Dynamic Task Allocation Using Multi-Agent Mobile Robots

Raghavv Goel[1], Sha Yi[2], Jaskaran Singh Grover[2], Katia Sycara[2]

[1] IIIT Delhi, [2] Carnegie Mellon University

Introduction

- Multi-agent systems are useful in surveillance, search and rescue, monitoring of crops etc.
- Above real world tasks are dynamic: constantly changing and uncertain and we hence prefer using multiple agents as they provide provide wide-area coverage, fault tolerance, and robustness to missions.
- We address this issue of covering all the tasks present in the environment by proposing an algorithm which solves an optimization problem to assign agents to different tasks after every time step.
- Tasks are in the form of rooms as shown below and as the agents only discover rooms when they are in the doorways of the rooms, the proposed optimization is solved at every time step.

Environment

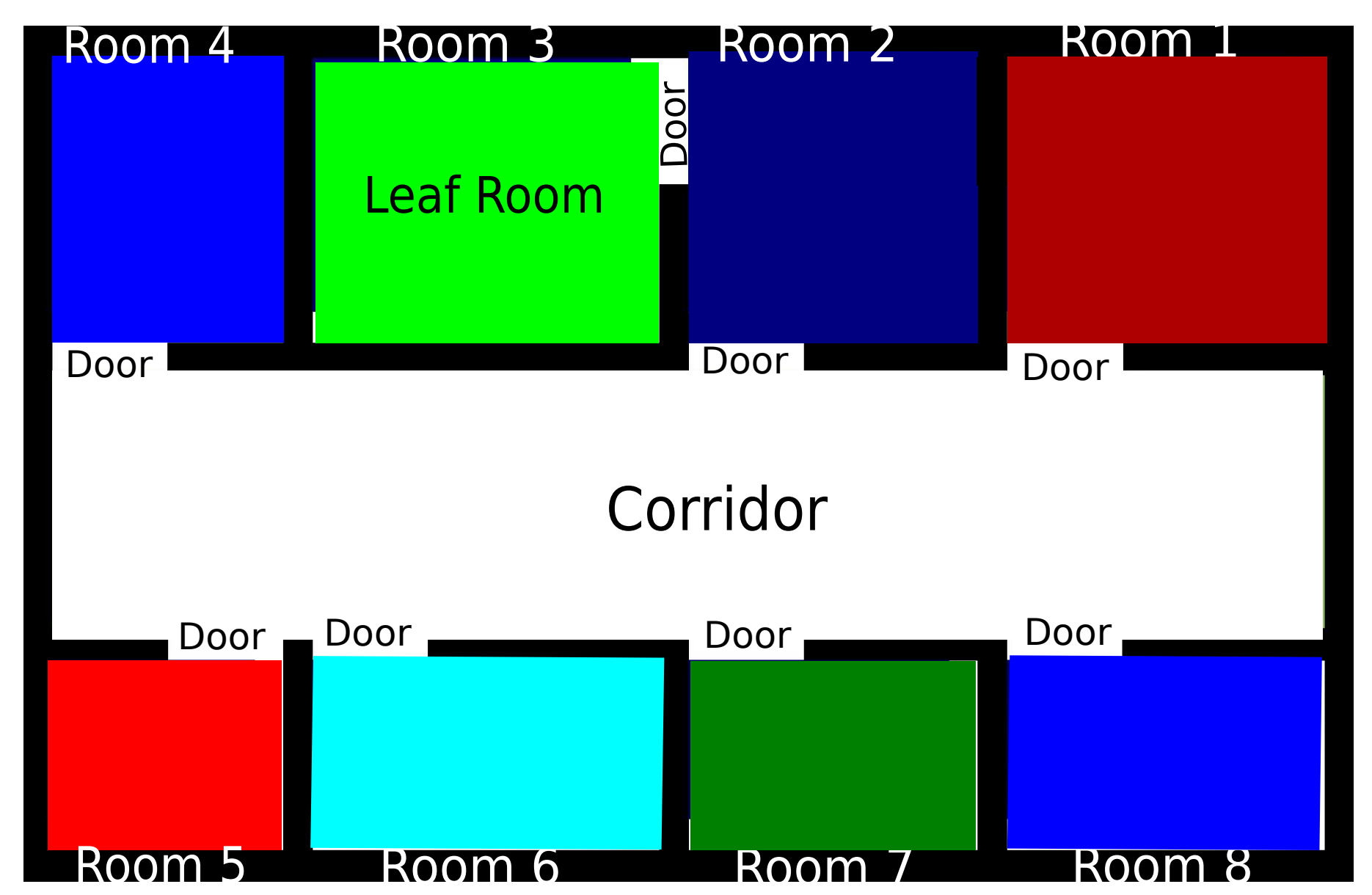


Figure 1: These rooms are tasks present in the environment (which need to be completed), the room color denotes the type of agent it needs and the darker a room is the more number of agents is required.

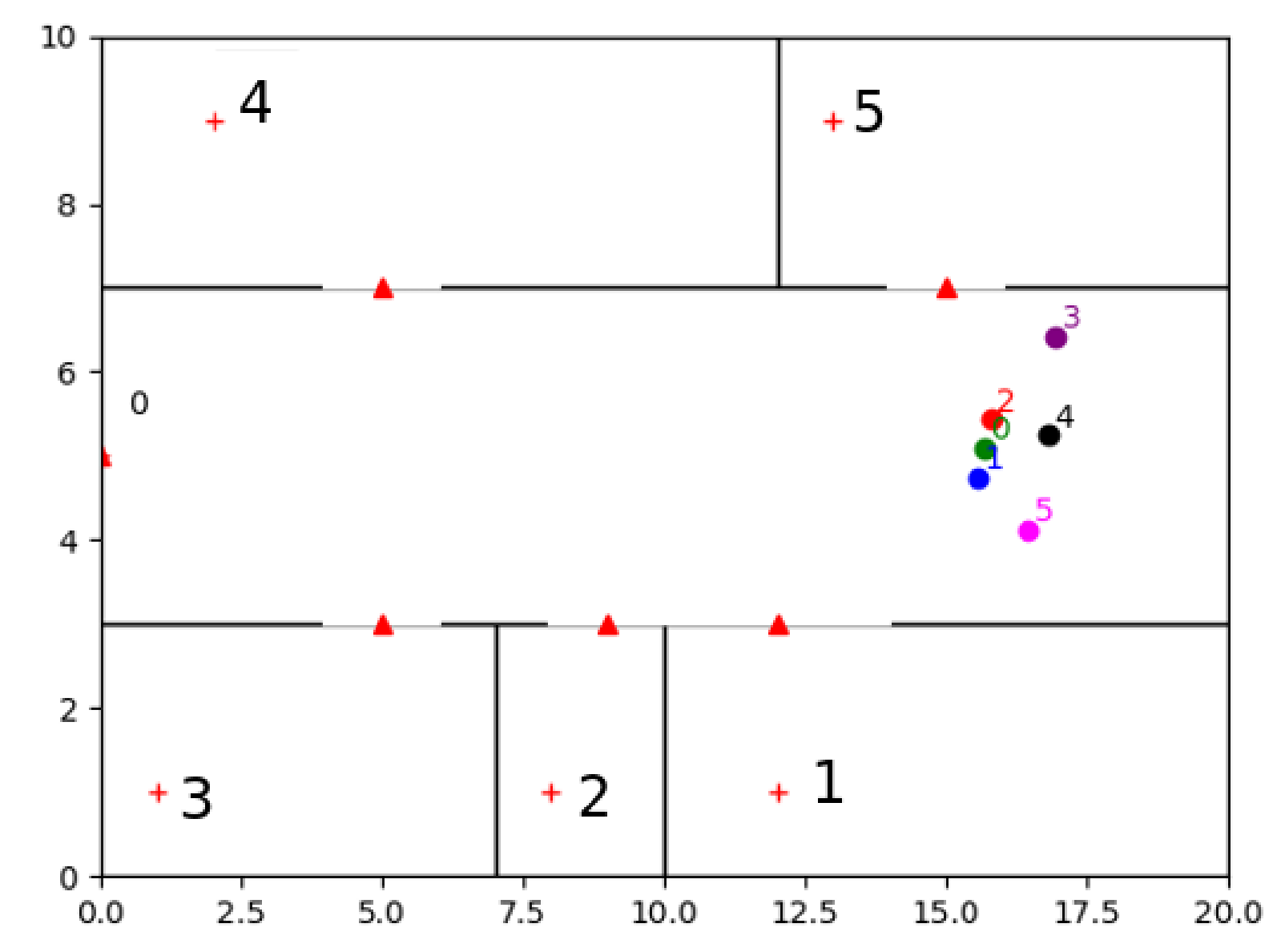


Figure 2: Environment without leaf room and same rooms, used for testing our proposed method.

Scenarios

	Room type	Agents Type	Room req.	Leaf Rooms
1.	same color	same color	fixed	NA
2.	diff color	RGB	fixed	NA
3.	diff color	RGB	varying	A
4.	diff color	RGB + white	varying	A

Method

We define a set of rooms(R), agents(A), doors, goals(G), available rooms(R_A), checked rooms(R_C), etc. and utility(U) and distance(D) matrix to solve an optimization problem based on constraints as shown below.

Scenario 1

Solve below objective after every time step

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & \lambda_1 u^\intercal x + \lambda_2 d^\intercal x \\ \text{subject to} \quad & E_1 x = \mathbf{1}_N, \\ & TE_2 x \geq TR_{req}, \\ & E_3 x = \mathbf{0}_{M+1}, \\ & x \in \mathbb{B}^{N(M+1)\times 1} \end{aligned} \tag{1}$$

where, $\lambda_1 \in [0,1]$ and $\lambda_2 \in [0,1]$,

Scenario 2 (different color rooms & agents)

Another constraint added to make same color agent go to same room by using $S \in \mathbb{B}^{(M+1)\times N}$ defined as

$$S(j,i) = \begin{cases} 1 & \text{if color}(R_j) = \text{color}(a_i) \\ 0 & \text{otherwise} \end{cases}$$

$$vec(S)^\intercal x == N(\text{ no. of agents}) \tag{2}$$

Scenario 3 (Leaf room as shown in figure 1)

1. Green leaf room undiscovered till blue agent enters blue room up to certain point.
2. Agents made to move slowly when crossing different colored rooms, so that if same color room discovered, then agents do not need to move in backward direction.
3. N adjacency matrix to account for the neighbours

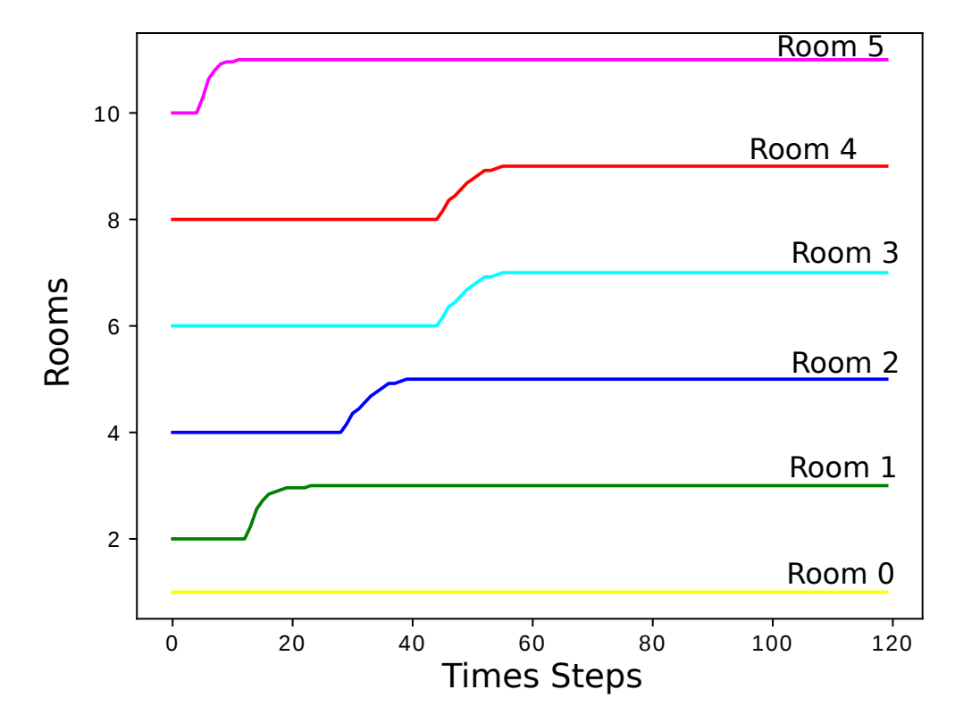
$\mathbf{N} \in \mathbb{B}^{(M+1)\times(M+1)}$ defined as

$$\mathbf{N}(j,i) = \begin{cases} 1 & \text{if door present} \\ 0 & \text{otherwise} \end{cases}$$

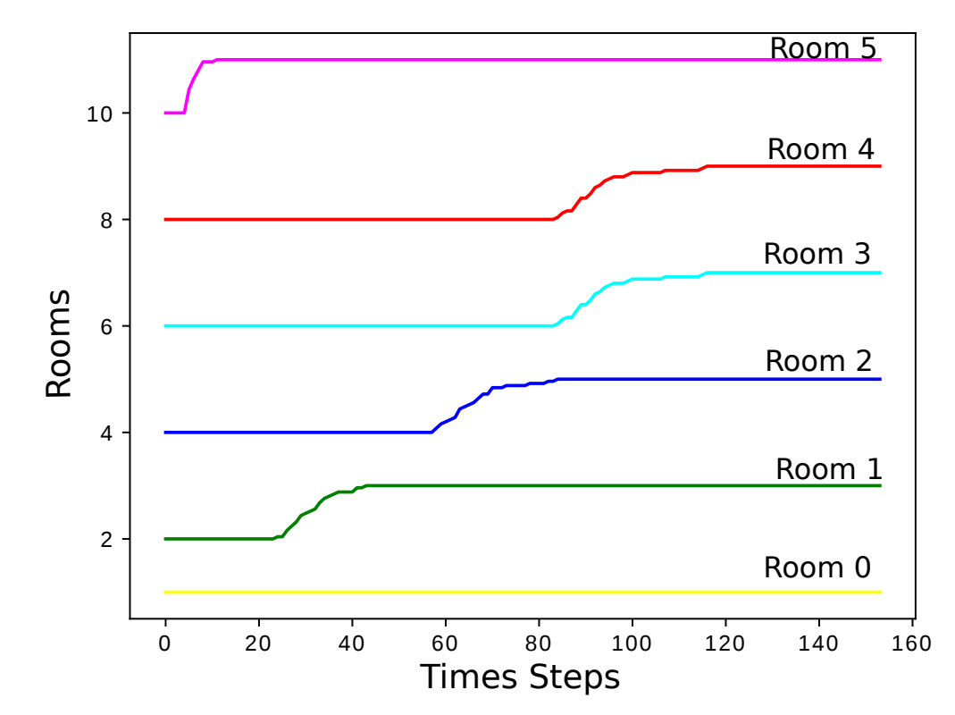
Scenario 4 (White agents)

1. White agents can enter any color room and then change color once room is checked.
2. Priority given to white agents.
3. if a room has a neighbouring leaf room, then white agents can reduce the number of other agents needed.

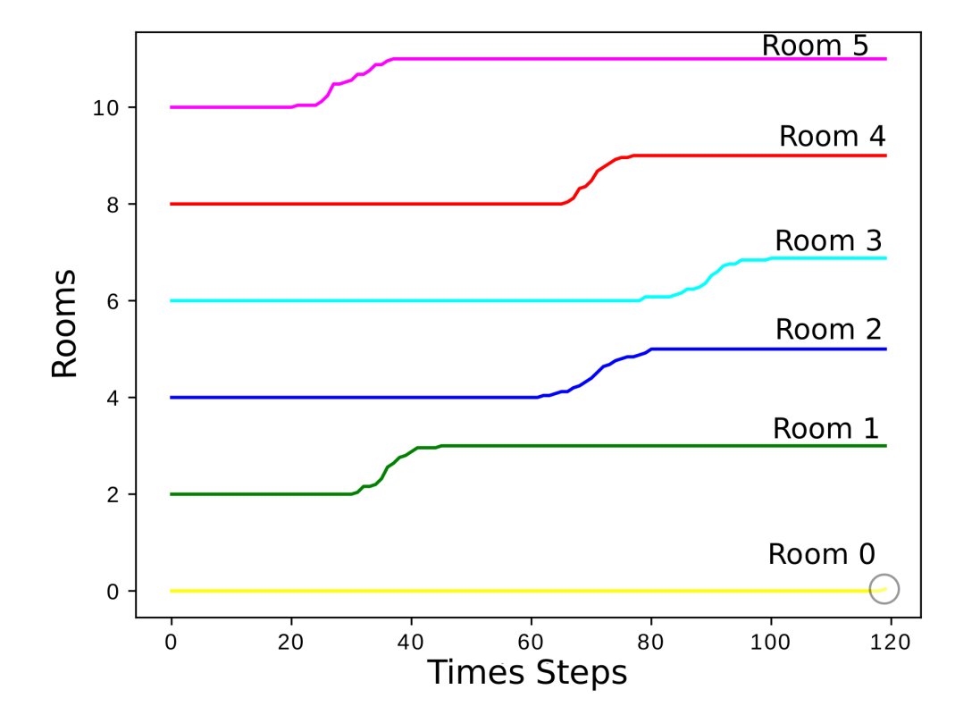
Results



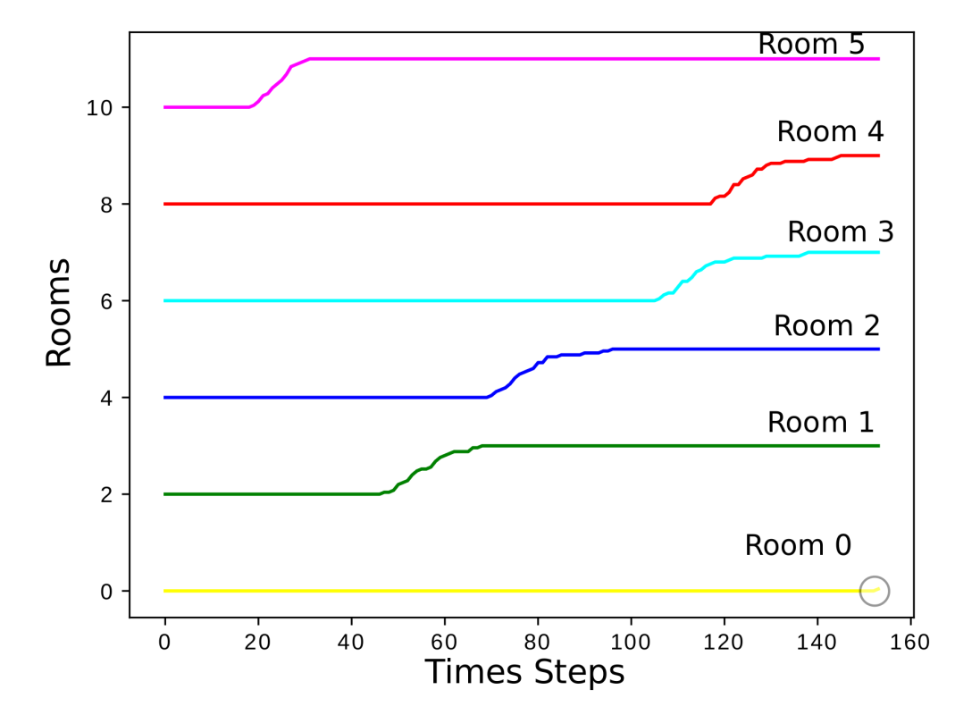
(a) Room available



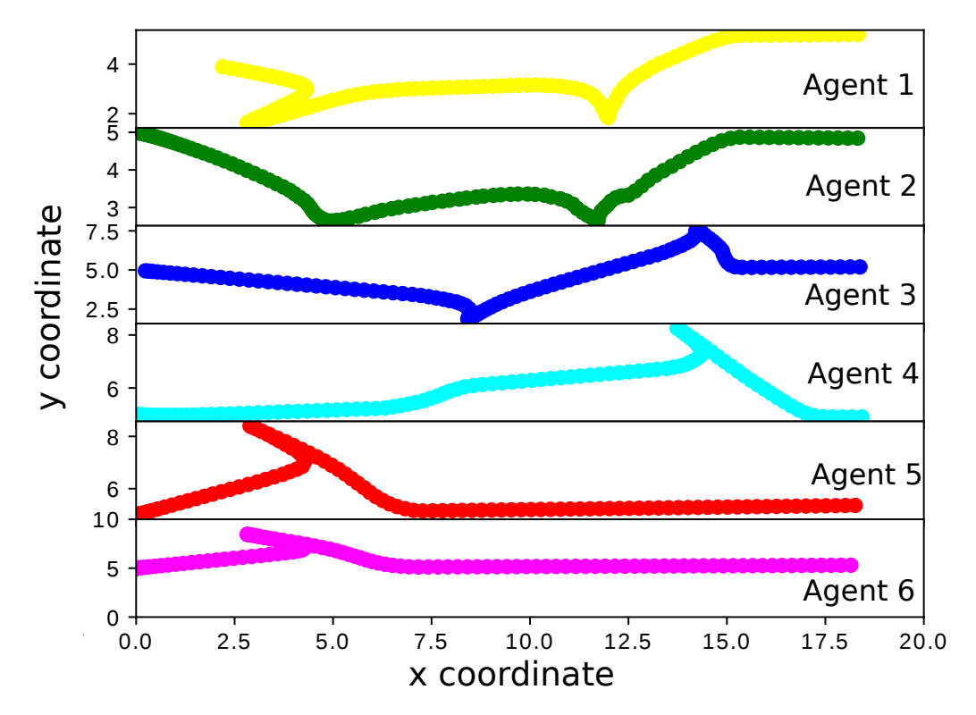
(b) Room available



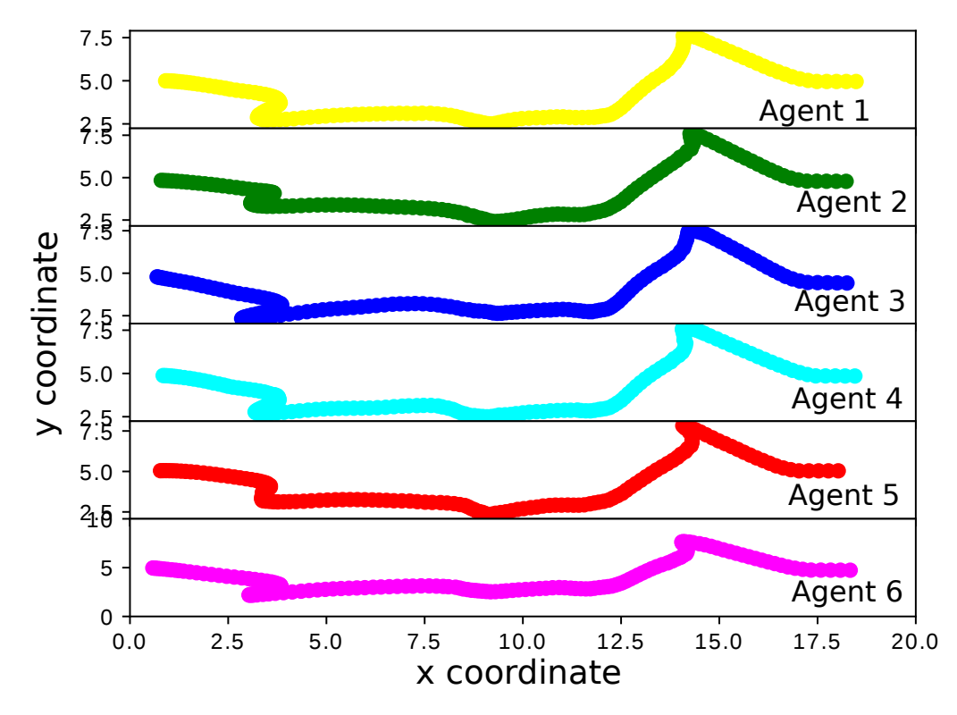
(a) Rooms Checked



(b) Rooms Checked

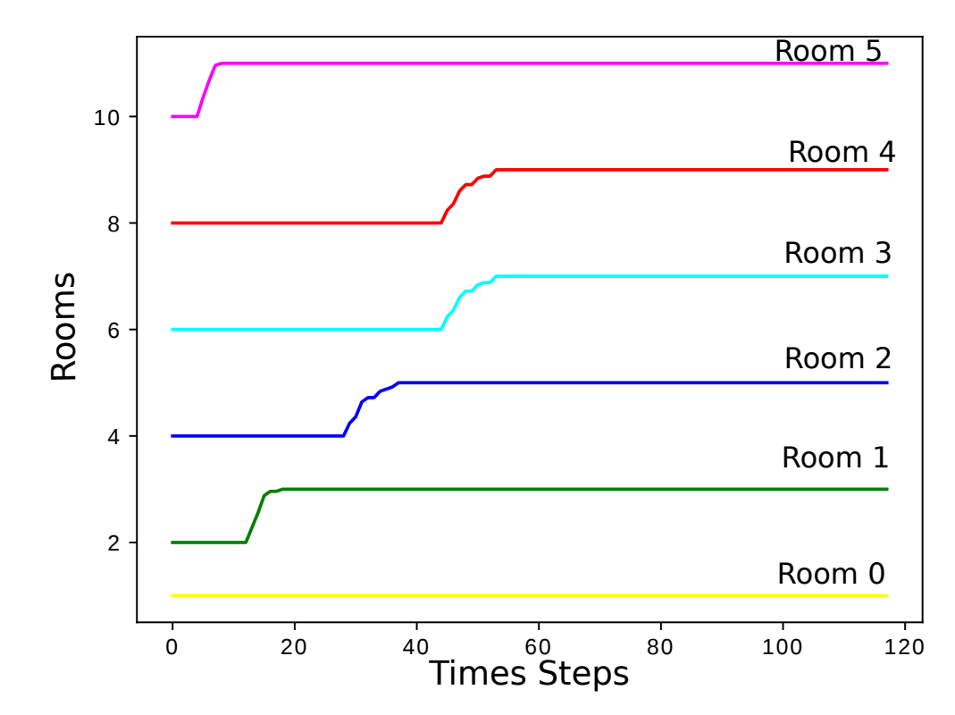


(a) Agents trajectory

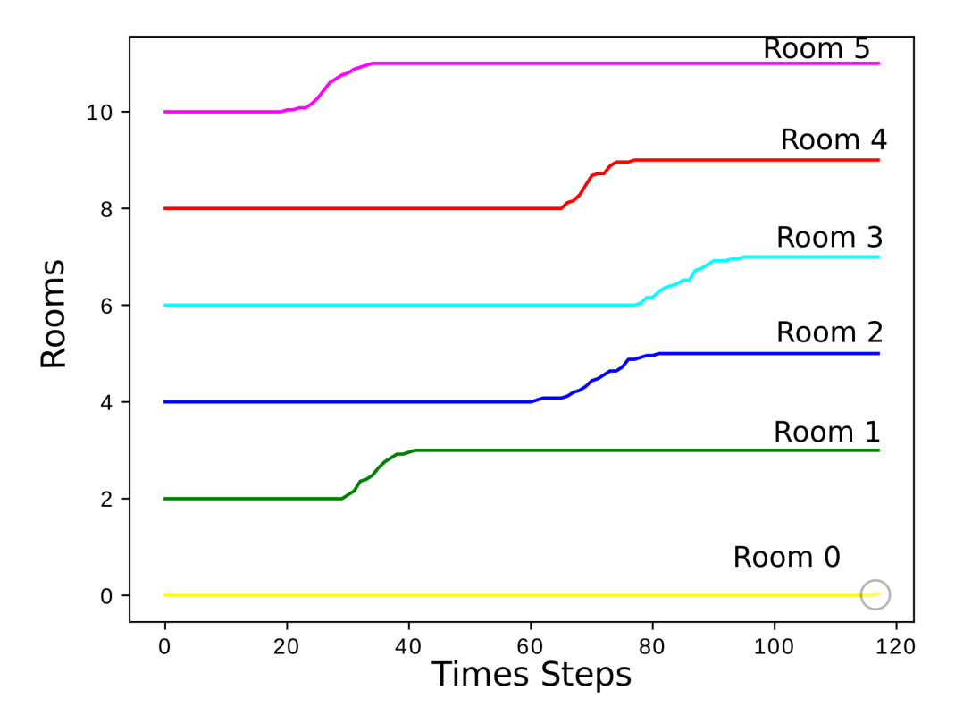


(b) Agents trajectory

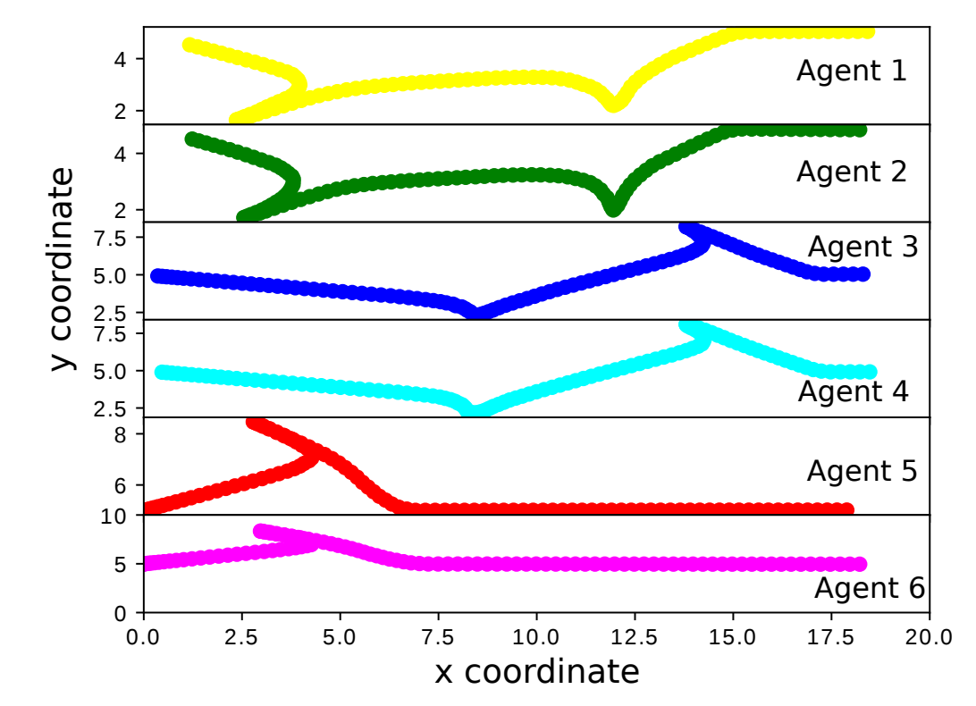
(a) have $\lambda_1 = 1.0, \lambda_2 = 0.0$ while (b) have $\lambda_1 = 0.0, \lambda_2 = 1.0$. The avg time for checking all rooms in (a) $= 132.84$ while in (b) $= 162.68$ when ran over 25 trials.



Rooms Available



Rooms Checked



Agents trajectory

Environment setting is as shown in figure 2 with the changes that R_1, R_3 : green; R_2, R_5 : blue; $R4$: red and A_1, A_2 : green; A_3, A_4 : blue; A_5, A_6 : red. The plots for the following cases was the same:
$\lambda_1 = 1.0$ (b): $\lambda_2 = 0.0$ and $\lambda_1 = 0.0, \lambda_2 = 1.0$. Hence, only plots for first case shown. Avg time for both was 136.

Conclusion and Discussion

- We observe that higher dependence on utility u i.e. λ_1 closer to 1, leads to faster coverage versus distance.
- However, when we introduce room color constraint then both the cases have almost equal maximum time taken to check all the rooms and average checking time is same.
- We still need to integrate our latest approach with barrier certificate to prevent inter agent collisions. Also, we need to further build for last two scenarios.

Future work

- Imagine a map with one room which has a pyramid of neighbouring room and these neighbours have more neighbours and so on, here sending as team of white agents seems the only feasible solution.
- Learning based techniques were also explored with similar inclination in the form of predator prey capturing, where the preys are like mobile tasks. But the environment was obstacle free and hence modelling an environment with obstacles and restrictions on certain areas being allowed for agents needs to be done.

References

[1] A. J. Smith, G. Best, J. Yu, and G. A. Hollinger, "Real-time distributed non-myopic task selection for heterogeneous robotic teams," 2019.

[2] A. Prorok, M. A. Hsieh, and V. Kumar, "Fast redistribution of a swarm of heterogeneous robots," 2016.

Acknowledgements

[1] would also like to acknowledge the support of the Robotics Institute Summer Scholars program and its organizers: Dr. John Dolan and Ms. Rachel Burcin. Also [1] is really grateful to all members of the AART Lab especially, Dr. Katia Sycara, Jaskaran Singh Grover, Sha Yi & Sumit Kumar