Improvements in automatic ship collision avoidance using AI

Motivation

- Collision prevention is critical for navigation safety at sea
- In the early days, navigational assistance systems were there to help humans in ships tackle collision occurrence
- But autonomous vehicles have gained a remarkable amount of attention recently, focusing on solving collision problems by machines
- Most of previous works used A* for path planning in maritime environment but not for collision avoidance
- Here we worked on improving performance of A* to apply it for collision avoidance

Problem Statement

Automated navigation in congested waters is more challenging. In this project we tried to come up with a collision avoidance system in multi ship environment with certain assumptions.

- ➢ Own Ship
- ➢ 'N' Target Ships in the locality

Need to ensure our ship never collides with any of the target ships and reaches goal.

Image reference : https://www.marineinsight.com/wp-content/uploads/2015/04/traffic.jpg

Objective

Generate a geometric path by finding set of way points to navigate through from the start position to the end position

Decision made should ensure :

1. Optimal route

2. Safety of navigation

3. Less computational complexity

4. Smoothness

Proposed Methodology

- Formulating the problem as dynamic constraint based graph search problem.
- Trajectories are given as input to system. Chance of collision is computed based on the region of intersection of trajectories(safe boundary zone).
- Once the collision is detected the AI module performs heuristic based search guided by Fuzzy module to prune the search space(Fuzzy A* with dynamic heuristic).
- Output of AI module is the move(direction) to be taken from current position.
- Assumptions : speeds are constant and trajectories are certain.

Proposed Methodology

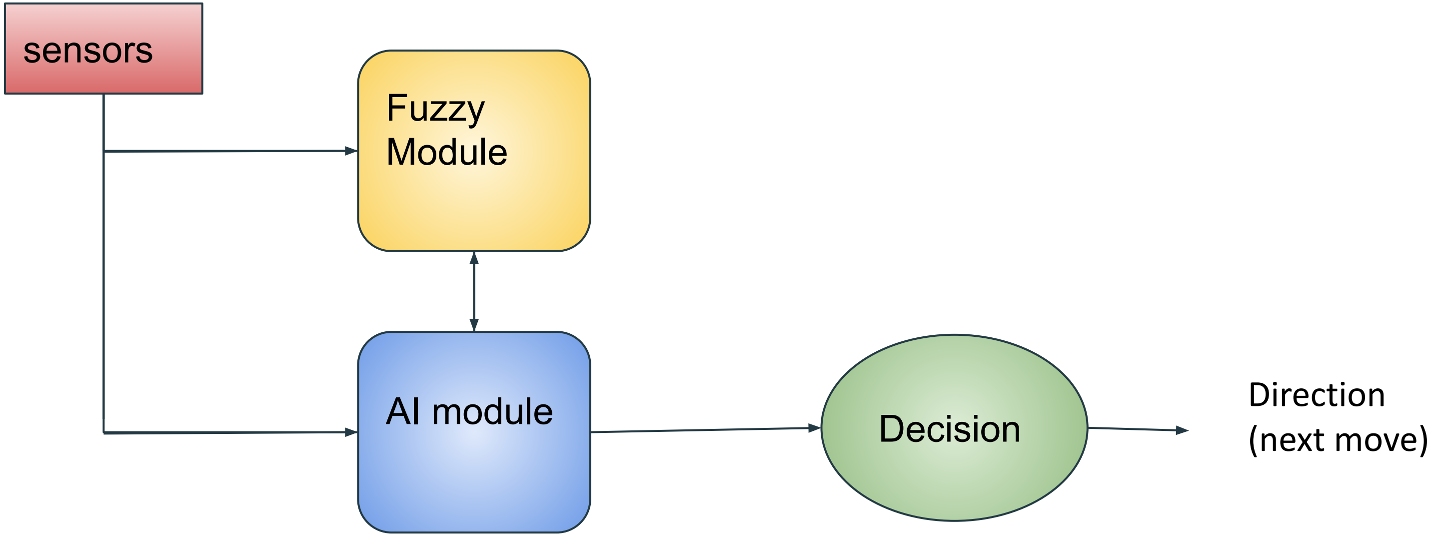


Fig: Working of the Algorithm. The sensor gives input to both the Fuzzy modules as well as AI module. Both modules interact with each other and come up with a direction for next move of the ship.

Fuzzy logic module

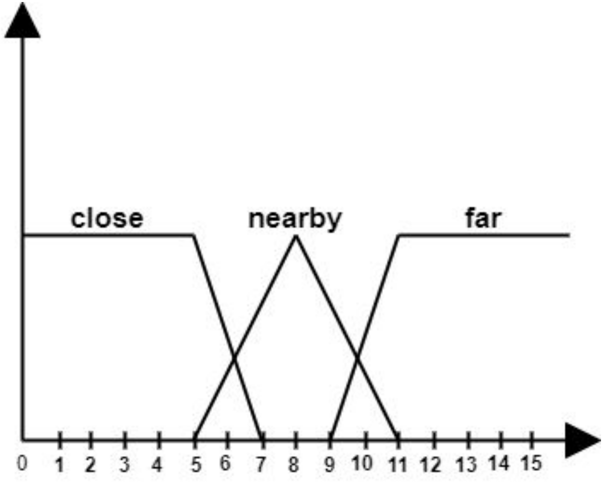


Fig: Distance to other ship from own ship

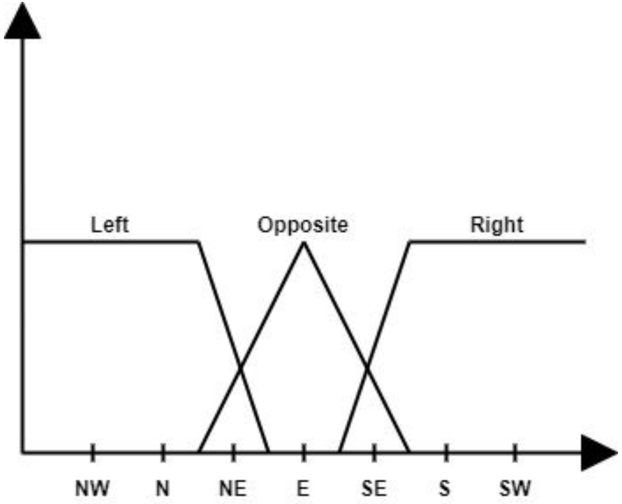


Fig: Direction of approach of other ship towards own ship

Fuzzy rules for direction of Movement

The following rules are applied in case a collision is detected.

- ➢ If other ship approaching from **left** direction and location is **close** then direction of ship is -**45** degree.
- ➢ If other ship approaching from **left** direction and location is **nearby** then direction of ship is **45** degree.
- ➢ If other ship approaching from **left** direction and location is **far away** then direction of ship is **0** degree.
- ➢ If other ship approaching from **right** direction and location is **close** then direction of ship is **45** degree.

Fuzzy rules for direction of Movement

- If other ship approaching from **right** direction and location is **nearby** then direction of ship is **-45** degree.
- If other ship approaching from **right** direction and location is **far away** then direction of ship is **0** degree.
- If other ship approaching from **opposite** direction and location is **close** then direction of ship is **45 or -45** degree(based on one which is close to destination).
- If other ship approaching from **opposite** direction and location is **nearby** then direction of ship is **-45** degree.
- If other ship approaching from **opposite** direction and location is **far away** then direction of ship is **0** degree.

Heuristic function

- Heuristic function is sum of manhattan distance from present location to destination and costs of adjacent moves
- The cost of adjacent moves will be updated based on the output(direction) of fuzzy module
- At every decision step, own ship will try to move close to destination at same time moving away from grids having more cost indicating chance of collision

Collision Avoidance Algorithm using A*

1. Generate random trajectories for other ships
2. Estimated trajectory of own ship using A*
3. Use trajectory of own ship and other ships to detect chance of collision by checking overlap in safe zones(region around ship with radius 1)
4. Costs of the cells in the zone of collision are updated using Fuzzy module
5. Find best next move using A*
6. Repeat step 2 until reached destination

Time and Space complexity

Time Complexity : O(N*b^d)

Space Complexity : O(b^d)

Here,

N - max number of steps taken by own ship to reach destination

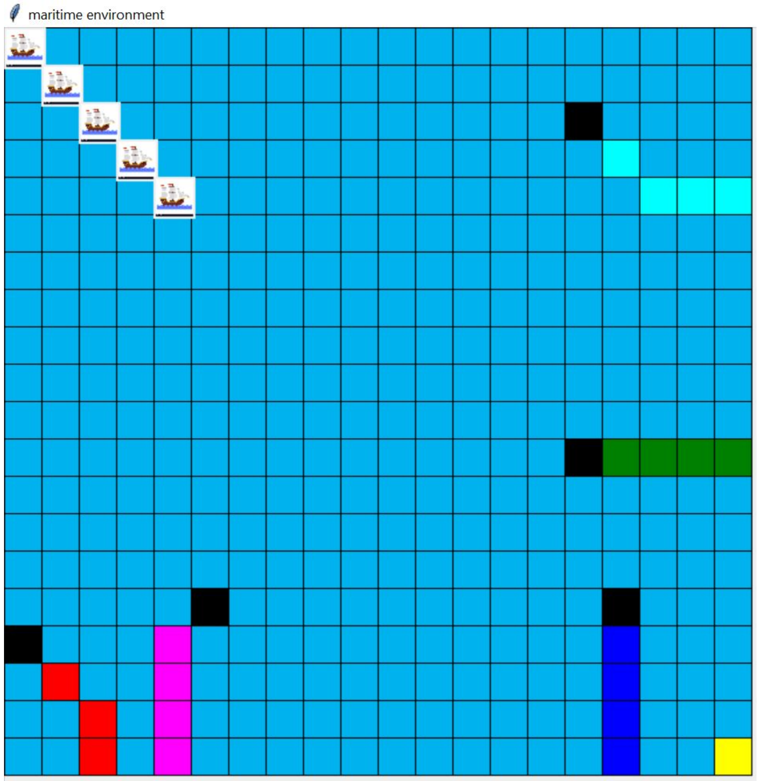
b - branching factor(here it is 3)

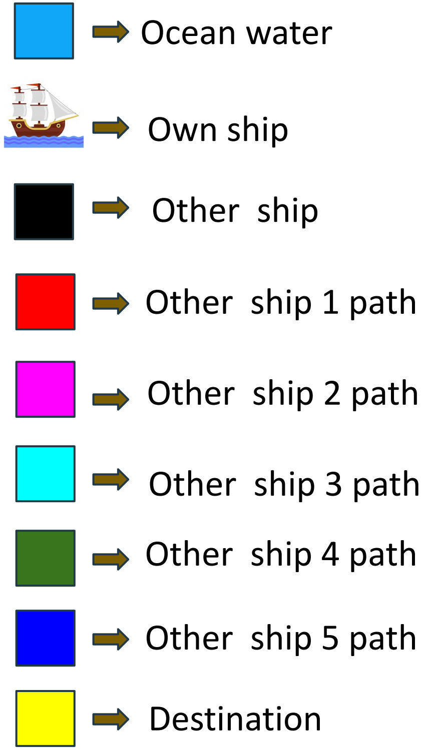
d - depth of search tree

Ship moves: break

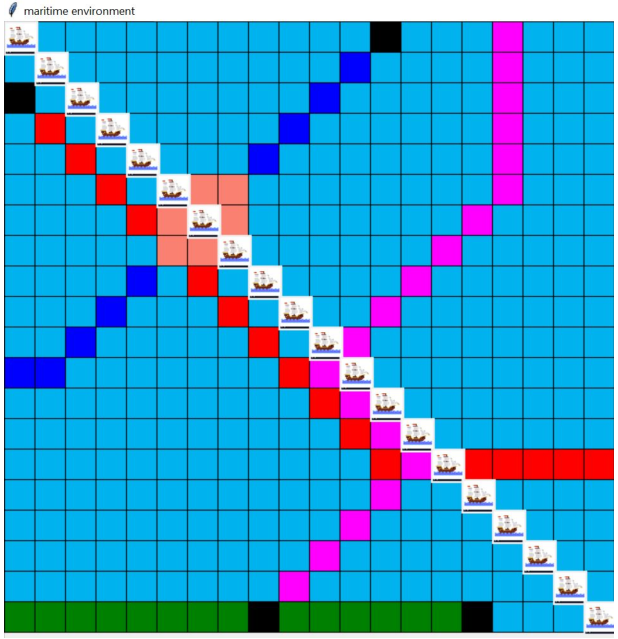
- Breaks were implemented for own ship
- In general, decision of the move will be taken by Fuzzy+AI module, but in extreme cases.
- This extreme case occurs when our ship has been deviated because of other ship(s) and if it moves along other ship direction.
- In water, there won’t be sudden breaks. Hence, the decision of applying break is made 2 grids earlier so as to stop at the correct location.

Ship water environment



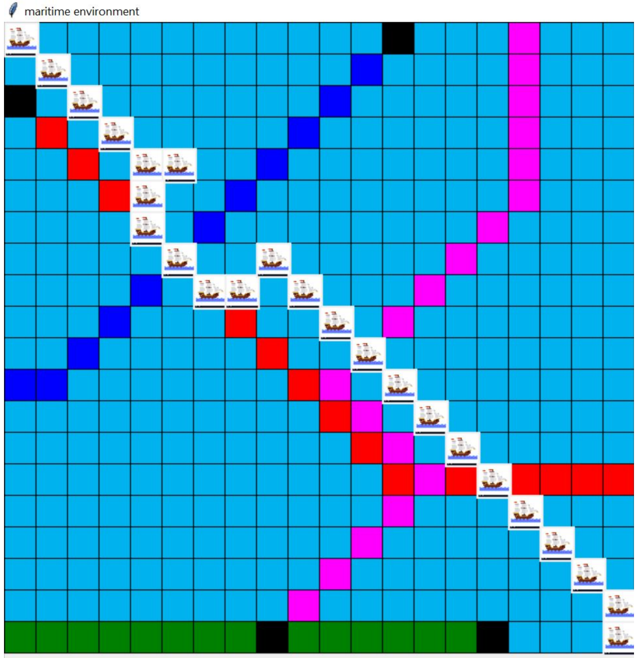


Collision Scenario



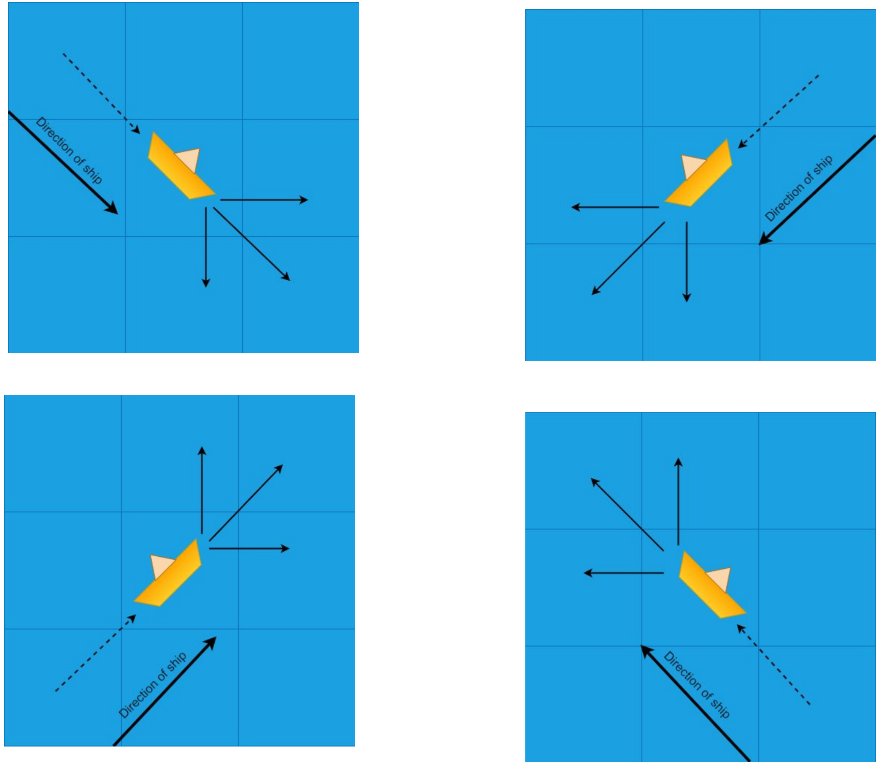
- ➢ This image is observation of using simple A* without any collision avoidance mechanism
- ➢ Own ship started at left corner and all other ships started at random positions
- ➢ Her own ship collides with other ship having path in blue colour, at time step 6
- ➢ The region surrounding the collision was indicated with **salmon** colour

Requirement of smoothing

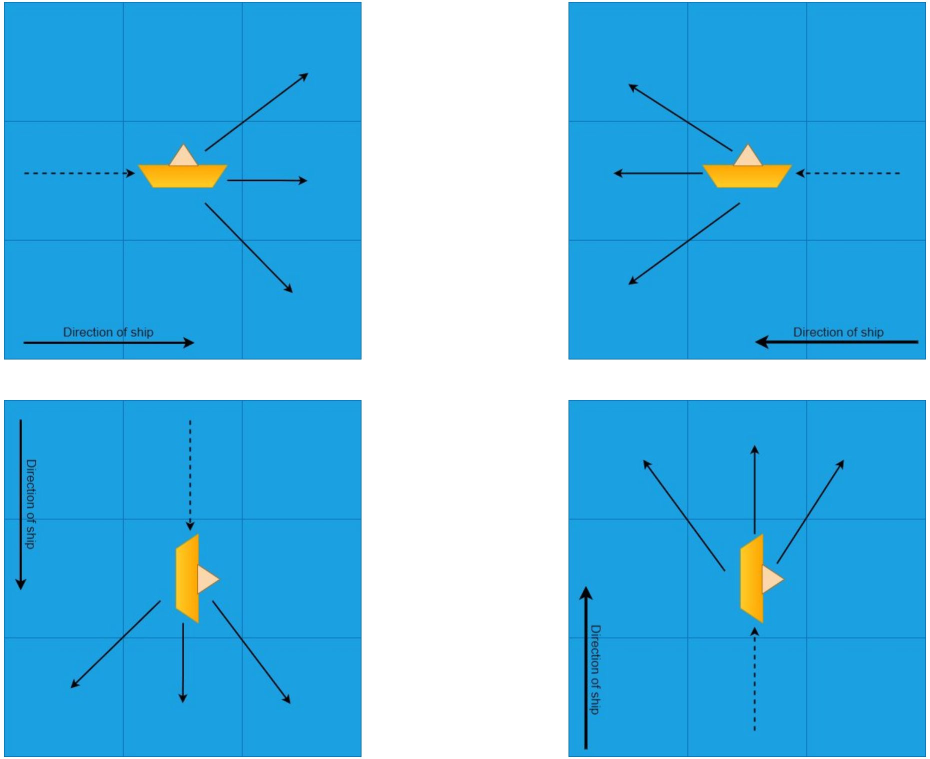


- This image shows the scenario of own ship trajectory using A* with collision avoidance
- Here in the path it is visible that some moves of ship are not reliable(Ex: suddenly moving back from present location, grid (4,5) to (4,4))
- So to address this problem smoothing was applied

Ship moves for smoothness



Ship moves for smoothness



Experiments

- **Board size** : 20x20 (can be any size nxn)
- **Number of other ships** : 4,5 (experimented with 4 and 5 other ships, can even have more)
- **Costs** : weighted costs based on the distance from other ships((1/d)*100)
- **Total steps of own ship** : Number of moves + breaks
- Here it is purely decision based algorithm so there won’t be any kind of training, decision will be made on the go
- Simulations were done for different random paths of other ships

Results

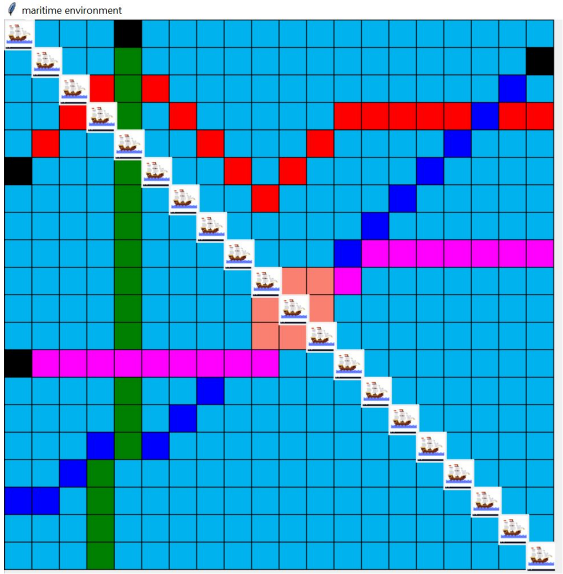


Fig: Collisions in path found by A*

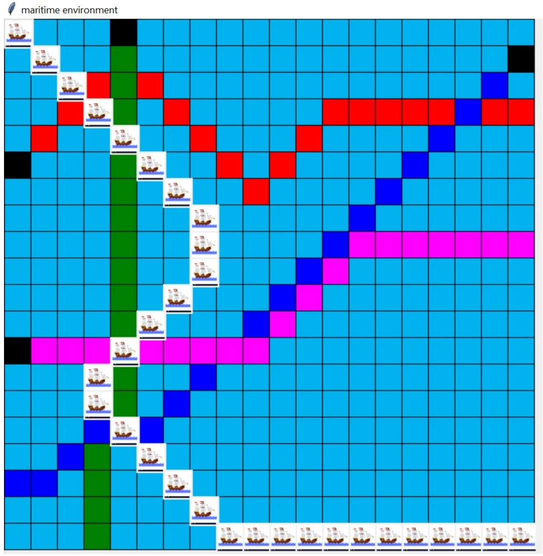


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 4

Steps: 30

Time: 0.027s

Breaks: 0

Moves: 30

Results

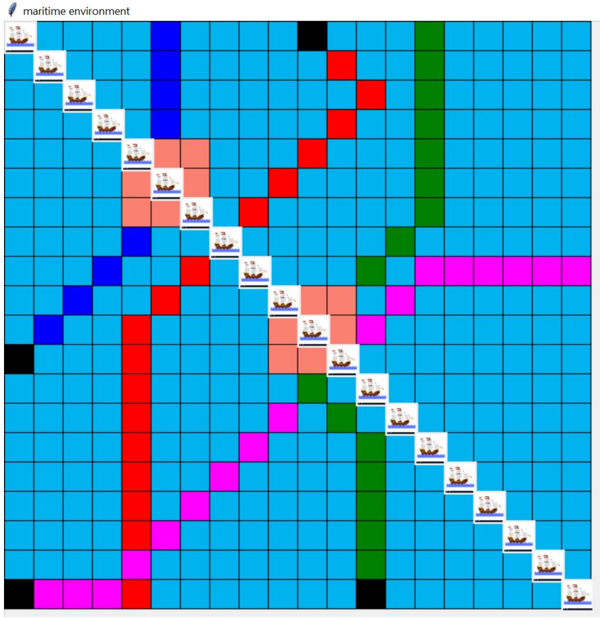


Fig: Collisions in path found by A*

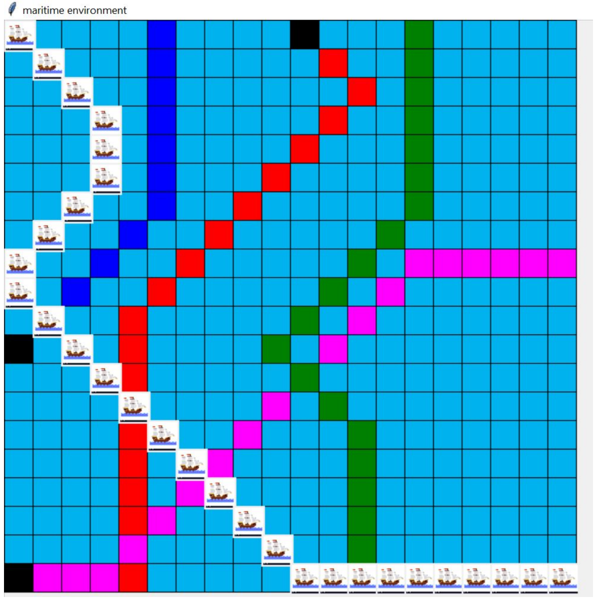


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 4

Steps: 31

Time: 0.029s

Breaks: 2

Moves: 29

Results

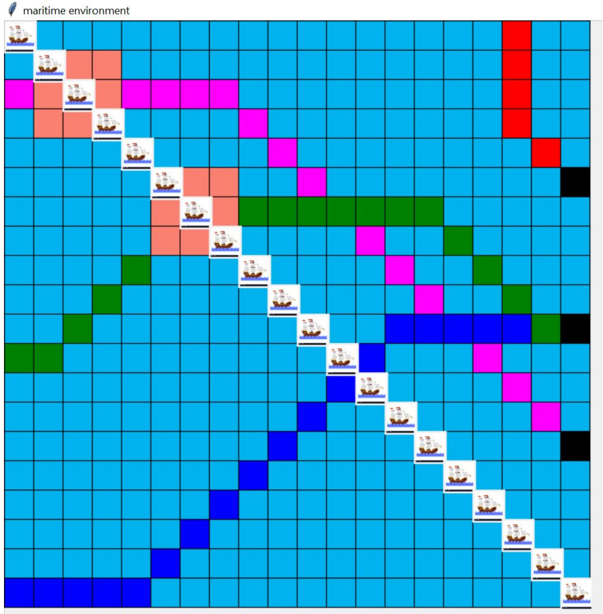


Fig: Collisions in path found by A*

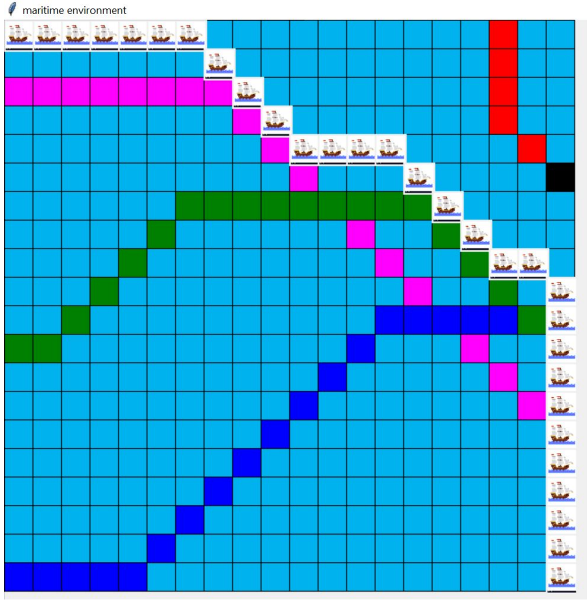


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 4

steps: 31

Time: 0.050s

Breaks : 2

moves : 29

Results

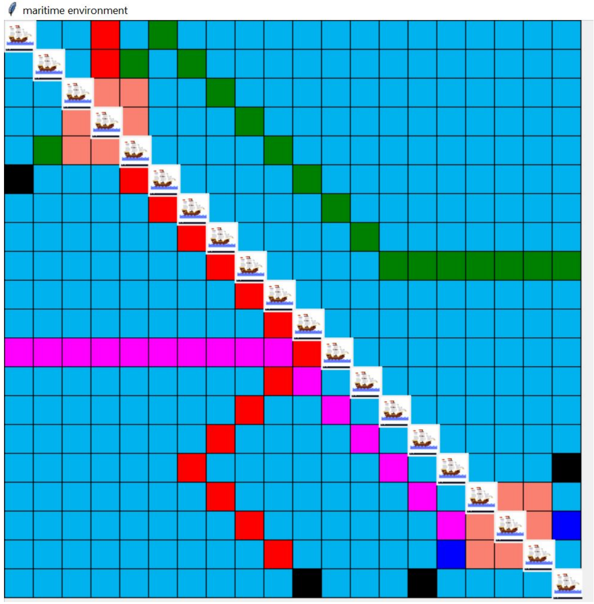


Fig: Collisions in path found by A*

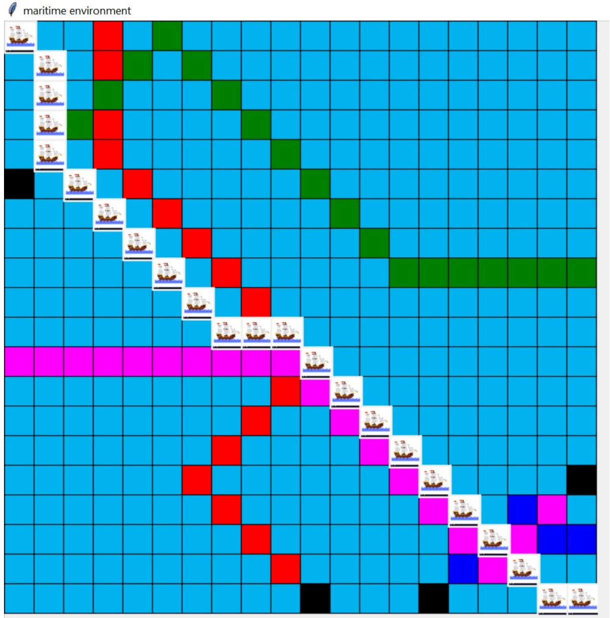


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 4

Cost : 23

Time: 0.025s

Breaks : 1

Moves: 22

Results

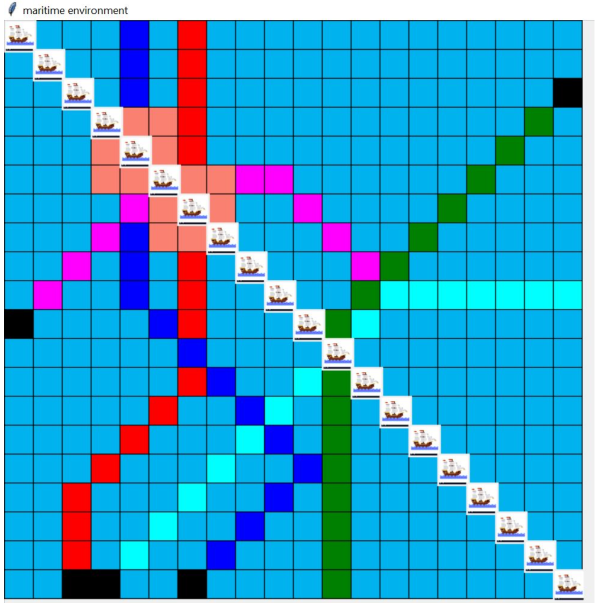


Fig: Collisions in path found by A*

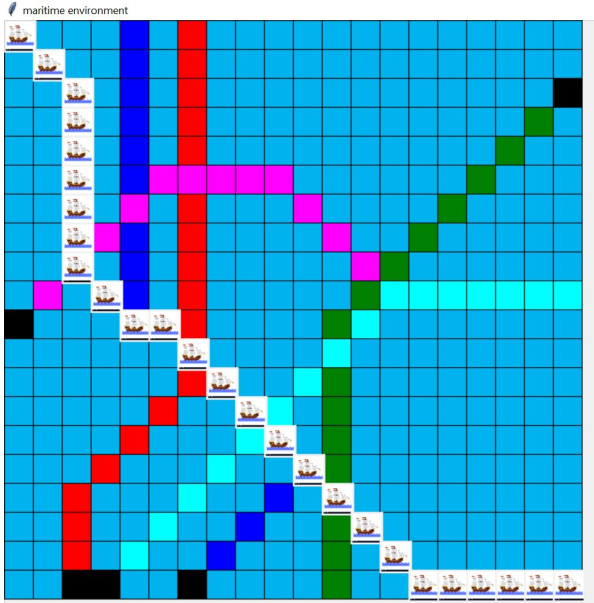


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 5

Steps: 27

Time: 0.037s

Breaks : 2

Moves: 25

Results

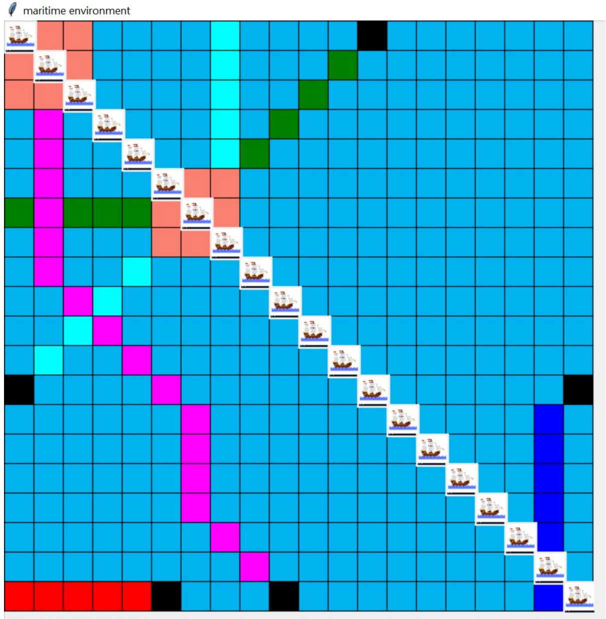


Fig: Collisions in path found by A*

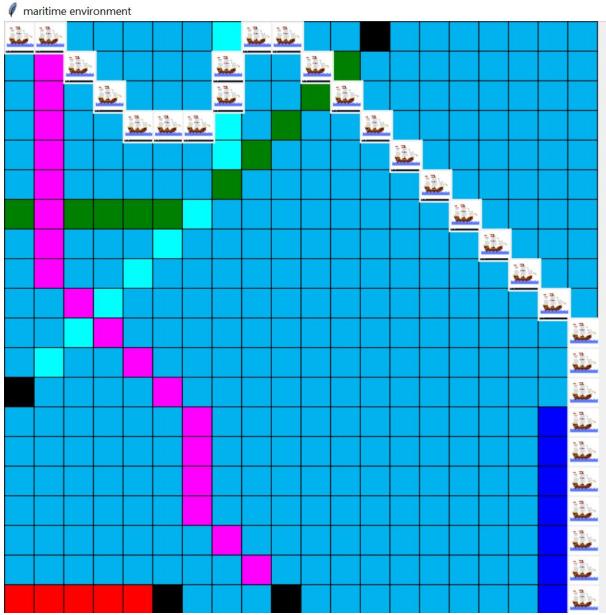


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 5

Steps: 31

Time: 0.038s

Breaks : 2

Moves: 29

Results

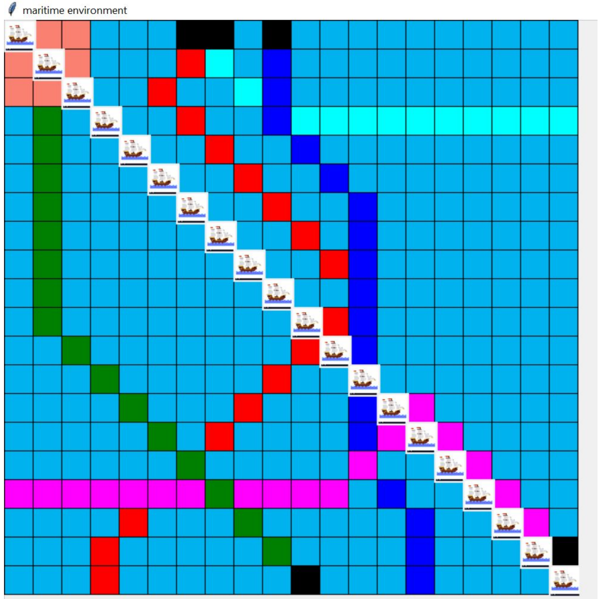


Fig: Collisions in path found by A*

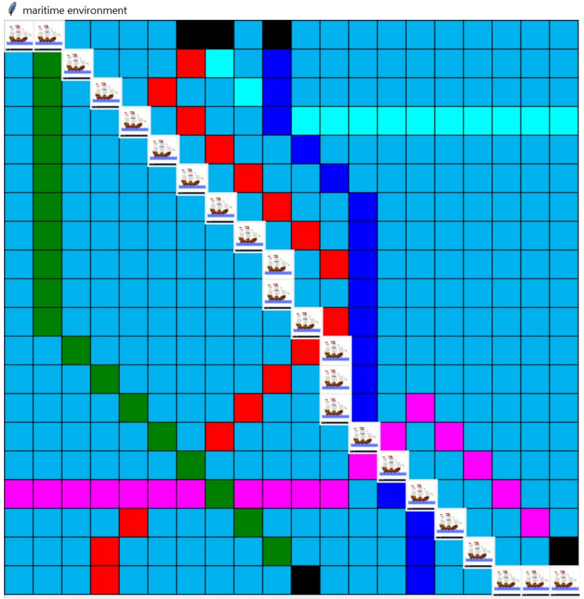


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 5

Steps: 24

Time: 0.080s

Breaks : 2

Moves: 22

Results

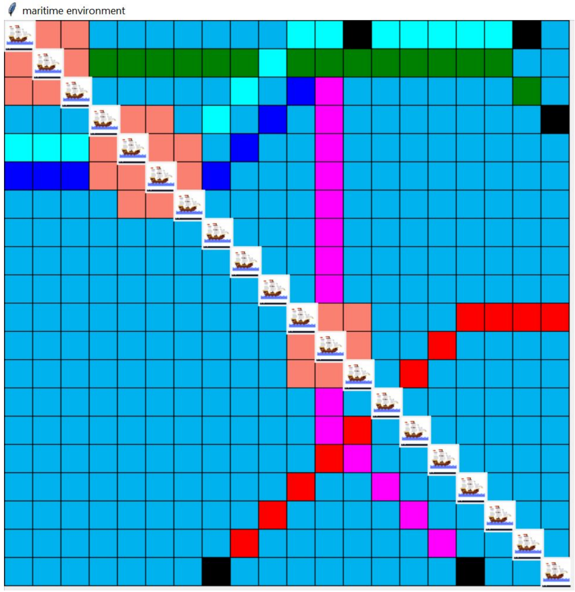


Fig: Collisions in path found by A*

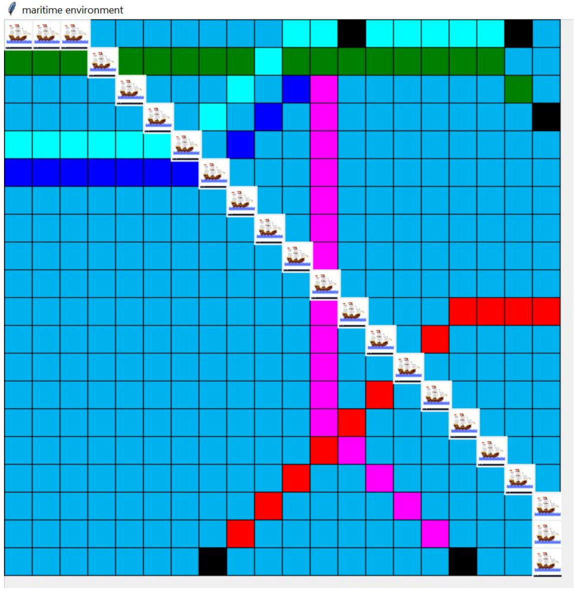


Fig: Trajectory using collision avoidance algorithm with smoothing

Other Ships: 5

Steps: 22

Time: 0.022s

Breaks : 1

Moves: 21

Conclusion and Future Work

- ➢ A* with dynamic heuristic function with help of fuzzy logic was implemented for ship collision avoidance in congested water environment
- ➢ Paths of Own ship computed from ship collision avoidance algorithm from different experiments has avoided collision, ensured smoothness and optimality
- ➢ With smoothness constraint sometimes optimality has to be compromised
- ➢ Future work can be done by considering more uncertainties and including speed factor

References

- Calle, Miguel Angel & Alves, Marcilio. (2011). Ship collision: A brief survey.
- Chen L., Negenborn R.R., Lodewijks G. (2016) Path Planning for Autonomous Inland Vessels Using A*BG. In: Paias A., Ruthmair M., Voß S. (eds) Computational Logistics. ICCL 2016.
- Vagale, A., Oucheikh, R., Bye, R.T. et al. Path planning and collision avoidance for autonomous surface vehicles I: a review. J Mar Sci Technol (2021)

Tools Used

- **For GUI** : tkinter, termcolor, PIL
- **Programming Language** : Python

Thank you