Medusa: Towards Simulating a Multi-Agent Hide-and-Seek Game

by

Akshat Tandon, Kamalakar Karlapalem

in

27th International Joint Conference on Artificial Intelligence and the 23rd European Conference on Artificial Intelligence
(*IJCAI-ECAI-2018*)

Stockholm, Sweden

Report No: IIIT/TR/2018/-1

Centre for Data Engineering
International Institute of Information Technology
Hyderabad - 500 032, INDIA
July 2018

Medusa: Towards Simulating a Multi-Agent Hide-and-Seek Game

Akshat Tandon and **Kamalakar Karlapalem**
Agents and Applied Robotics Group
Kohli Center on Intelligent Systems
International Institute of Information Technology, Hyderabad
akshat.tandon@research.iiit.ac.in, kamal@iiit.ac.in

Abstract

In the game of hide and seek, one or more agents (hiders) can hide behind objects to prevent other agents (seekers) from seeing them through their line of sight (visibility). These hider agents can be eventually found by seekers who explore the environment. The hide and seek model serves as an abstraction of many real world scenarios such as police agents trying to capture intruders hiding in a warehouse, security/patrolling agents on the lookout for any suspicious activity, spying drones trying to prevent themselves from being caught. Medusa allows modeling such scenarios by supporting the simulation, visualization and analysis of hider and seeker agent strategies under different environments.

1 Introduction

The game of hide and seek has been addressed by [Alpern and Gal, 2006], [Foreman, 1977], [Dagan and Gal, 2008], [Baston and Kikuta, 2013] and [Lidbetter, 2013] for certain classes of networks and multidimensional environments. [Chapman *et al.*, 2014] have employed hide and seek game strategies to tackle cyber security problems of attack attribution and attack pivoting. We explore the game from a multi-agent perspective in which the agents utilize visibility based percepts.

A feature of hide and seek based social scenarios is the notion of visibility and obstruction. A hider agent may be seen and eliminated from the game even if it is not in close proximity of any seeker but is visible to one. Depending on the domain, visibility may either be conical and restricted (humans) or may be all-around and unrestricted (drones/robots). Another feature of hide and seek social scenarios is the importance of abstracting spatial information. Certain positions allow greater coverage and some provide better obstructions. The hider and seeker agents can take advantage of these spatial abstractions during their planning phase to meet their objectives.

Multi-Agent simulators, such as Netlogo ([Tisue and Wilensky, 2004]), can be used for modeling visibility cones. However, no simulator exists which enables easy abstraction of spatial information which is appropriate for hide and seek social scenarios. Medusa stems from this need of enabling multi-agent simulation of *hide and seek game like social scenarios* by providing coverage and obstruction based spatial abstractions as well as easily configurable visibility based percept models to thousands of agents.

2 The Medusa Platform

We built Medusa[1], a multi-agent simulator which enables observation and study of individual and emergent behaviors in multi-agent social scenarios from a hide and seek game perspective. It provides API's for defining environments, spatial abstractions, agent behaviors and field of view, hider and seeker team structures and communication models. Medusa is scalable to thousands of agents.

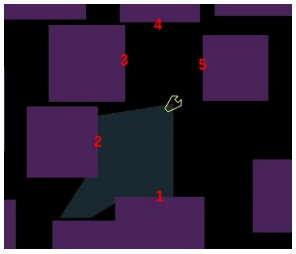
2.1 Environment, Agents and Teams

Medusa simulates the hide and seek game in a 2D bounded and continuous environment $\mathcal{E} \subset \mathbb{R}^2$. The environment contains many polygonal obstacles. For simulation purposes we consider a square as a basic obstacle block and construct any arbitrary polygonal shape as a composition of squares. A hider or a seeker agent is a point object in this environment. Both type of agents are usually mobile and can take one of the sixteen actions, each aligned to a compass direction. Hider and seeker teams can be constructed by grouping related agent types.

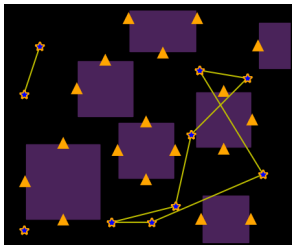
2.2 Objectives, Elimination and Visibility

The objective of the hider team is to delay their elimination time and that of the seeker team is to minimize it. A hider is eliminated from the game if it is visible to some seeker. To approximate the notion of visibility in simulation, Medusa associates a visibility region with each agent. A hider is visible to a seeker if the hider lies inside the seeker's visibility region. The visibility region depends on the current state of an agent and changes as the agent moves. For each agent, at each step, the simulator constructs the visibility region. It is constructed by tracing the path of uniformly spaced rays emitted from the agent's current position, spread uniformly along a fixed angle to the left and right of head facing direction of the agent. The maximum allowable distance and spread angle, of the rays, can be set according to the use case.

[1] https://github.com/droftware/Medusa



(a) Positions labeled 1,2,3,4,5 are the strategic points. Note the visibility region associated with the agent.



(b) Triangles are $\mathcal{SP}$, stars are $\mathcal{CP}$ and the yellow lines form the contours.

Figure 1: Spatial abstractions used during the planning phase

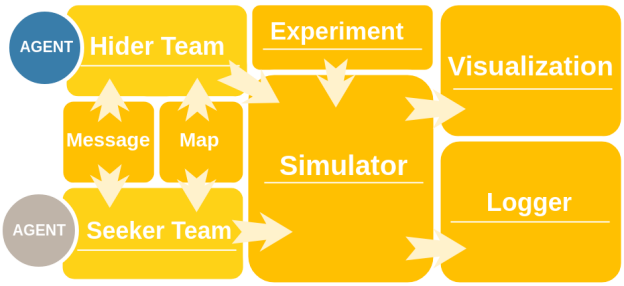


Figure 2: Architecture of Medusa

3 Spatial Abstractions

3.1 Strategic and Coverage Points

Given a map of the environment, Medusa provides several default spatial abstractions which can be utilized by agent strategies. A strategic point can be used as an abstraction for a hiding location. It is the mid point of an edge of a square obstacle. Each obstacle yields a set of strategic points and the union of these sets constitute the strategic points set $\mathcal{SP}$, of the environment $\mathcal{E}$.

A coverage point can be used as an abstraction for a seeking location. A coverage point is a point from which one or more strategic points are visible, if scanned in all the directions. It is said to cover the strategic points which are visible to it. An optimal set of coverage points $\mathcal{CP}$ is obtained by (1) enumerating over the maximal cliques [Bron and Kerbosch, 1973] of the visibility graph $\mathcal{VG}$ where $\mathcal{VG}.nodes = \mathcal{SP}$ and $\mathcal{VG}.edges = \{(u,v) \in \mathcal{SP}^2 | \exists w \in \mathcal{E},\ u$ and v are visible from $w\}$, (2) finding the smallest set of coverage points required for covering the strategic points corresponding to the nodes of enumerated clique and (3) taking the union of these coverage point sets. Maximal cliques of the visibility graph are considered because a maximal clique of $\mathcal{VG}$ encapsulates the set of strategic points which are visible to each other. Computational cost is reduced for the above algorithm by considering a discretized grid cell version $\mathcal{G}$ of environment $\mathcal{E}$. Each cell of $\mathcal{G}$ is represented by its center coordinates.

To find the smallest set of coverage points corresponding to a clique, iterate over all grid cells visible from some node (i.e. strategic point) of that clique and find the cell which covers the maximum number of nodes (i.e. strategic point) of that clique. If the found cell covers all the strategic points of the clique, return the center coordinates corresponding to the cell as the sole member of the coverage points set, else remove the strategic points covered by the found cell from the clique and feed the modified clique back to the algorithm to recursively find remaining coverage points.

3.2 Coverage Contours

We partition $\mathcal{CP}$ into ordered sets, called coverage contours, which have multiple common strategic points among themselves. Doing this enables a planner to exploit the locality around a coverage point better. These coverage contours can serve as abstractions of traversal routes for the seekers and can be obtained by enumerating over the connected components of the coverage graph $\mathcal{CG}$ where $\mathcal{CG}.nodes = \mathcal{CP}$ and $\mathcal{CG}.edges = \{(u,v) \in \mathcal{CP}^2 | \exists (w,x) \in \mathcal{SP}^2,\ w$ and x are visible from both u and $v\}$ followed by performing a postorder depth first search traversal to get an ordering.

4 Architecture

A hider or seeker agent's strategy is specified by the *Agent* module. It interacts with the *Message* module for the purpose of sending and receiving messages from other team mates and the *MapManager* module to process the environment and store it in the form of suitable spatial abstractions. A strategy can be described by defining the action which an agent should take and the messages it should send given a percept i.e. whether an agent is visible or not, history sequence and messages received. A hider or seeker team is specified using the *Team* module. A team can comprise of different agent types following different agent strategies. The *Team* module facilitates message passing among team members and connects individual agents to the *Simulator* module. The *Experiment* module specifies the context of the simulation i.e. environment map to be used, hider and seeker team types (agent strategies), number of hider and seeker agents, whether logging or visualization of the simulation is required or not, velocity of the agents, extent of an agents field of view and number of rays used to trace a visibility region. This context information is used by the *Simulator* module to instantiate the environment and the specified hider and seeker agents. It coordinates with the *Team* module to execute the actions dictated by the agents by changing their position as well as their visibility region. It also provides the visibility percept back to the agents. *Visualization* module shows the animation of the game played by hiders and seekers. It can provide both online (real-time) and off-line visualization (using replay log) of the game simulation.

5 Demonstration

In the demonstration, we will show the simulation in various environment scenarios. Visitors can even play against our strategies [Tandon and Karlapalem, 2018]. Video of our system and related simulations can be found at - https://youtu.be/c_U5MB6YhDY.

References

[Alpern and Gal, 2006] Steve Alpern and Shmuel Gal. *The theory of search games and rendezvous*, volume 55. Springer Science & Business Media, 2006.

[Baston and Kikuta, 2013] Vic Baston and Kensaku Kikuta. Search games on networks with travelling and search costs and with arbitrary searcher starting points. *Networks*, 62(1):72–79, 2013.

[Bron and Kerbosch, 1973] Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[Chapman *et al.*, 2014] Martin Chapman, Gareth Tyson, Peter McBurney, Michael Luck, and Simon Parsons. Playing hide-and-seek: an abstract game for cyber security. In *Proceedings of the 1st International Workshop on Agents and CyberSecurity*, page 3. ACM, 2014.

[Dagan and Gal, 2008] Arnon Dagan and Shmuel Gal. Network search games, with arbitrary searcher starting point. *Networks*, 52(3):156–161, 2008.

[Foreman, 1977] Joe G Foreman. Differential search games with mobile hider. *SIAM Journal on Control and Optimization*, 15(5):841–856, 1977.

[Lidbetter, 2013] Thomas Lidbetter. *Hide-and-seek and other search games*. PhD thesis, The London School of Economics and Political Science (LSE), 2013.

[Tandon and Karlapalem, 2018] Akshat Tandon and Kamalakar Karlapalem. Agent strategies for the hide-and-seek game. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems*, 2018.

[Tisue and Wilensky, 2004] Seth Tisue and Uri Wilensky. Netlogo: A simple environment for modeling complexity. In *International conference on complex systems*, volume 21, pages 16–21. Boston, MA, 2004.