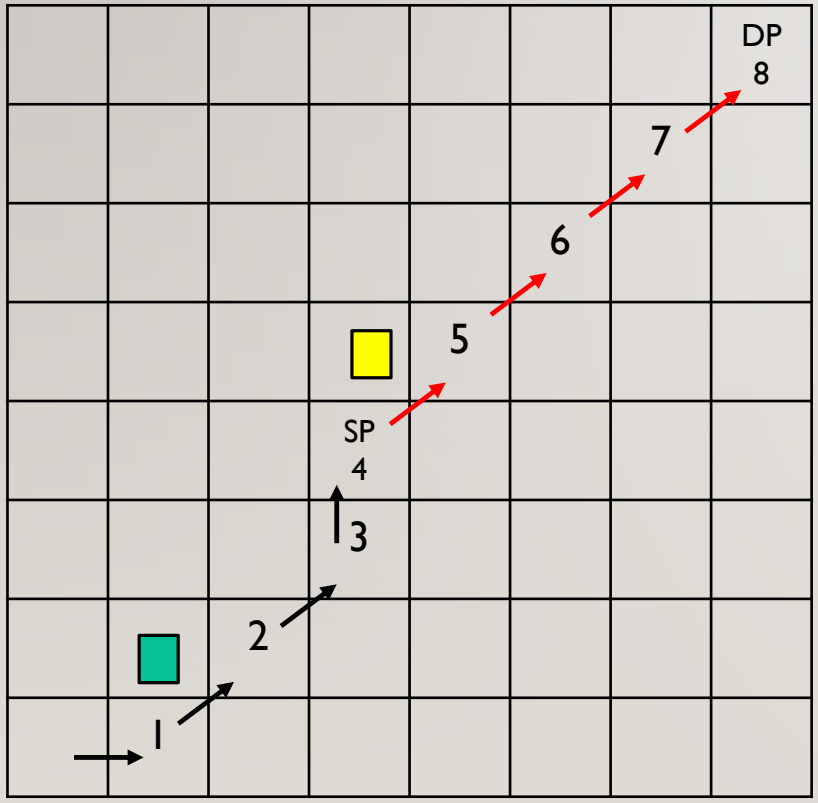


IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

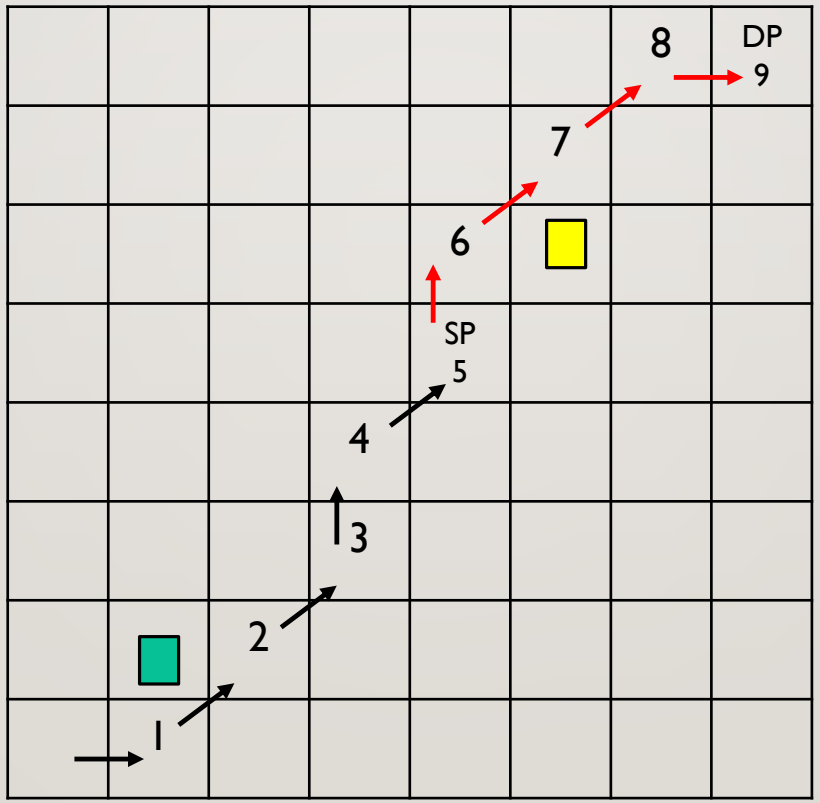
c. Case 1with one static obstacle and eight dynamic obstacles: (continuation)

SP: Source Point, DP: Destination Point, : Static Obstacle, : Indicates the path taken, : Dynamic Obstacle,

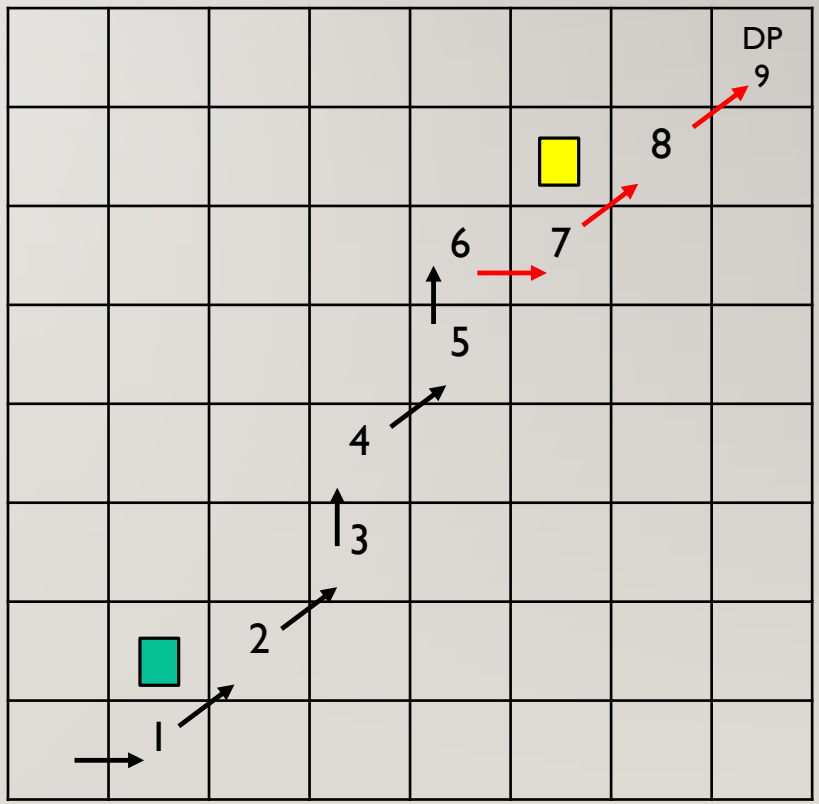
Path planning with 6 dynamic obstacles



Path planning with 7 dynamic obstacles



Path planning with 8 dynamic obstacles



IMPLEMENTATION OF SARSA ALGORITHM FOR MORE THAN ONE DYNAMIC OBSTACLES FOR AN 8X8 GRID IN HARDWARE

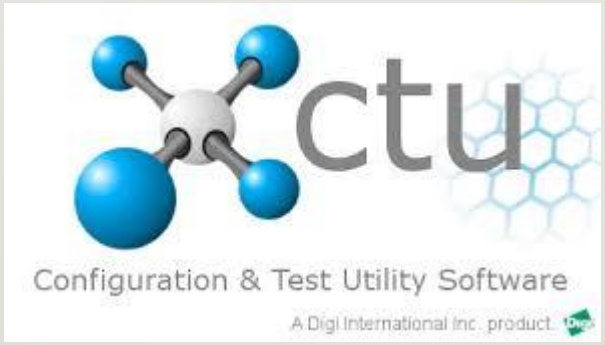
d. Analysis

SL NO	No of Static Obstacles	Position of Static Obstacles	No of Dynamic Obstacles	Position of Dynamic Obstacles	Time Taken (in sec)
1	0	(1.5,1.5)	0	-	45.63
2	1	(1.5,1.5)	1	(0.5,1.5)	48.07
3	2	(1.5,1.5)	2	(0.5,1.5),(3.5,2.5)	54.58
4	3	(1.5,1.5)	3	(0.5,1.5),(3.5,2.5),(2.5,2.5)	59.32
5	4	(1.5,1.5)	4	(0.5,1.5),(3.5,2.5),(2.5,2.5),(4.5,3.5)	65.81
6	5	(1.5,1.5)	5	(0.5,1.5),(3.5,2.5),(2.5,2.5),(4.5,3.5),(4.5,4.5)	77.06
7	6	(1.5,1.5)	6	(0.5,1.5),(3.5,2.5),(2.5,2.5),(4.5,3.5),(4.5,4.5),(3.5,4.5)	77.60
8	7	(1.5,1.5)	7	(0.5,1.5),(3.5,2.5),(2.5,2.5),(4.5,3.5),(4.5,4.5),(3.5,4.5),(5.5,5.5)	88.30
9	8	(1.5,1.5)	8	(0.5,1.5),(3.5,2.5),(2.5,2.5),(4.5,3.5),(4.5,4.5),(3.5,4.5),(5.5,5.5),(5.5,6.5)	93.49

CONFIGURATION OF ZIGBEE MODULE AS MASTER AND SLAVE AND COMMUNICATION BETWEEN MBED AND PC USING ZIGBEE MODULE FOR GUI

a. Configuring Zigbee Modules as master and slave:

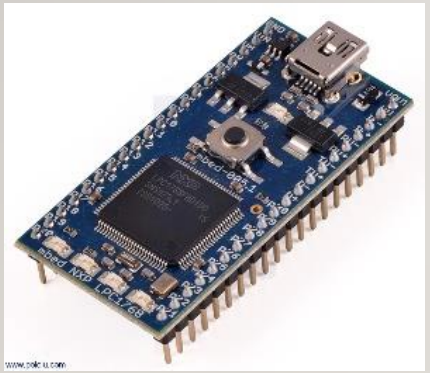
MASTER



SLAVE

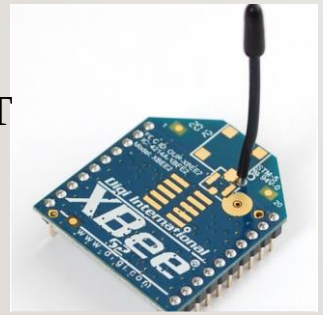
CONFIGURATION OF ZIGBEE MODULE AS MASTER AND SLAVE AND COMMUNICATION BETWEEN MBED AND PC USING ZIGBEE MODULE FOR GUI

b. Communication between mbed and pc using ZigBee module:



MBED Microcontroller

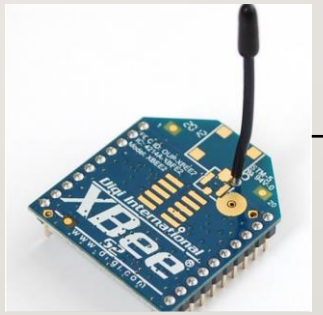
UART



MASTER

Powered up from mbed

ZigBee modules



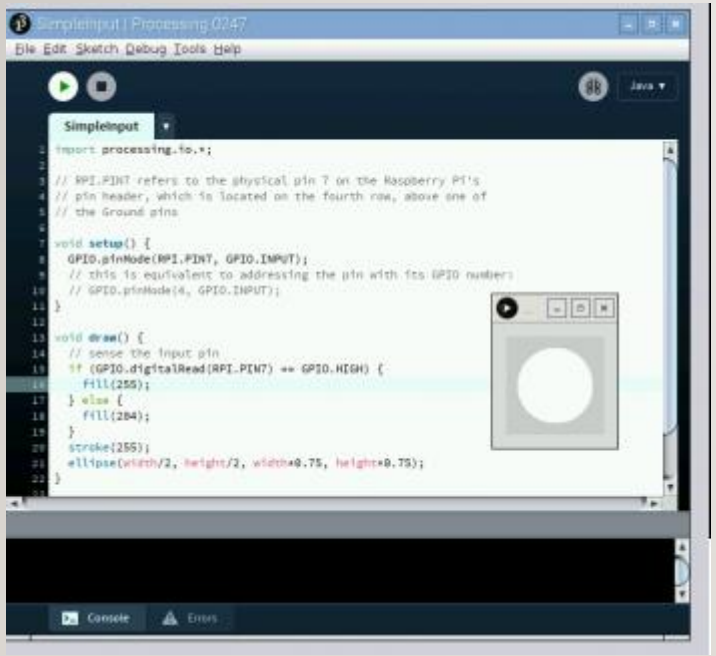
SLAVE

Powered up from external power source

Output tested in Tera term

TOOL USED FOR PLOTTING THE PATH TRAVELLED IN A GUI

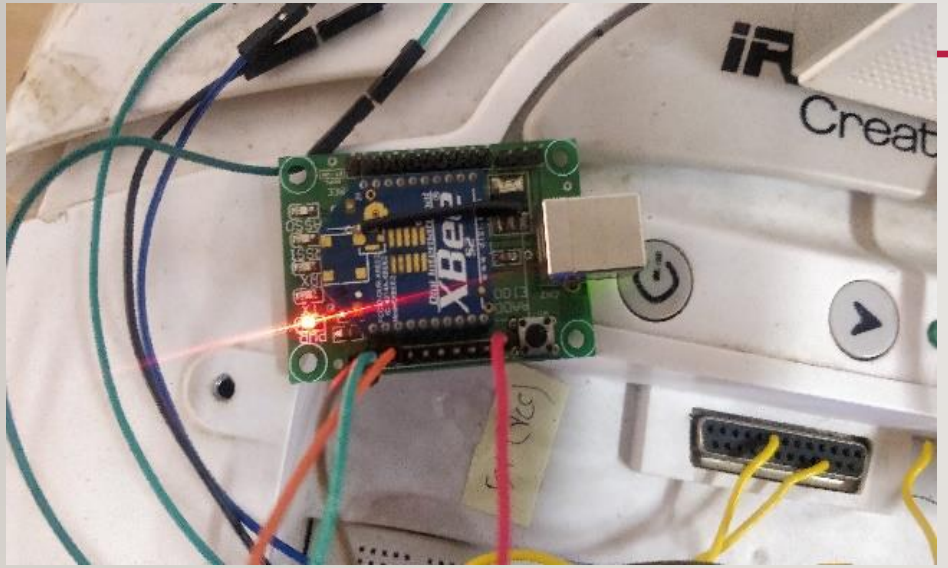
a. Processing Tool



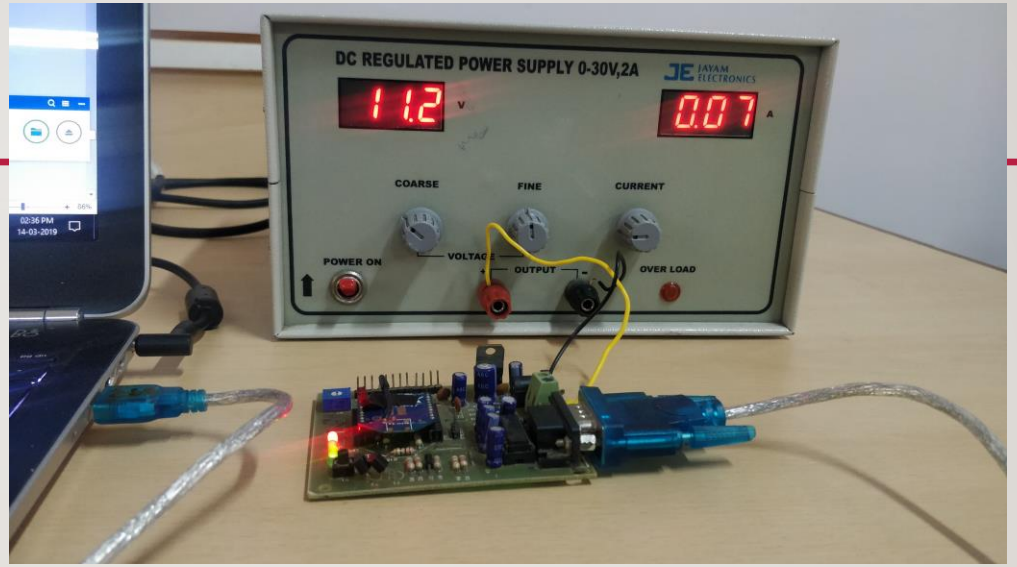
- ✓ Open Source
- ✓ Contains GUI
- ✓ C Programming

PLOTTING THE PATH TRAVELLED IN PROCESSING ACCORDING TO THE VALUES SEND FROM MBED USING ZIGBEE COMMUNICATION

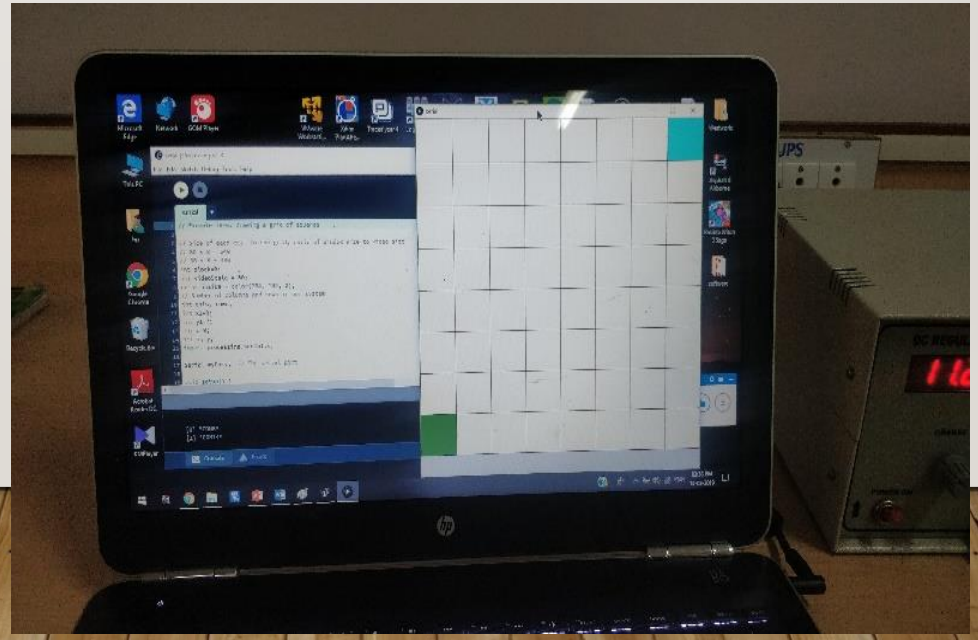
a. Hardware Results:



Transmitter Module (ZigBee) interfaced with mbed microcontroller



Receiver Module (ZigBee) serially connected to pc

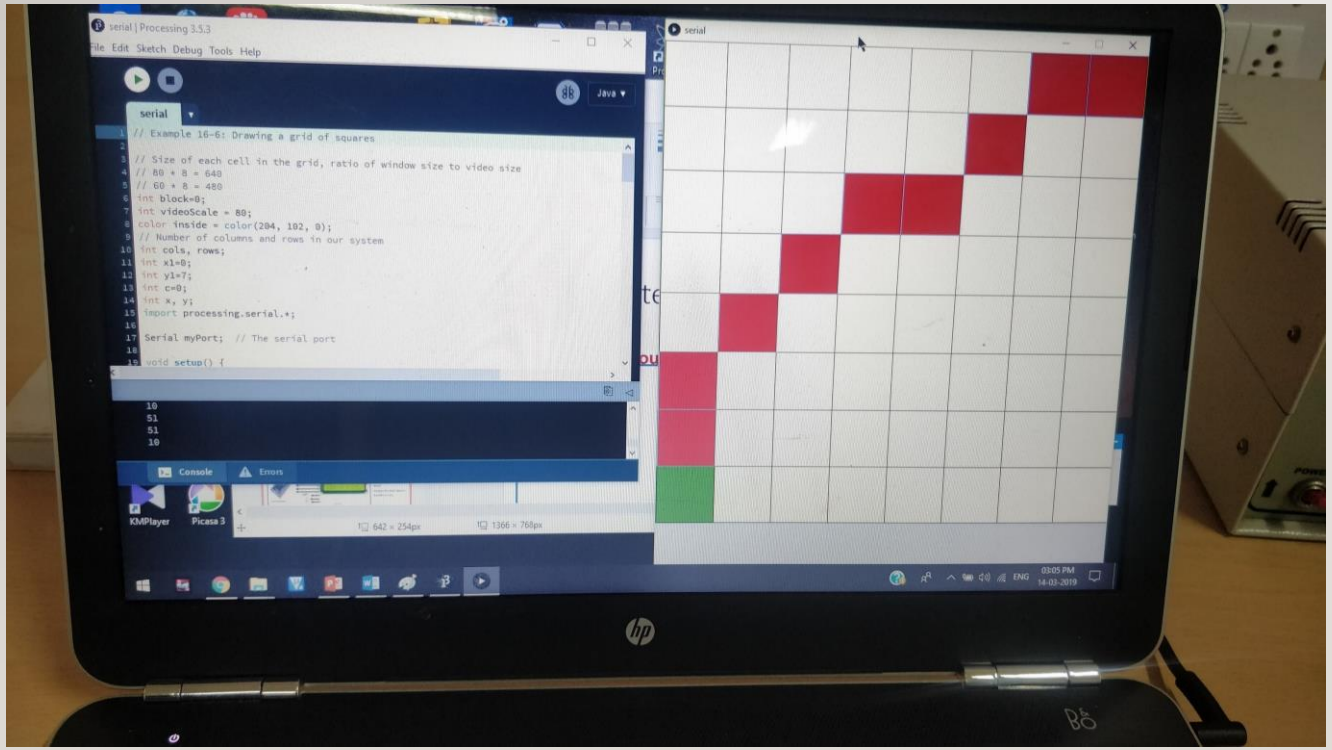


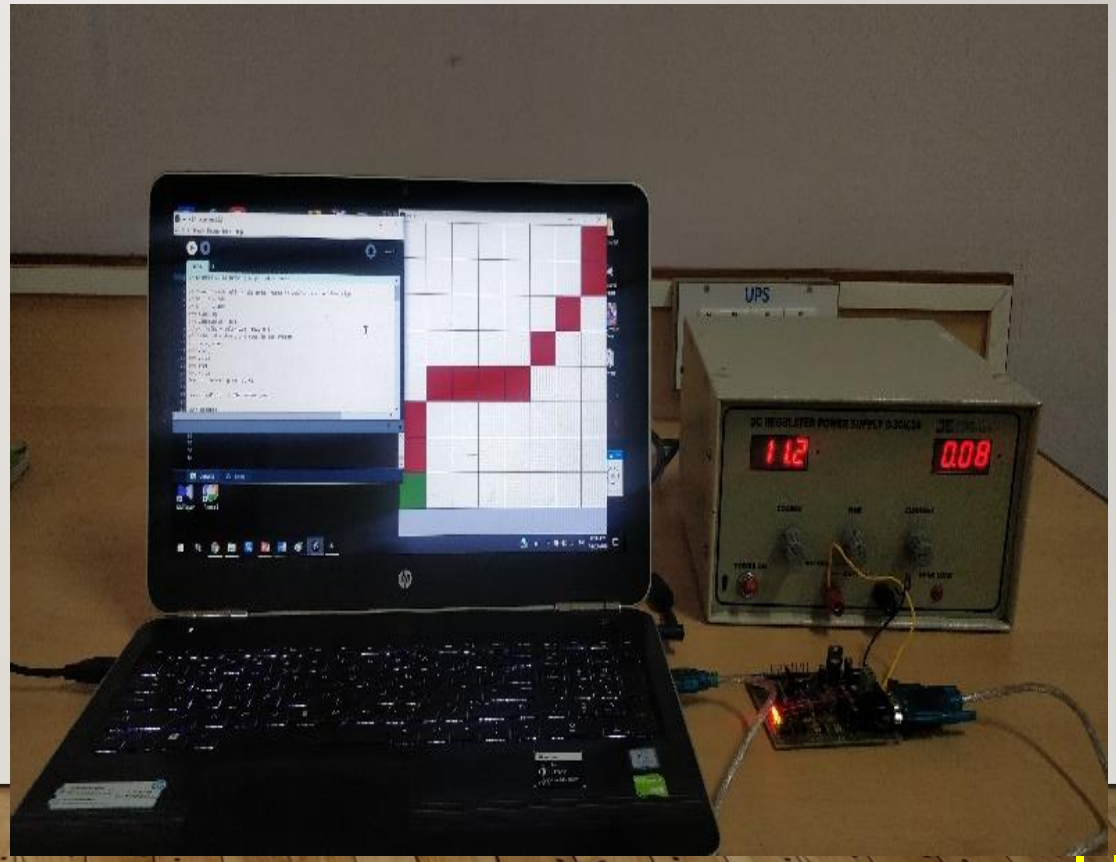
GUI results before communication

PLOTTING THE PATH TRAVELLED IN PROCESSING ACCORDING TO THE VALUES SEND FROM MBED USING ZIGBEE COMMUNICATION

a. Hardware Results: (continuation)

GUI results after communication

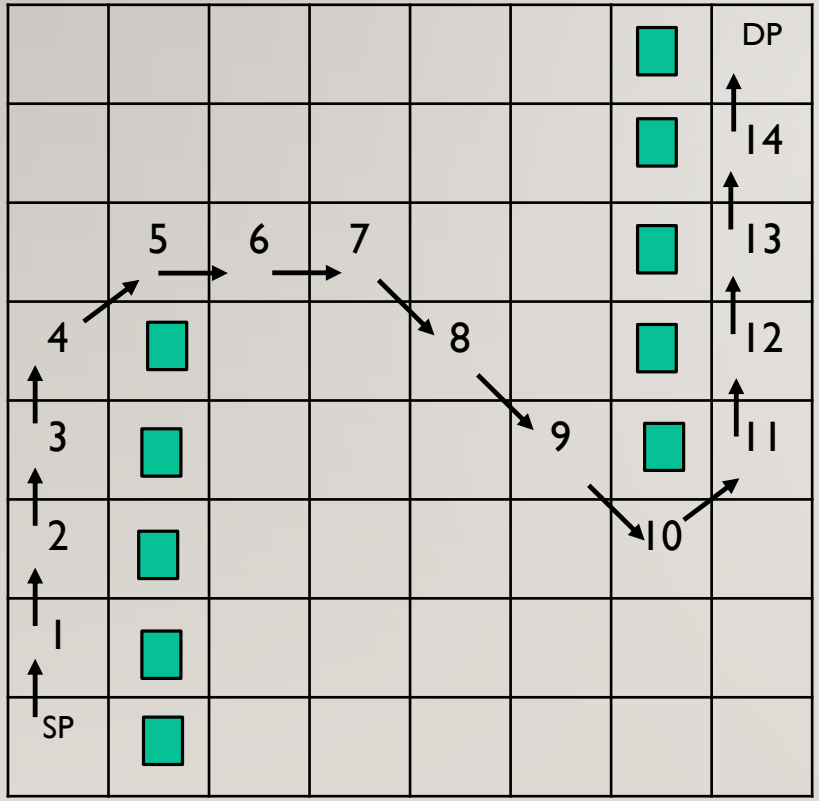


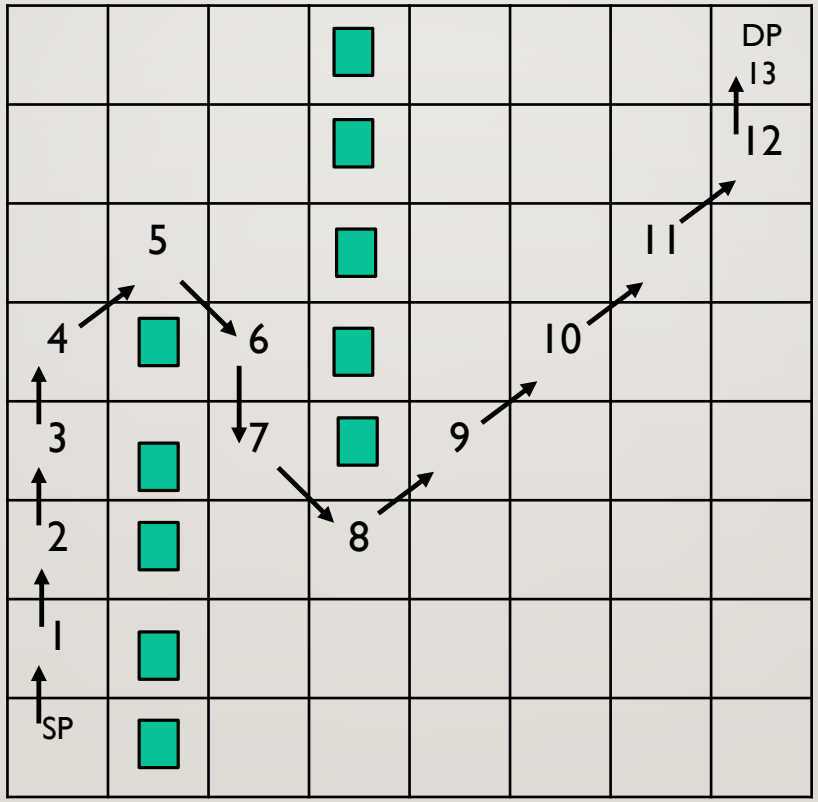


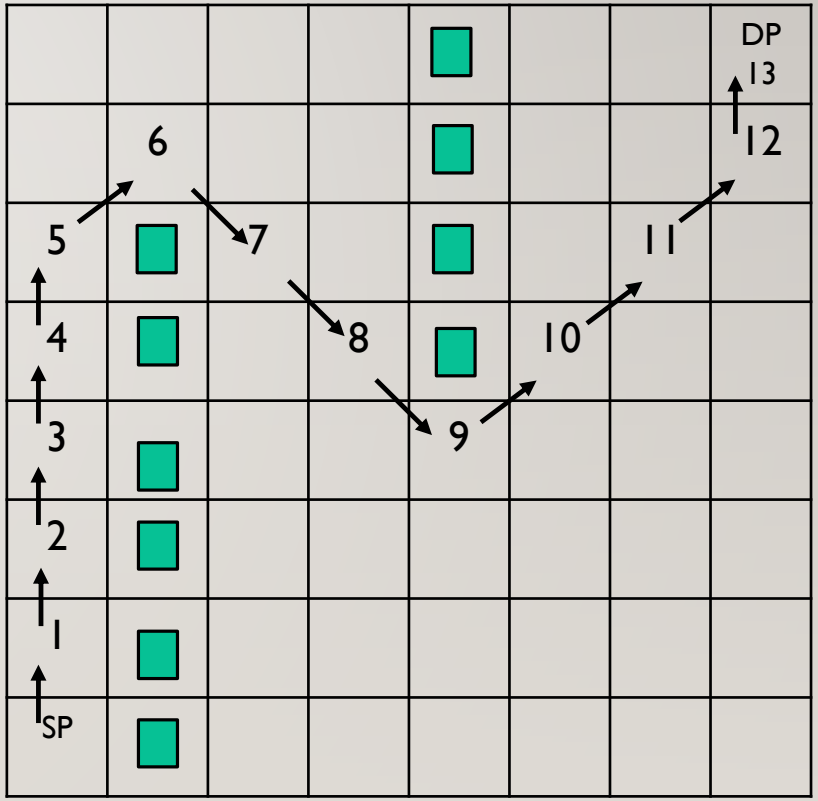
IMPLEMENTATION OF SARSA ALGORITHM WITH BACKWARD MOVEMENT FOR AN 8X8 GRID IN IROBOT

a. Cases tested with backward movement with static obstacles

SP: Source Point, DP: Destination Point, ■: Static Obstacle, →: Indicates the path taken





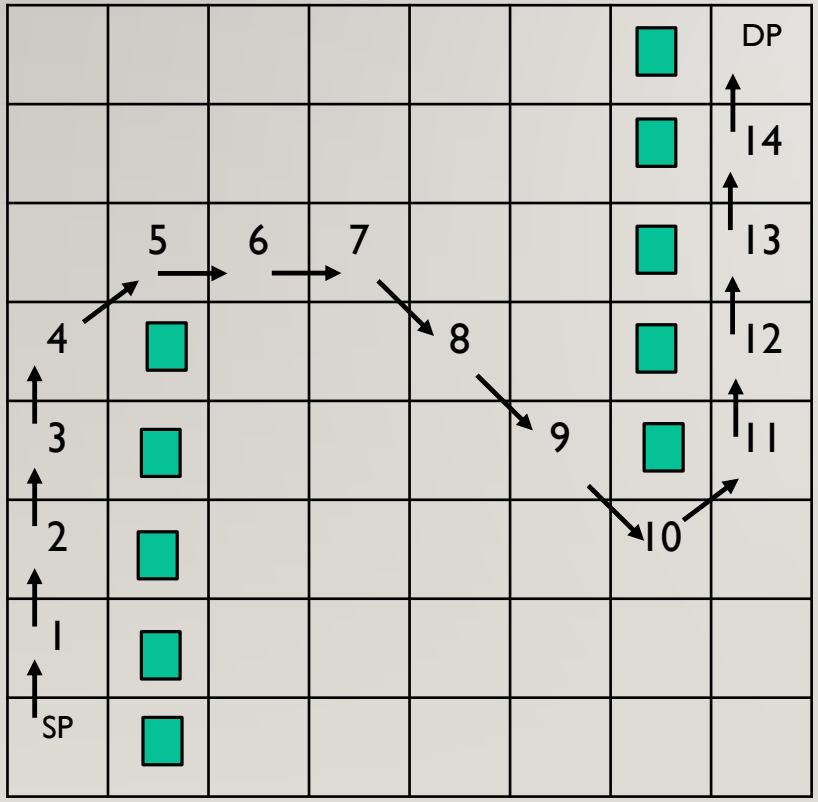


IMPLEMENTATION OF SARSA ALGORITHM WITH BACKWARD MOVEMENT FOR AN 8X8 GRID IN IROBOT

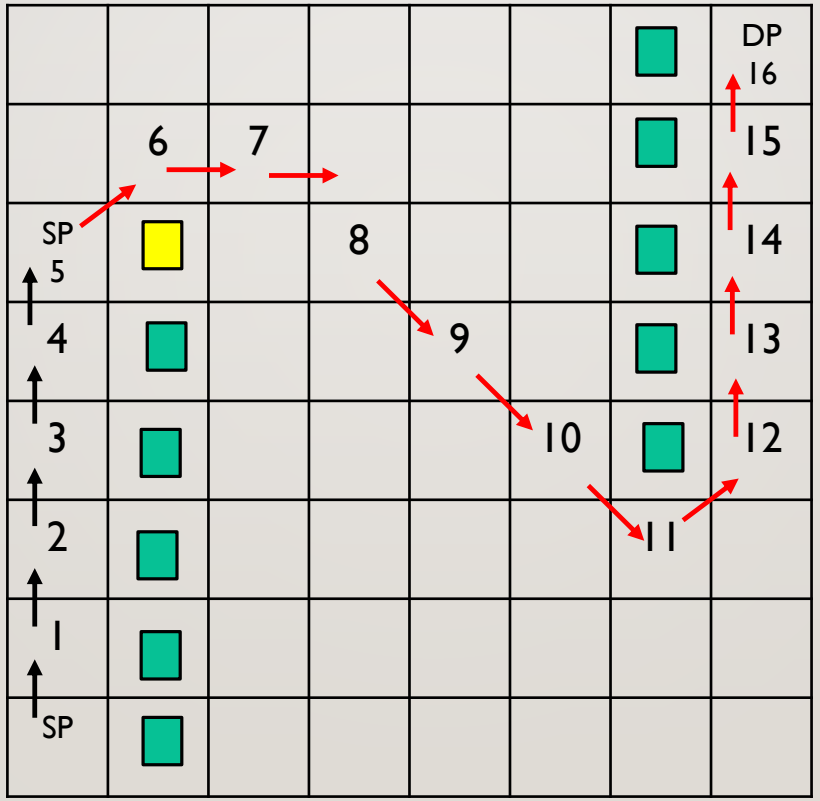
b. Case tested with backward movement with 10 static and 2 dynamic obstacles

SP: Source Point, DP: Destination Point, : Static Obstacle, ⟶: Indicates the path taken, : Dynamic Obstacle

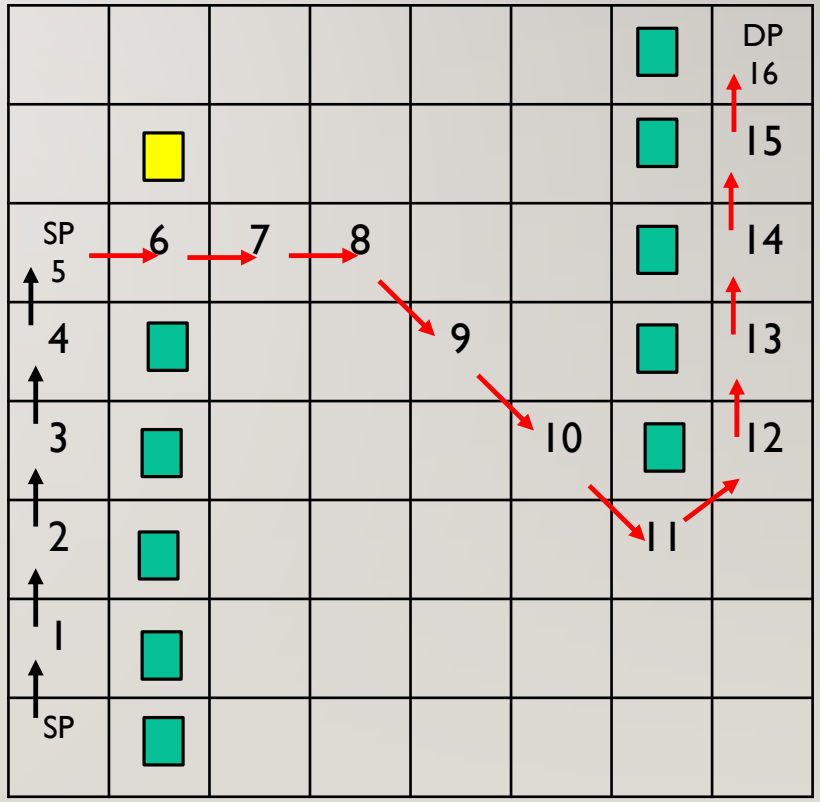
Path planning with no dynamic obstacles



Path planning with 1 dynamic obstacle



Path planning with 2 dynamic obstacles

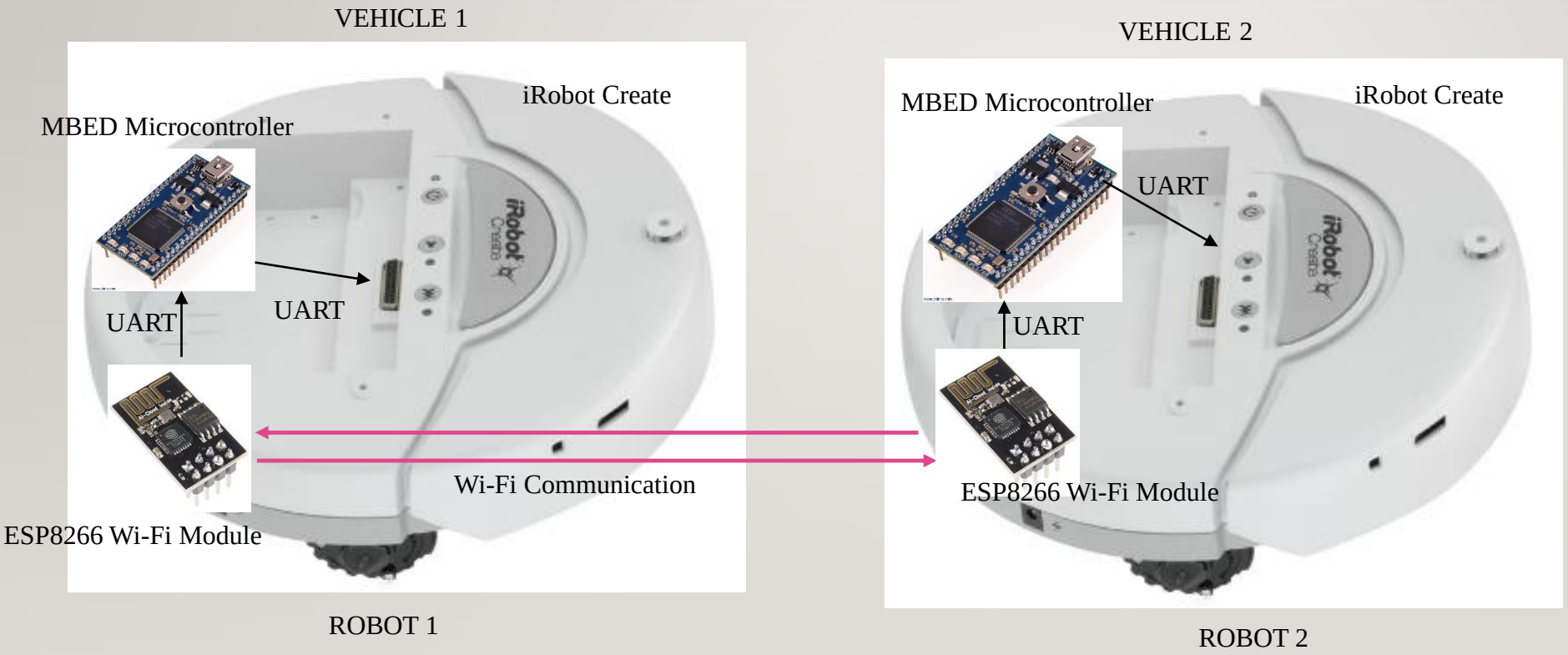


UNCERTAINTIES THAT MIGHT CAUSE THE ROBOT **NOT** TO REACH THE DESTINATION PROPERLY

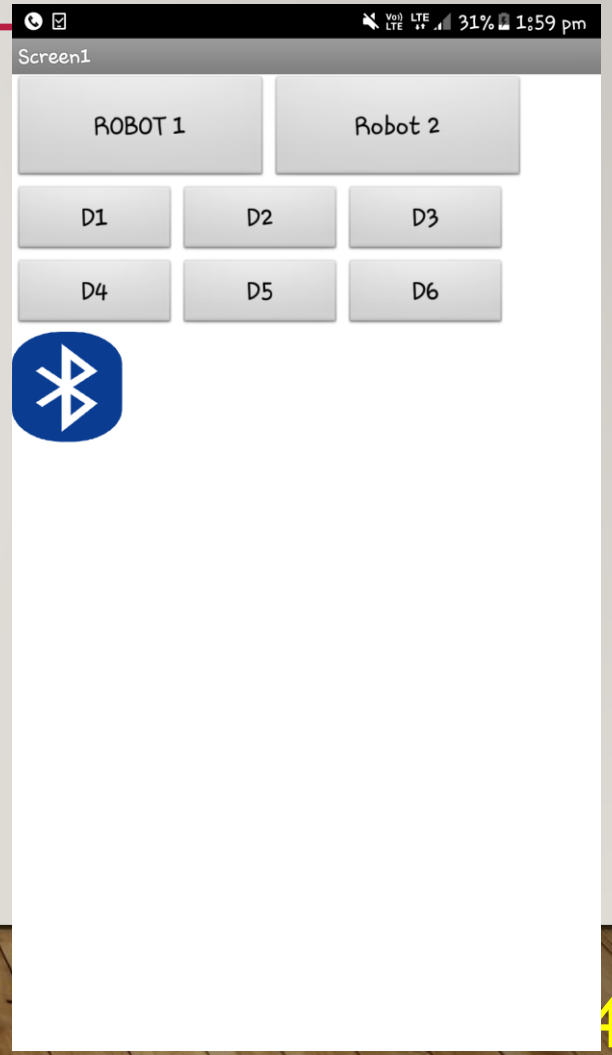
- Wheels of the robot might slip sometimes
- Sensors may be affected by noise
- Obstacles move unpredictably
- Not placing the robot correctly in the source point

IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI COMMUNICATION

a. Block Diagram



b. APP



IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI

COM16 - Tera Term VT

File Edit Setup Control Window Help

```
0,CONNECT
1,CONNECT

OK

---------- Get Connection Status ----------


-------------ESP8266 Hardware Reset-------------

---------- Setting Mode ----------

---------- Connecting to AP ----------
ssid = LAYA    pwd = laya4321
cσ╧RSf╒N[¦f╟N[fτ╪≤Ωf─f╟N?≈
Ai-Thinker Technology Co.,Ltd.

invalid
cσ╨╧RSf╒N[¦f╟N[fτ╪≤Ωf─f╟N?≈
Ai-Thinker Technology Co.,Ltd.

invalid

---------- Setting Connection Mode ----------
AT+CIPMUX=1

OK
WIFI CONNECTED
WIFI GOT IP

---------- Get IP's ----------
AT+CIFSR
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"2e:3a:e8:1d:f4:8b"
+CIFSR:STAIP,"192.168.43.94"
+CIFSR:STAMAC,"2c:3a:e8:1d:f4:8b"

OK

---------- Get Connection Status ----------
AT+CIPSERVER=1,80

OK

---------- Get Conn ----------
AT+CIPSTART=1,"TCP","192.168.43.181",222
1,CONNECT

OK
0,CONNECT

---------- Get Connection Status ----------
```

Flash Magic Terminal - COM 5, 9600

Options

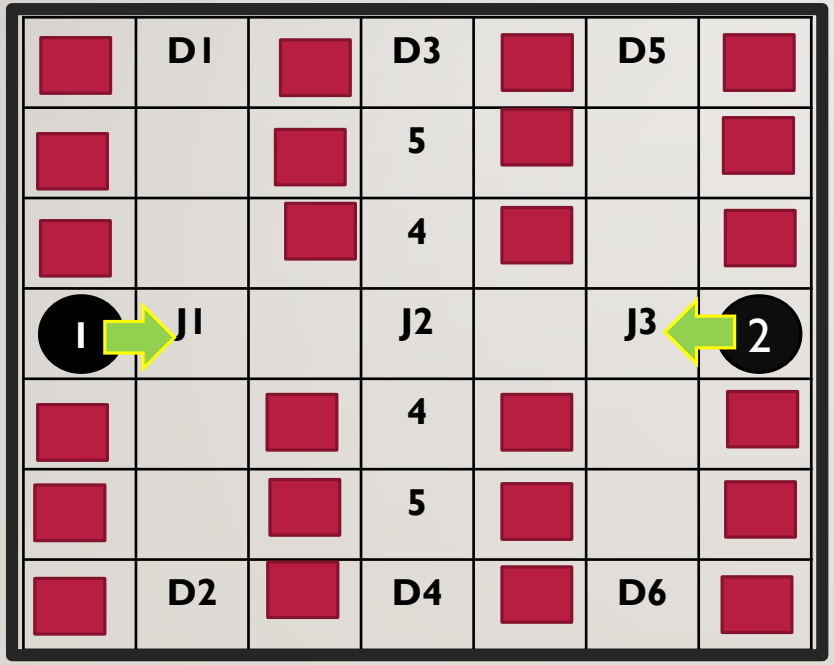
Output >>

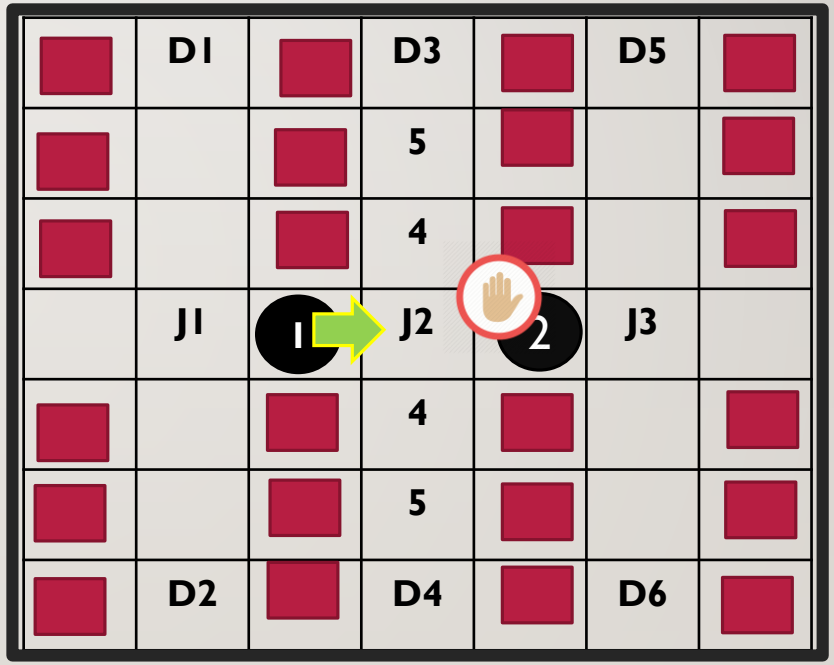
```
invalid
---------- Setting Connection Mode ----------
AT+CIPMUX=1

OK
WIFI CONNECTED
WIFI GOT IP
---------- Get IP's ----------
AT+CIFSR
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"62:01:94:5d:2b:88"
+CIFSR:STAIP,"192.168.43.181"
+CIFSR:STAMAC,"60:01:94:5d:2b:88"

OK
---------- Get Connection Status ----------
AT+CIPSERVER=1,222

OK
---------- Get Conn ----------
AT+CIPSTART=1,"TCP","192.168.43.94",80
1,CONNECT

OK
0,CONNECT
---------- Get Connection Status ----------
```

Input >>

IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI COMMUNICATION

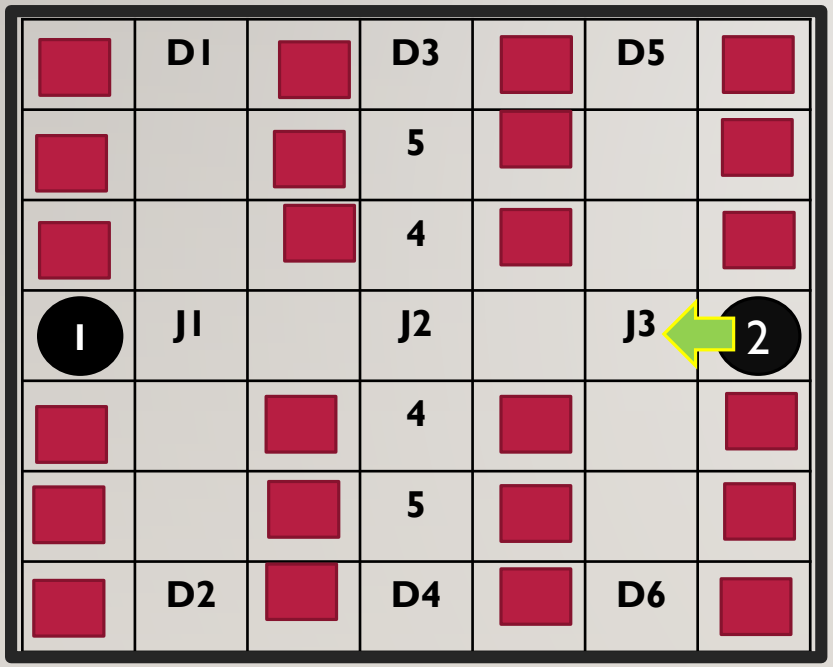
Case 1 where R1 goes to D3 and R2 goes to D4

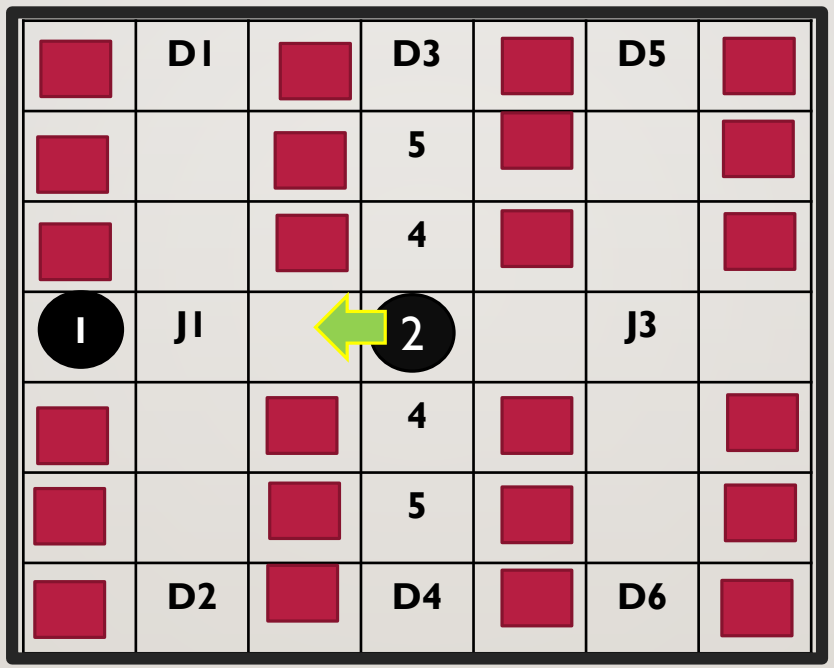


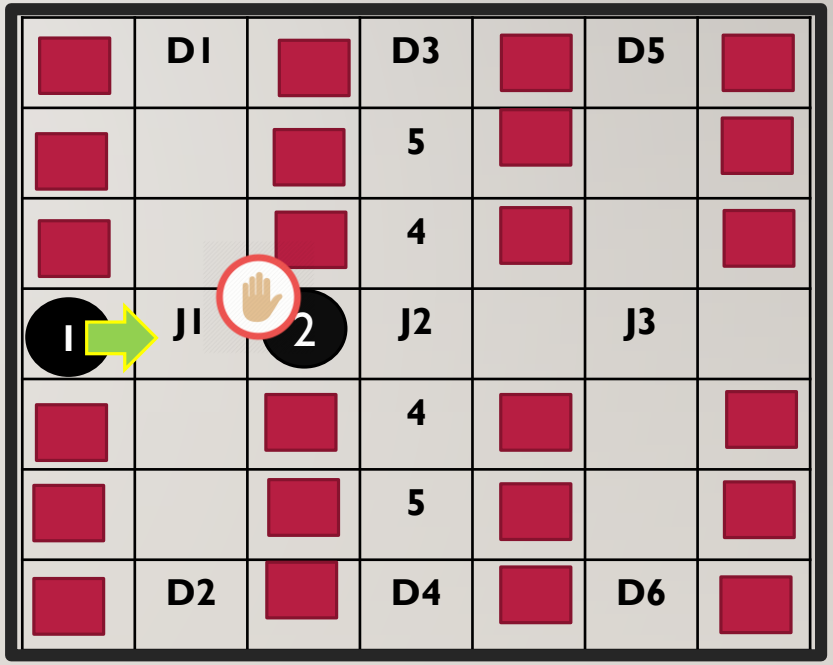


IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI COMMUNICATION

Case 2 where R1 goes to D2 and R2 goes to D1

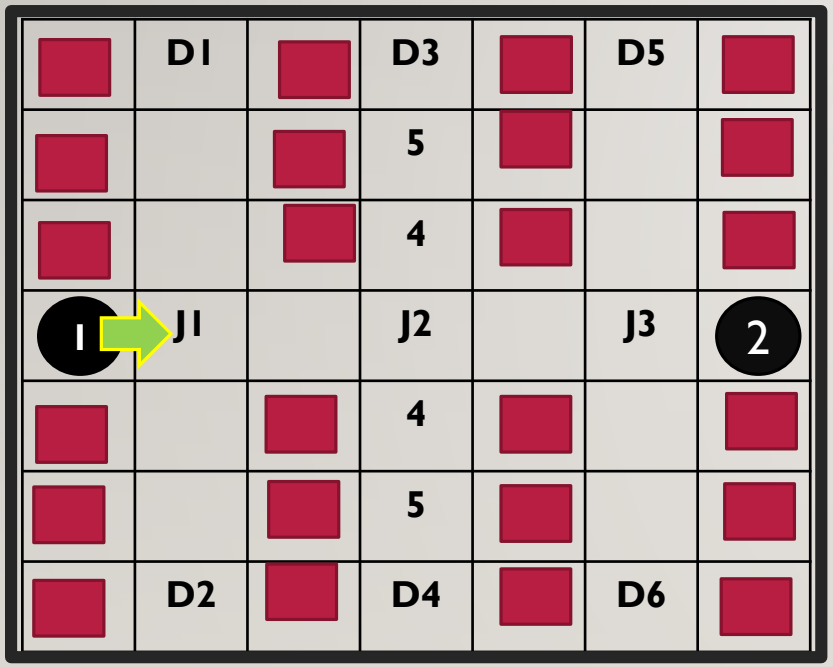


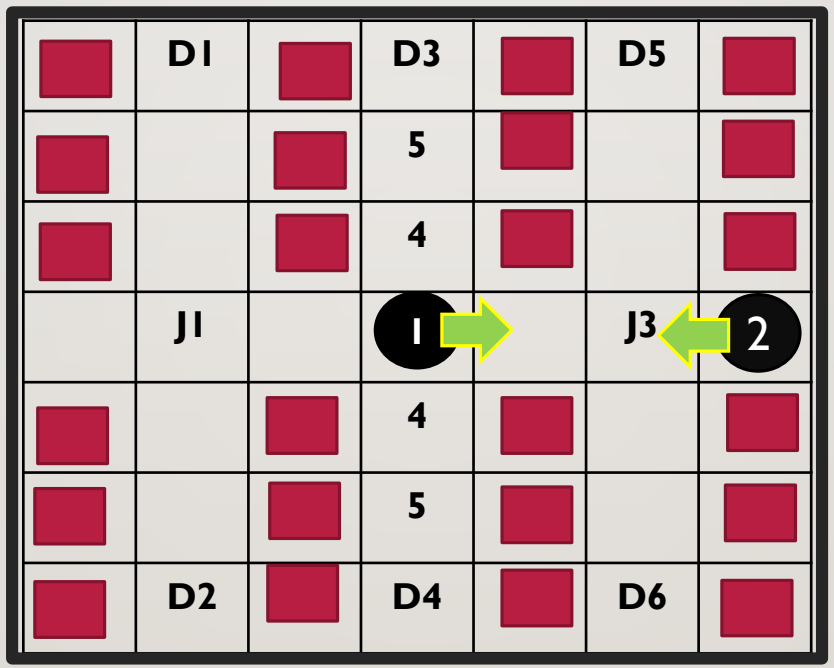


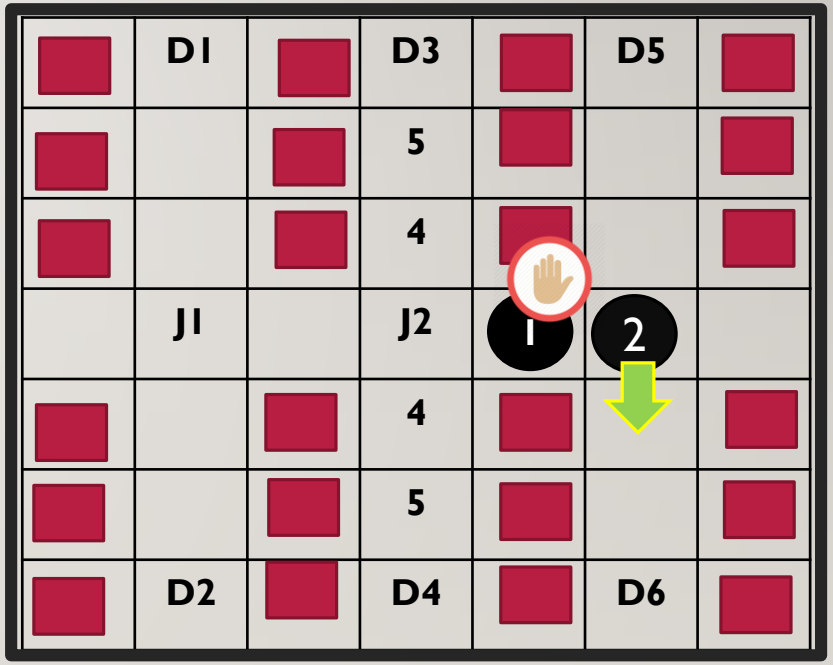


IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI COMMUNICATION

Case 3 where R1 goes to D5 and R2 goes to D6

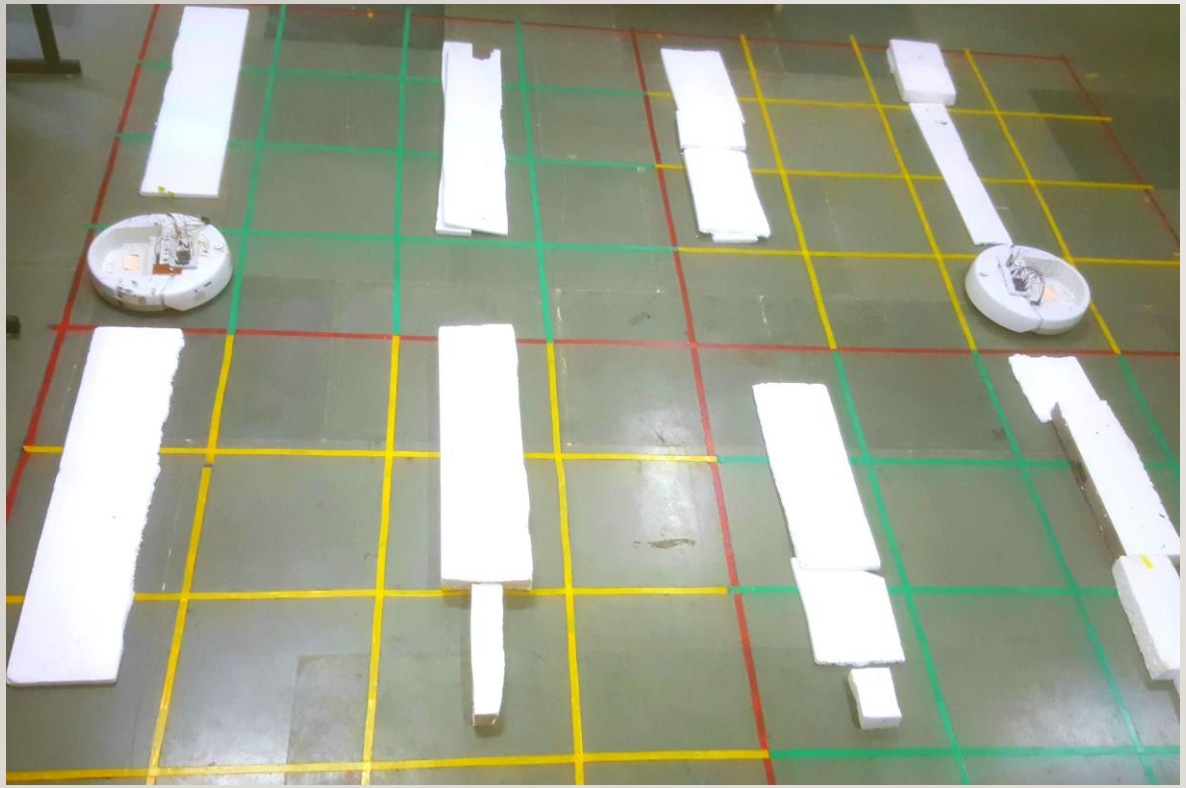


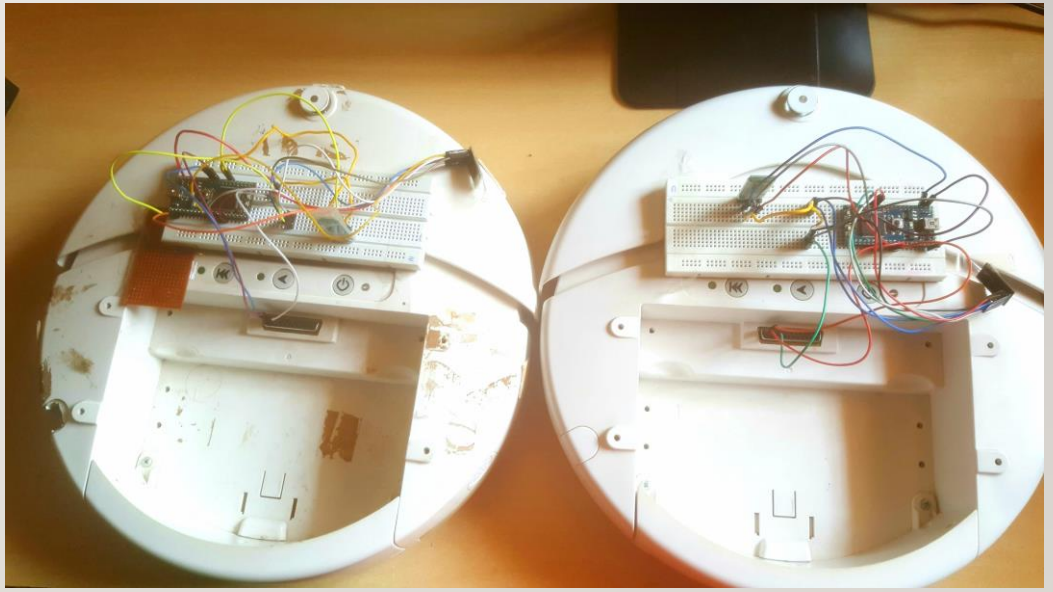




IMPLEMENTATION OF ROBOT TO ROBOT COMMUNICATION USING WI-FI COMMUNICATION

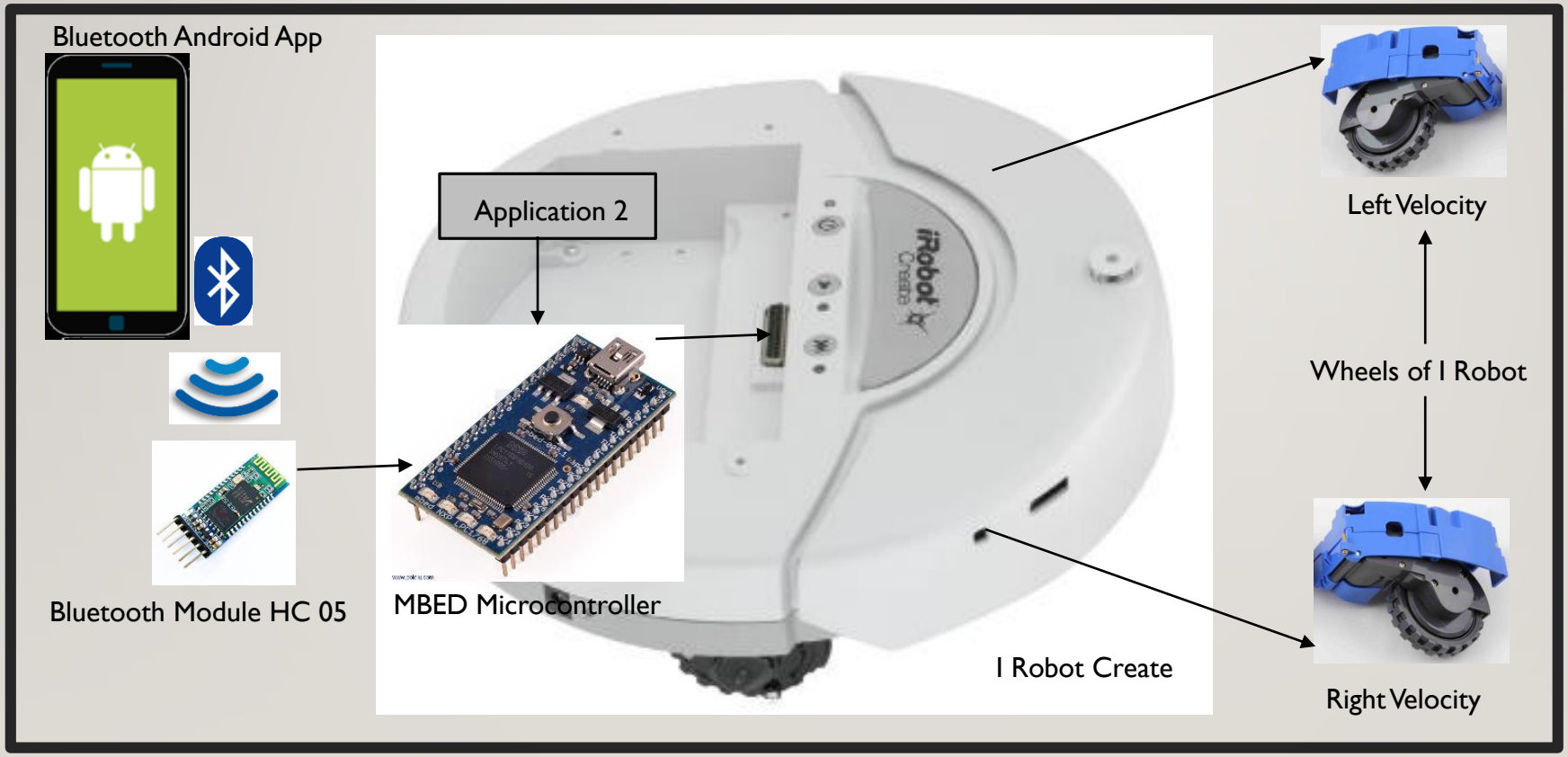
Hardware Setup





APPLICATION 2 : LOAD CARRYING ROBOT FOR AN INDUSTRY

a. System Layout



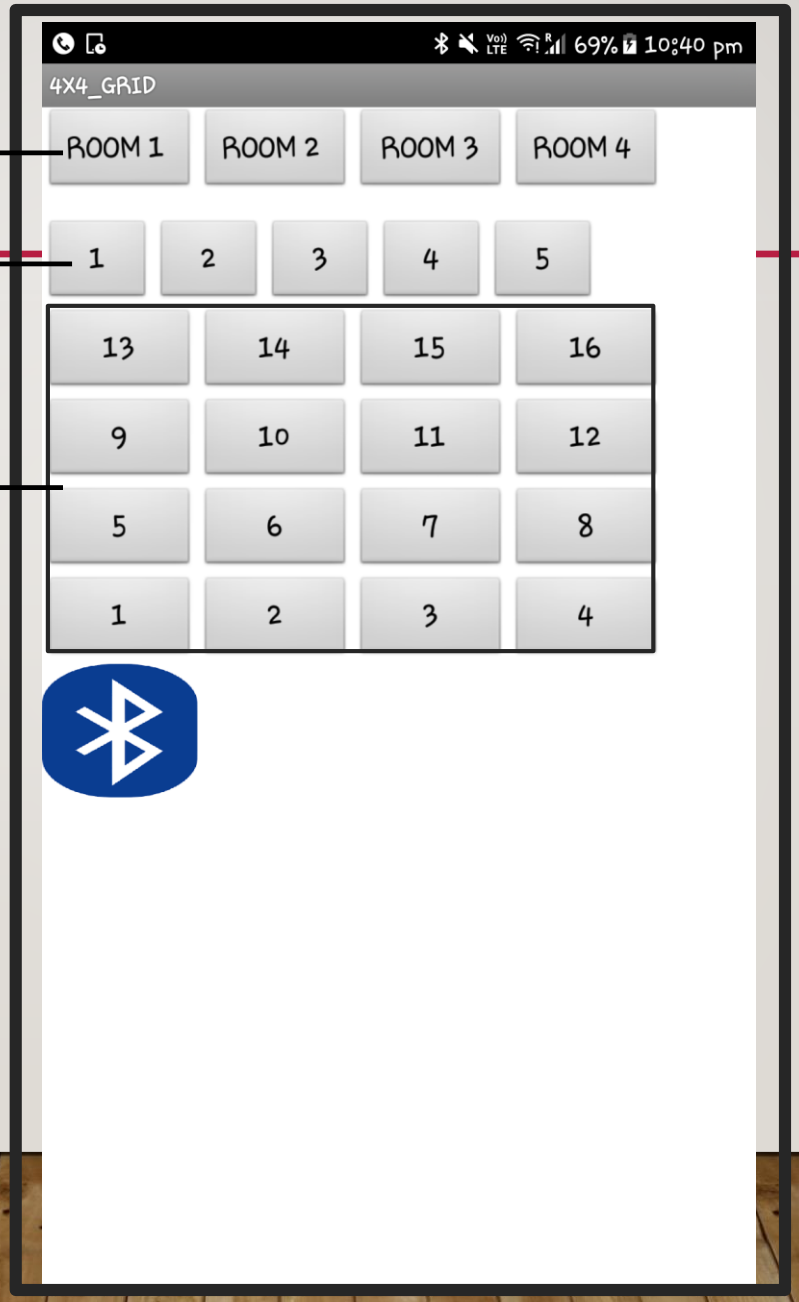
APPLICATION 2: LOAD CARRYING ROBOT IN AN INDUSTRY

Development of an app for Application2

Buttons to Select the rooms

Buttons to Select the number of obstacles

Buttons to Select the position of obstacles

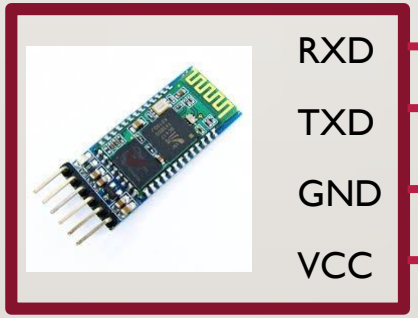


IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

System Layout

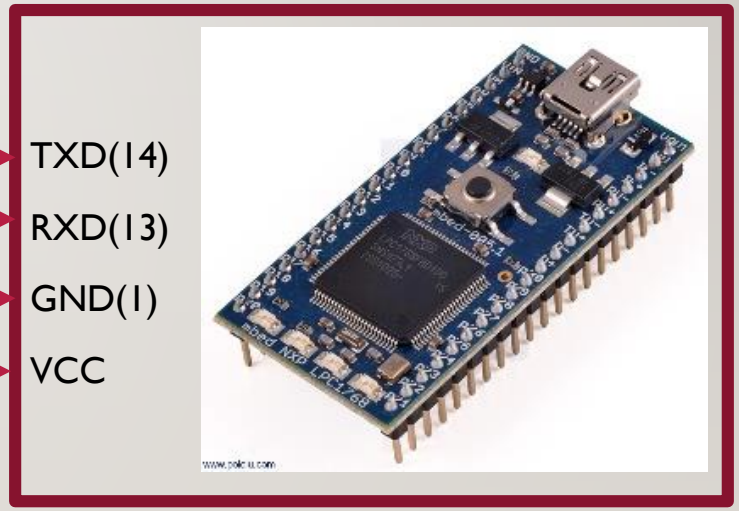
Bluetooth Android App

BLUETOOTH COMMUNICATION



Bluetooth Module HC 05

UART

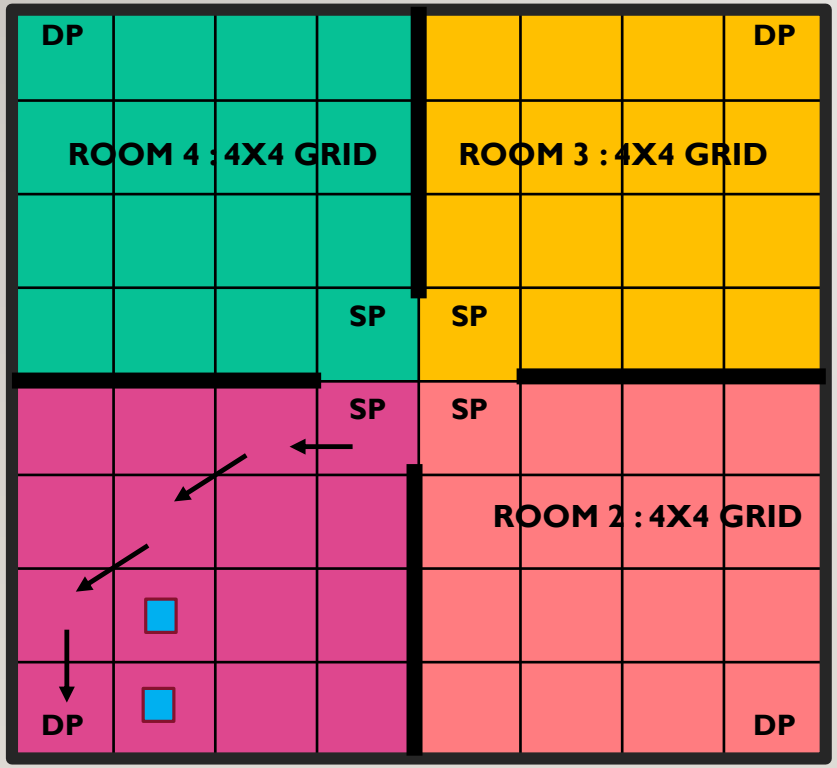


MBED Microcontroller

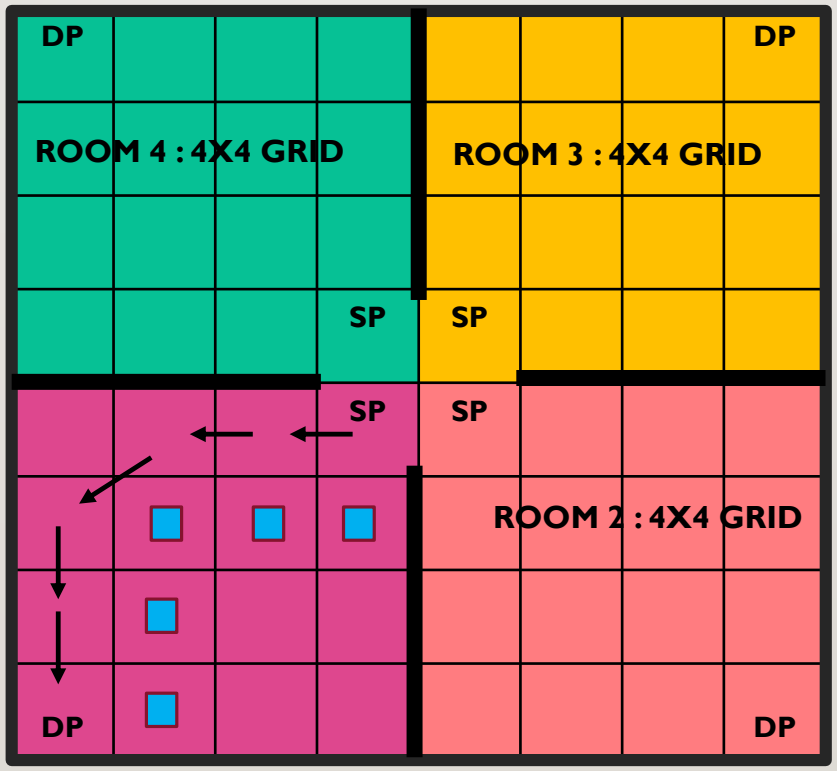
IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

Obstacle Occurrence scenarios tested in Room 1

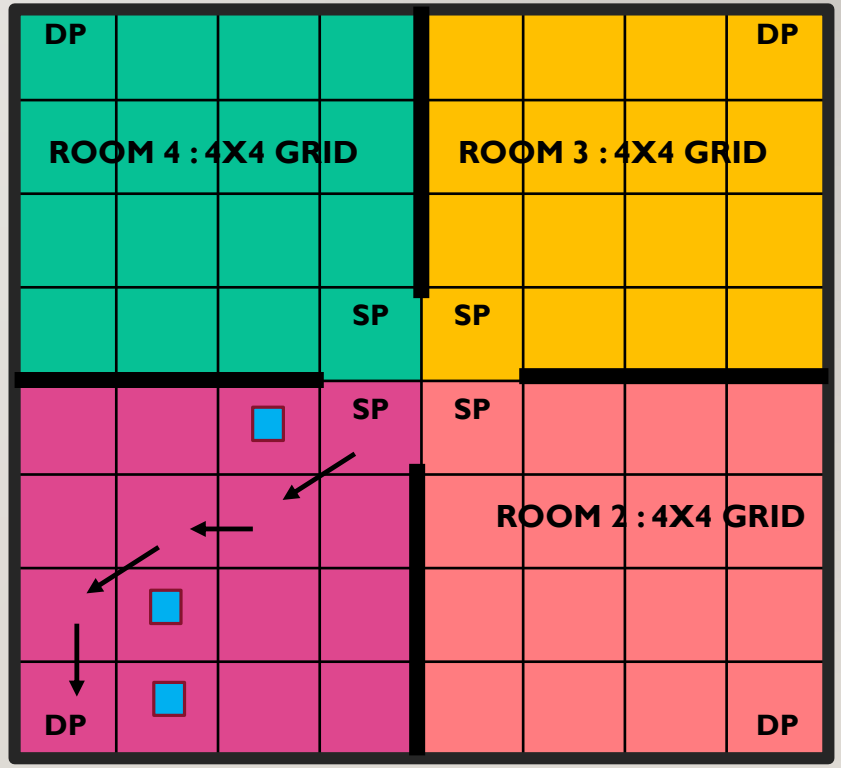
Path Planning with two obstacles



Path Planning with 5 obstacles



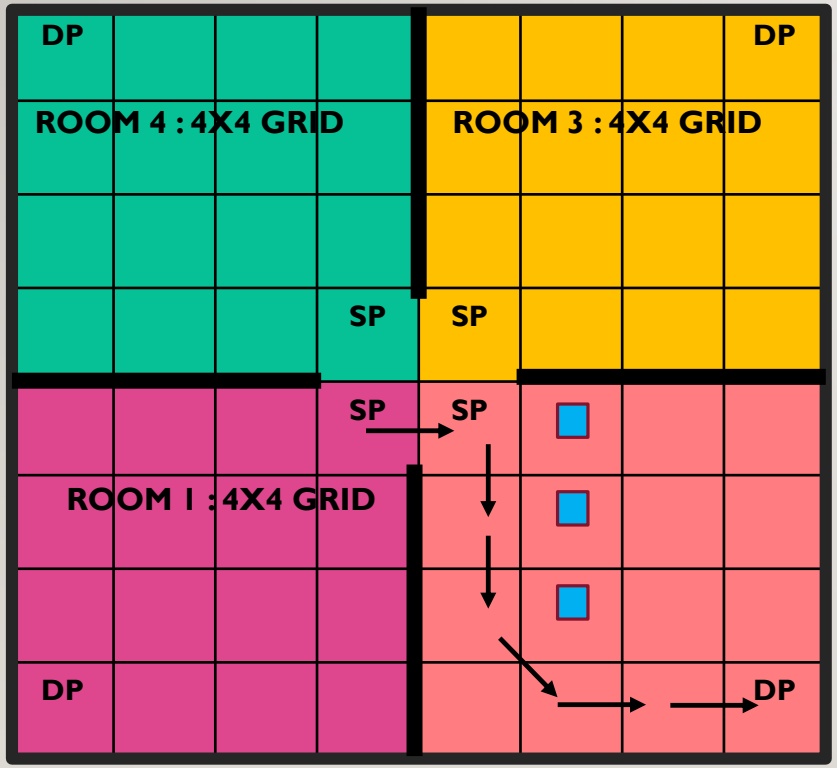
Path planning with 3 obstacles



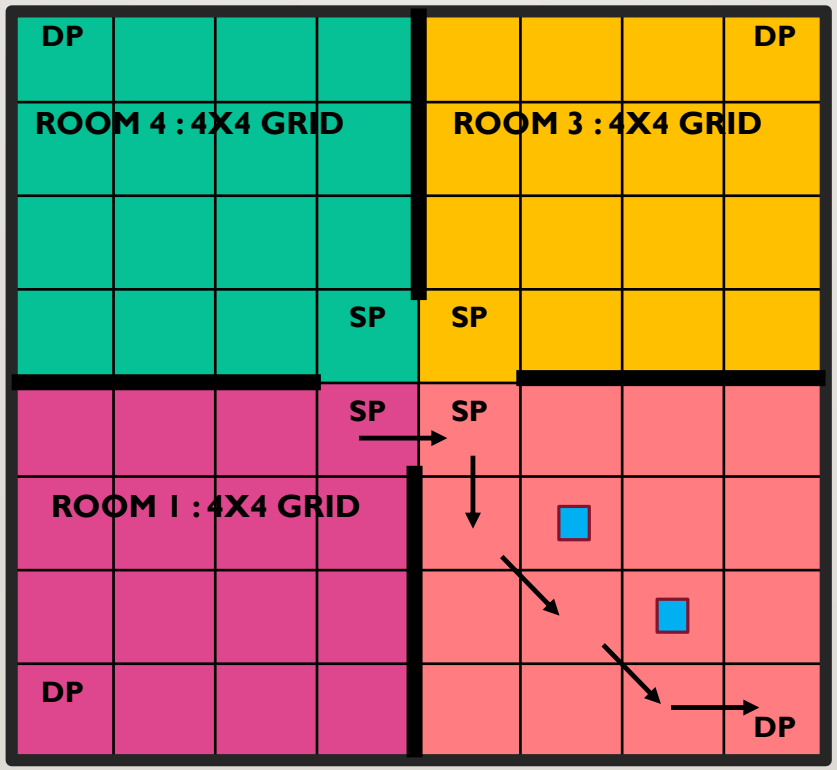
IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

Obstacle Occurrence scenarios tested in Room 2

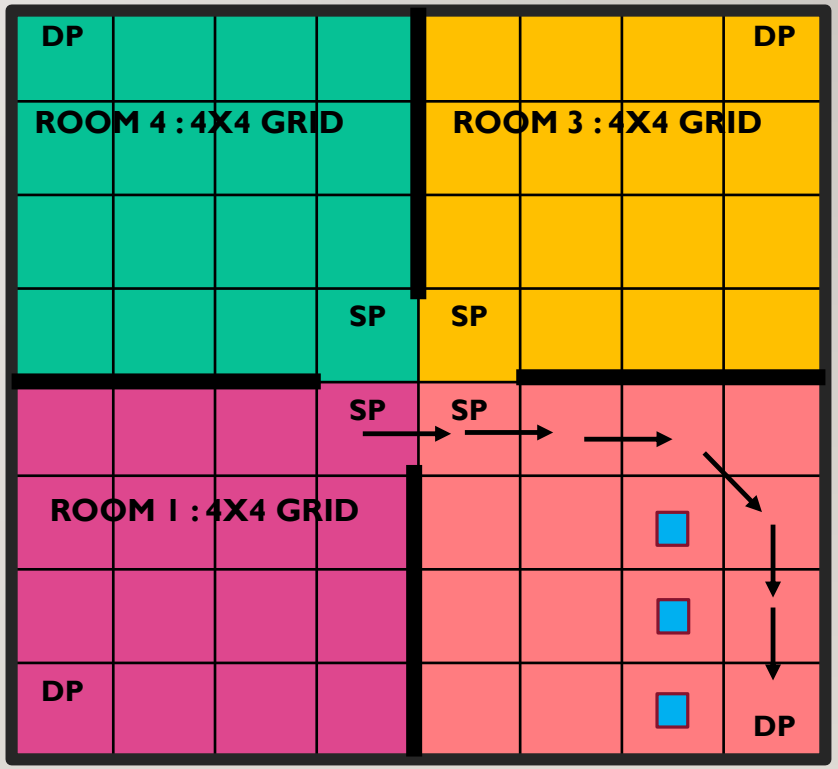
Path Planning with three obstacles



Path Planning with two obstacles



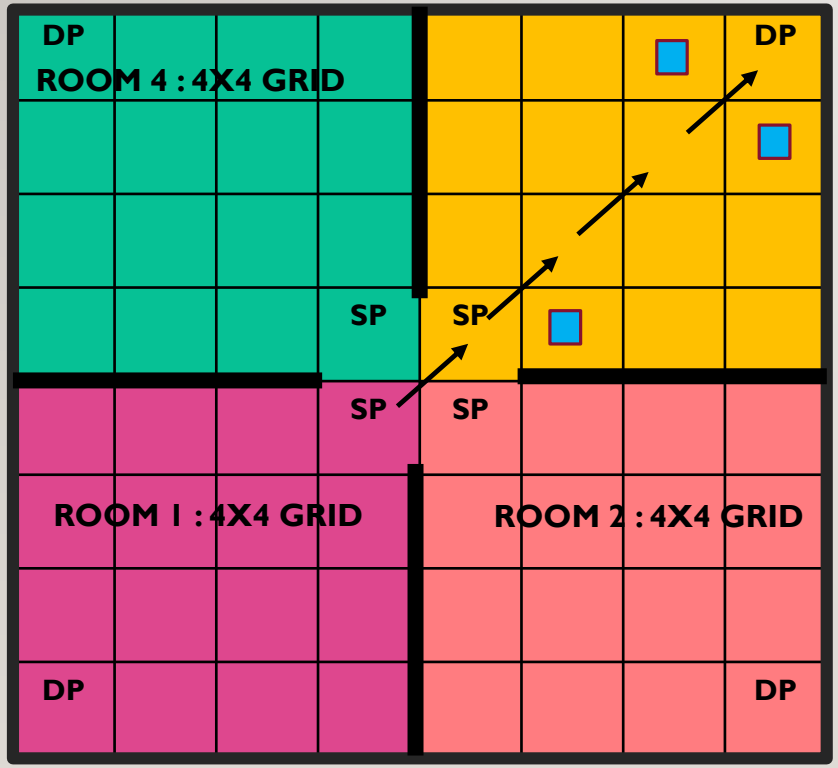
Path planning with 3 obstacles



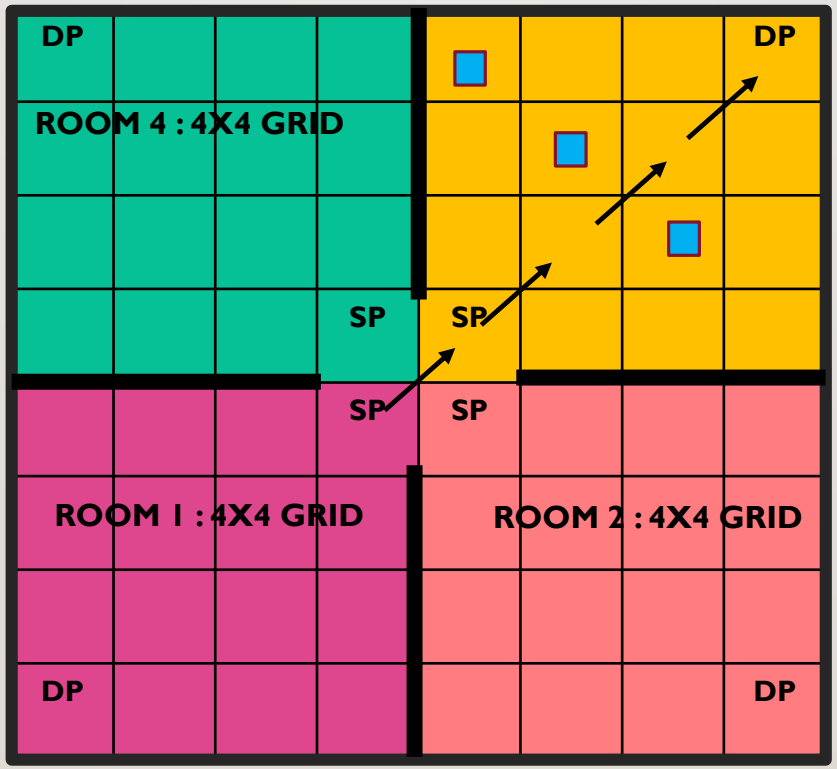
IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

Obstacle Occurrence scenarios tested in Room 3

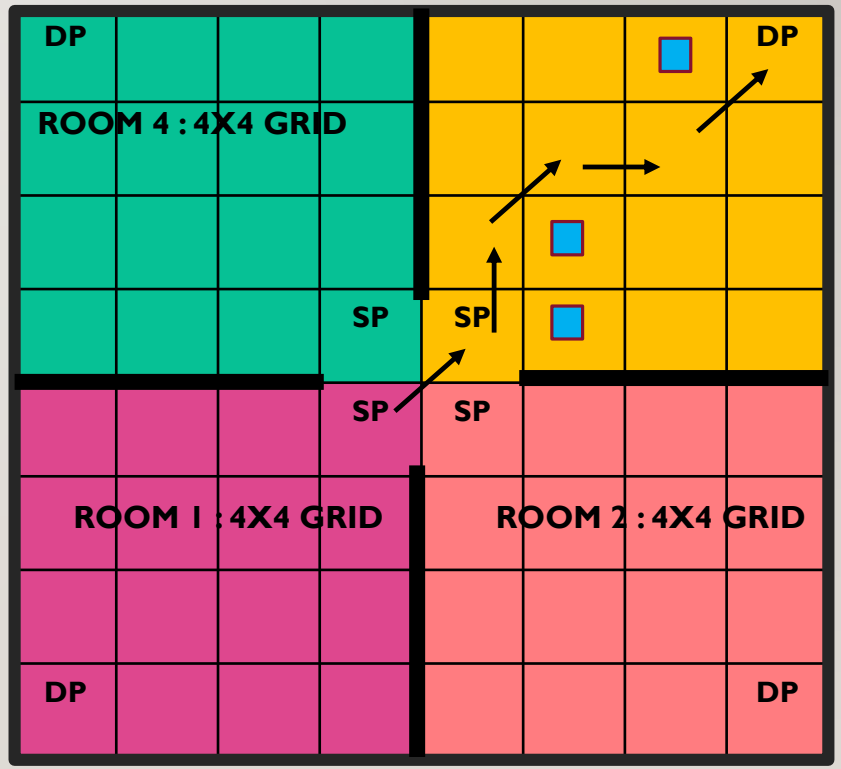
Path Planning with three obstacles



Path Planning with three obstacles



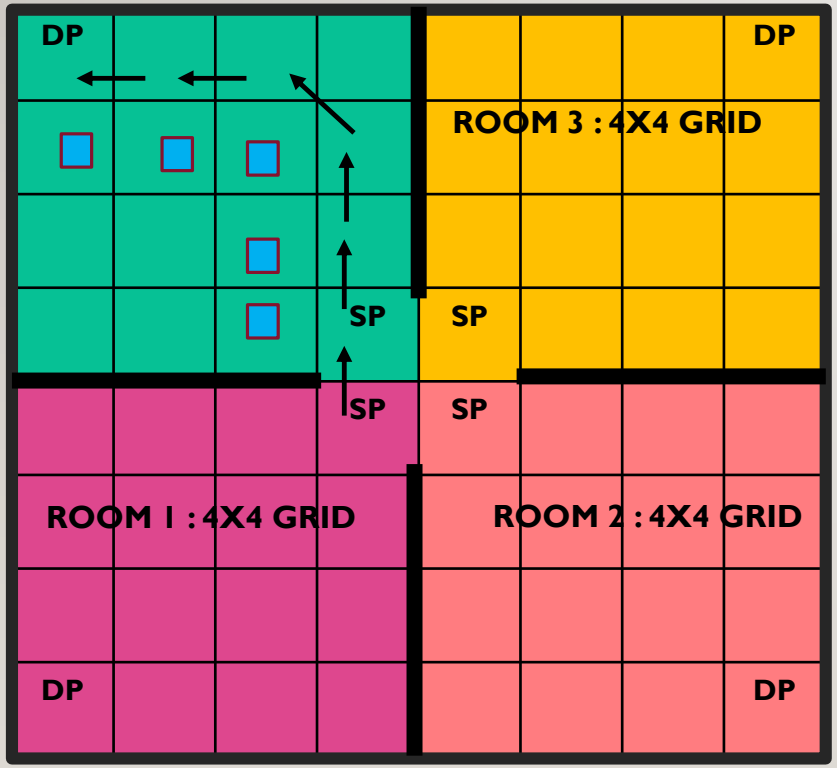
Path planning with 3 obstacles



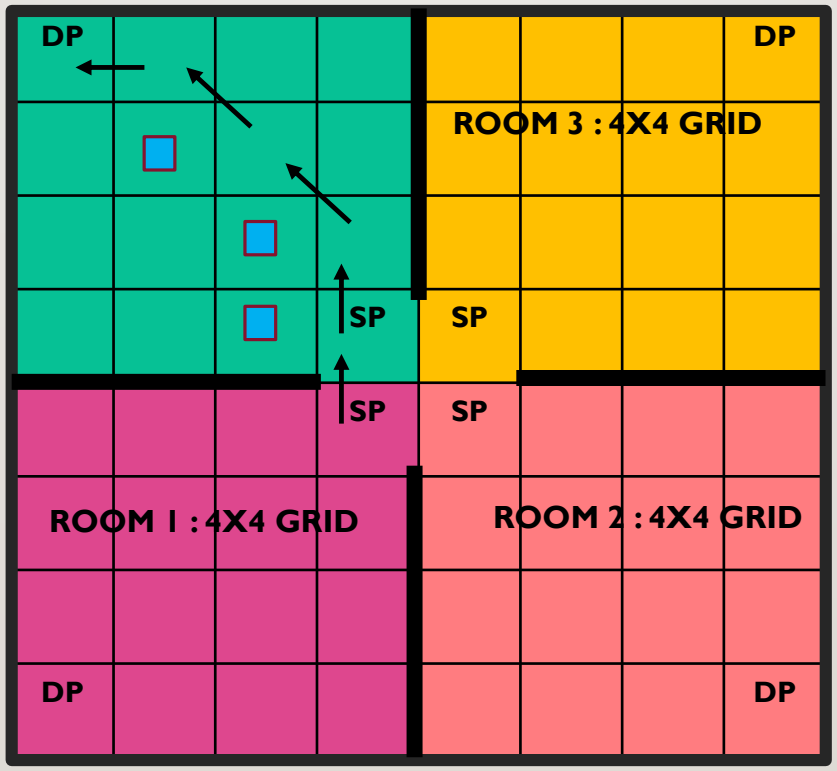
IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

Obstacle occurrence scenarios tested in Room 4

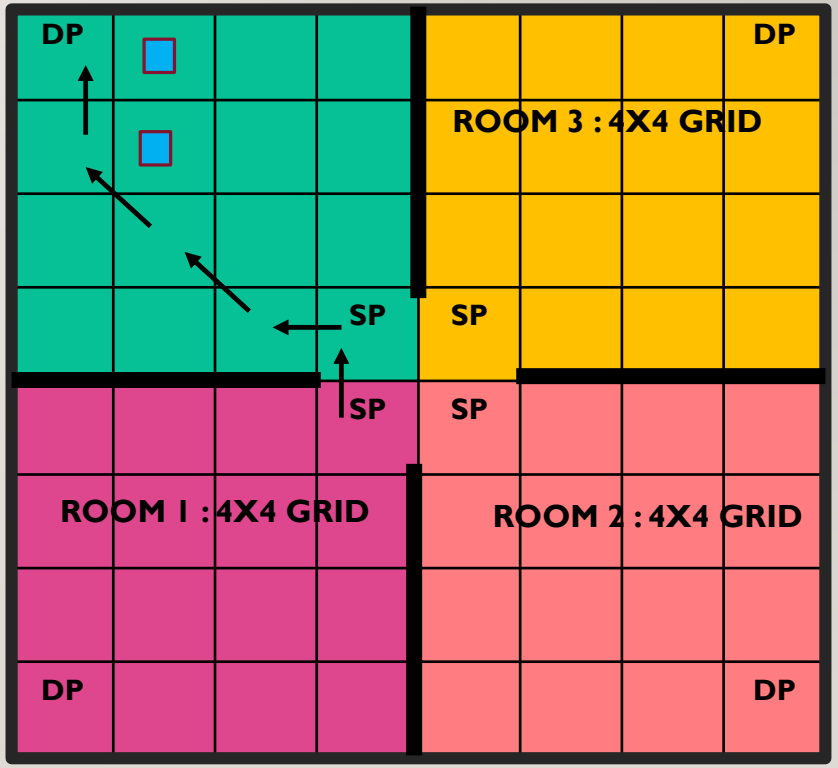
Path Planning with five obstacles



Path Planning with three obstacles

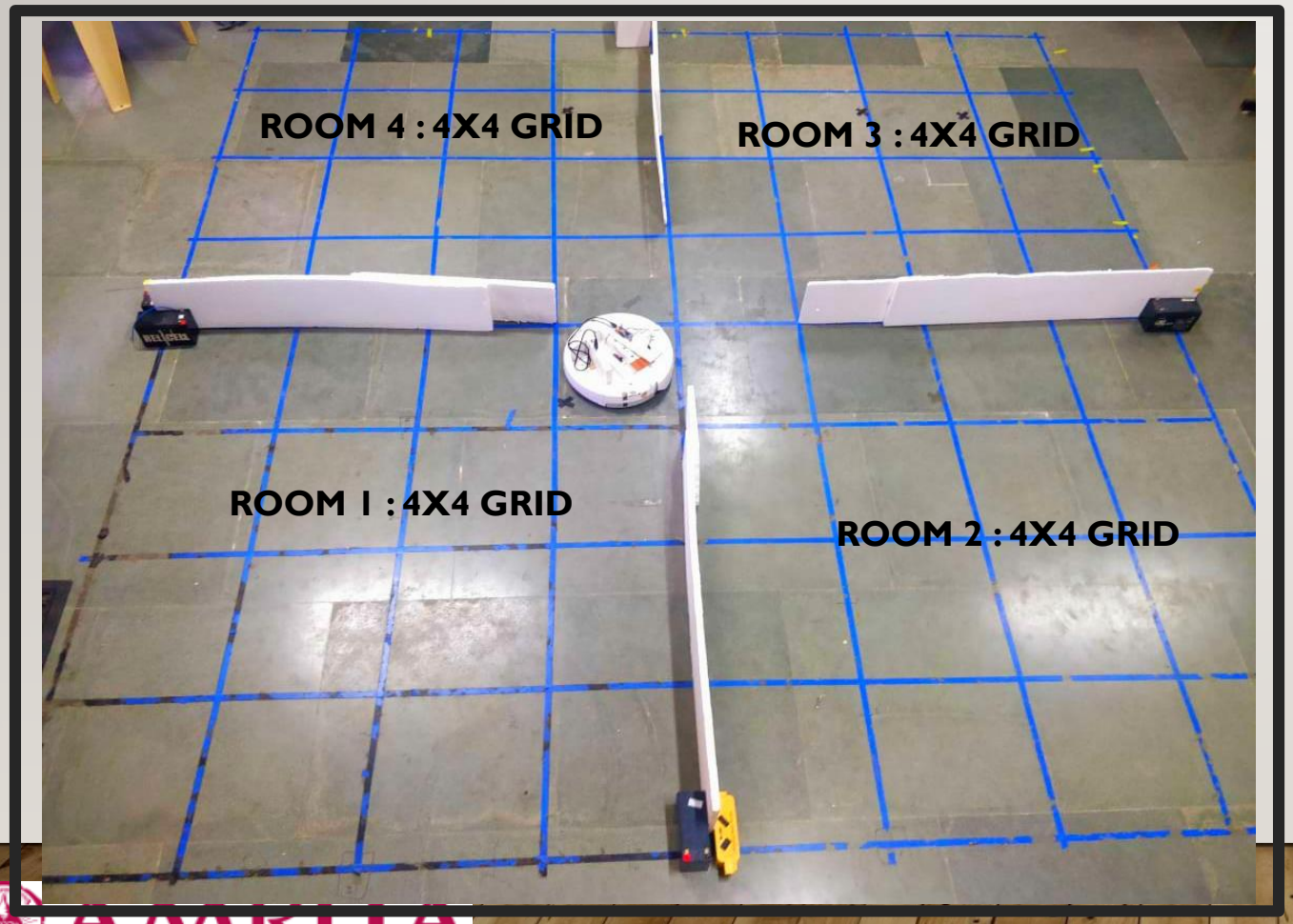


Path planning with two obstacles

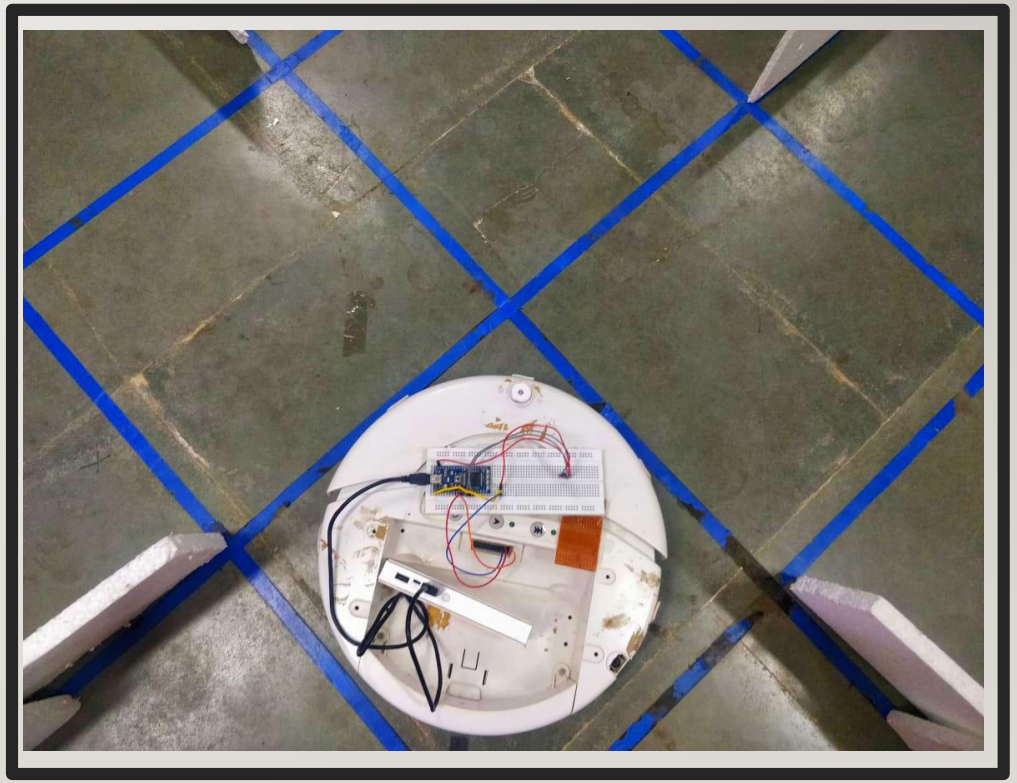


IMPLEMENTATION OF LOAD CARRYING ROBOT IN AN INDUSTRY

Pictures of hardware setup

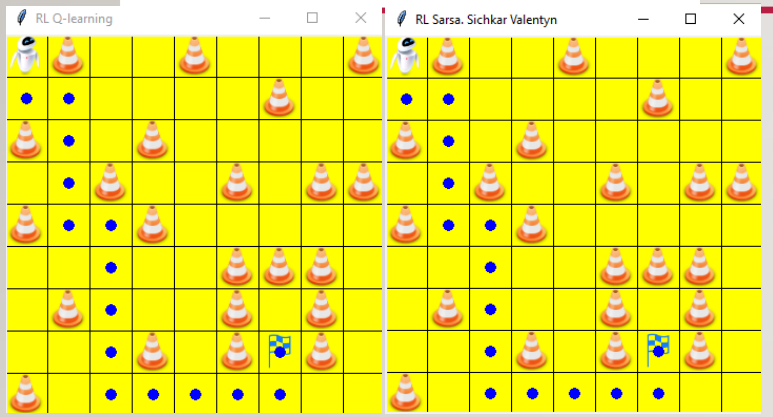


ORIENTATION OF IROBOT

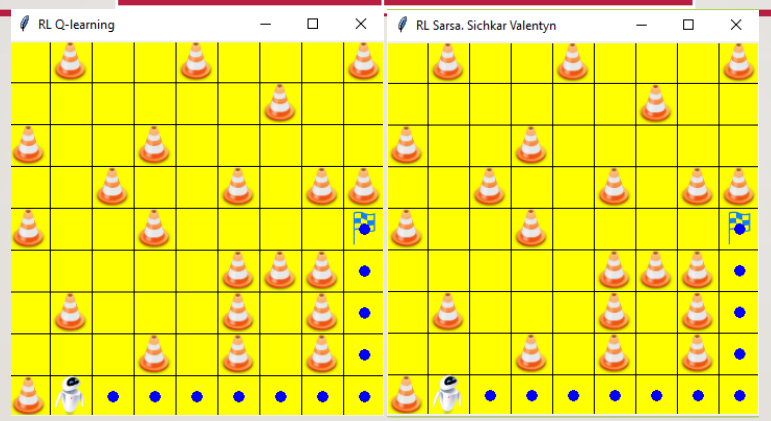


IMPLEMENTATION AND COMPARISON OF RL ALGORITHMS: SARSA AND Q LEARNING ALGORITHM

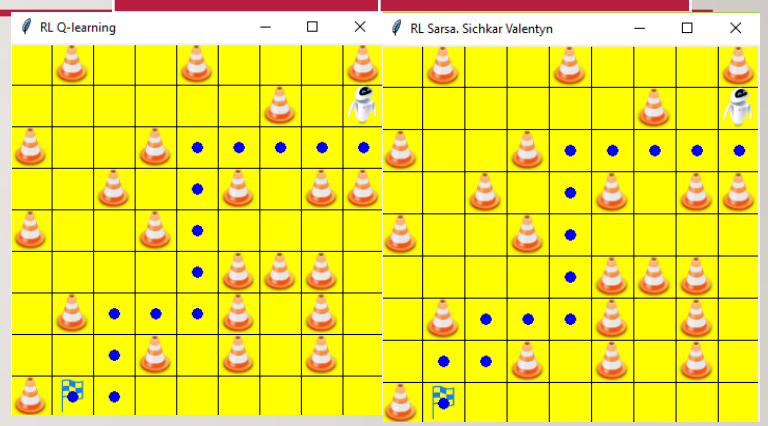
SP: (0,0) **DP: (6,7)**



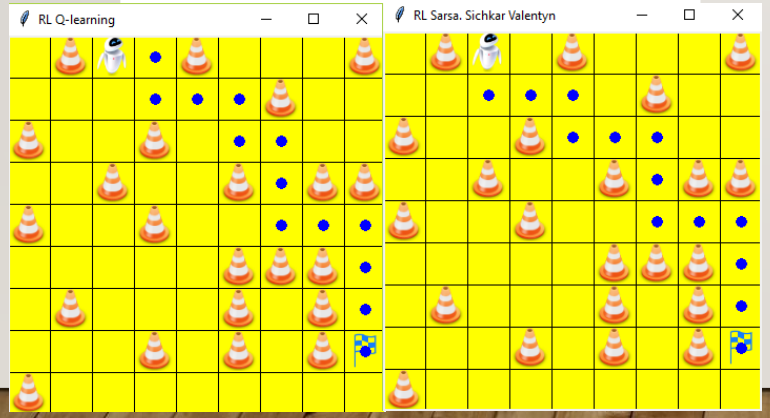
SP: (1,8) **DP: (8,4)**



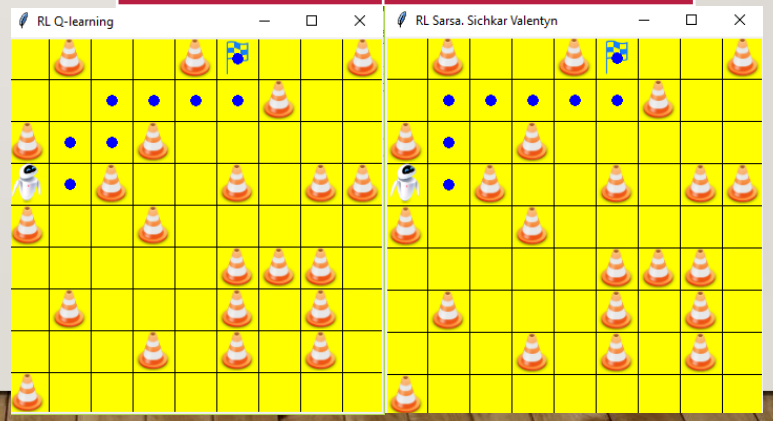
SP: (8,1) **DP: (1,8)**



SP: (2,0) **DP: (8,7)**



SP: (0,3) **DP: (5,0)**



VALUES OF THE CONSTANTS FOR SARSA AND Q LEARNING ALGORITHM

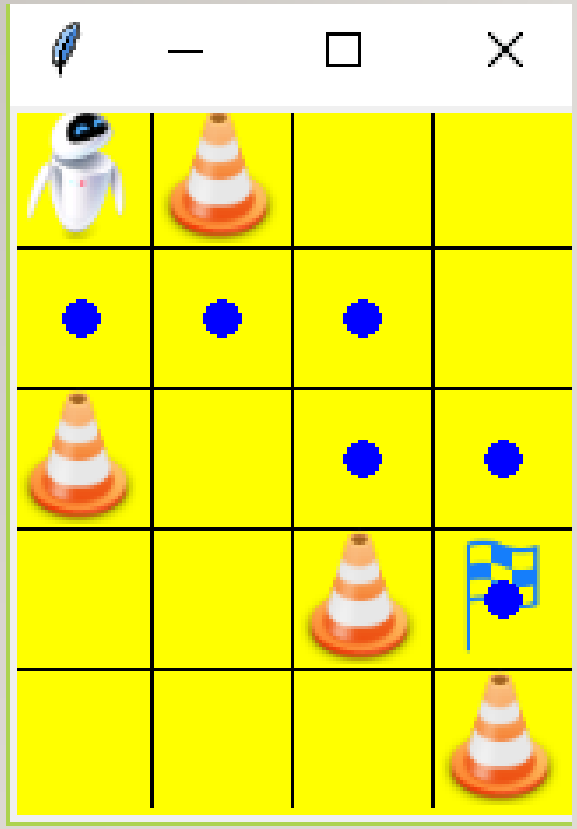
PARAMETERS	SARSA	Q LEARNING
No of Episodes	1000	1000
Learning Rate	0.01	0.01
Reward Decay	0.9	0.9

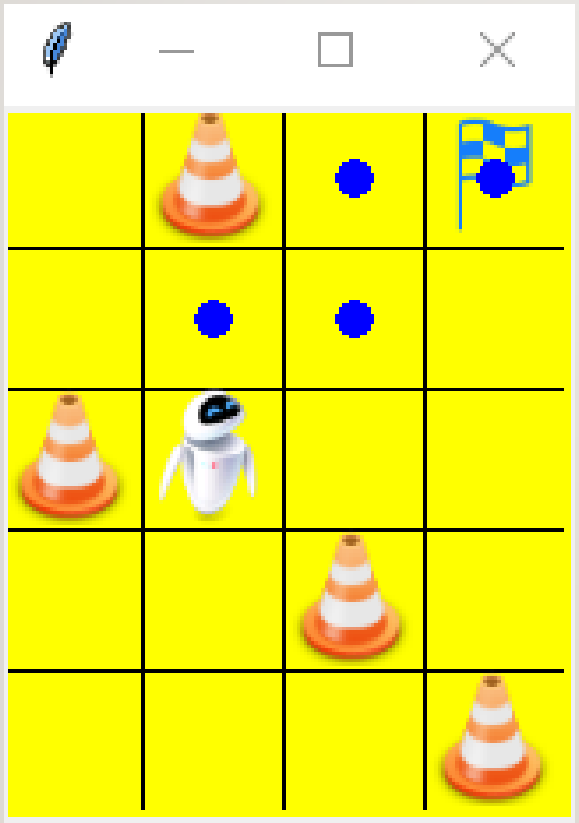
VALUES OF THE CONSTANTS FOR DEEP Q NETWORK ALGORITHM

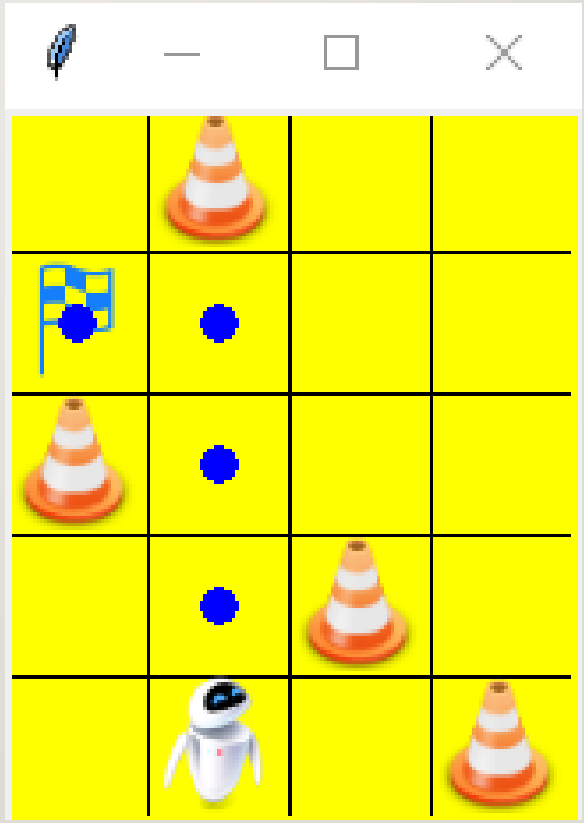
PARAMETERS	SARSA
No of Episodes	300
Learning Rate	0.01
Reward Decay	0.9
Memory Size	500

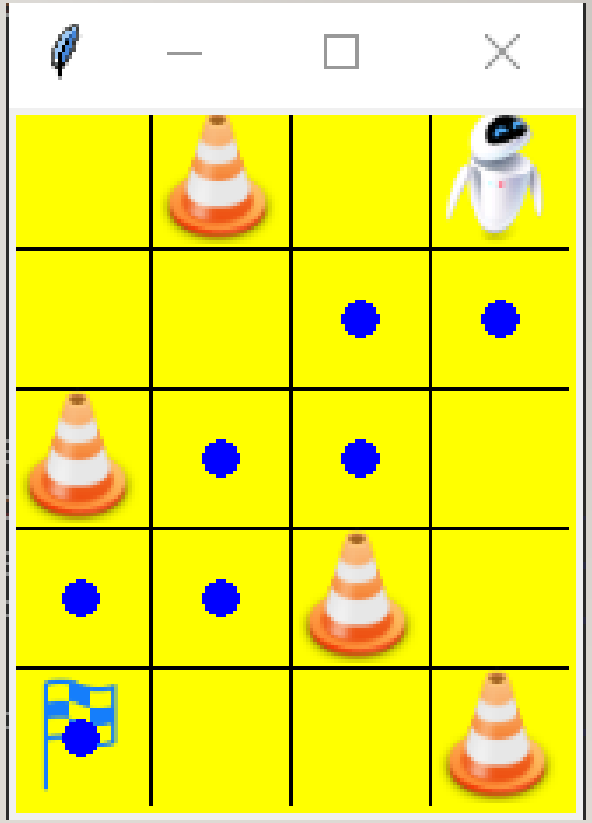
IMPLEMENTATION OF SARSA ALGORITHM FOR A RECTANGULAR GRID OF SIZE 4X5 IN PYTHON 3.7.3

a. Simulation Results for altering source and destination points









REINFORCEMENT LEARNING ALGORITHMS

TEMPORAL DIFFERENCE LEARNING ALGORITHM

- Implemented in MATLAB R2018b
- Implemented in Mbed microcontroller interfaced with iRobot Create

SARSA ALGORITHM

- Implemented in MATLAB R2018b
- Implemented in Mbed microcontroller interfaced with iRobot Create
- Implemented in Python IDE
- Implemented Robot to Robot communication using WIFI with SARSA algorithm
- Implemented a load carrying robot with room concept

Q LEARNING ALGORITHM

- Implemented in Python IDE

DEEP Q NETWORK ALGORITHM

- Implemented in Python IDE