Toward Simulating Realistic Pursuit-Evasion Using a Roadmap-Based Approach

Samuel Rodriguez, Jory Denny, Takis Zourntos, and Nancy M. Amato[⋆]

Parasol Lab, Dept. Computer Science and Engineering, Texas A&M University
{sor8786,jorydenny,takis,amato}@tamu.edu

Abstract. In this work, we describe an approach for modeling and simulating group behaviors for pursuit-evasion that uses a graph-based representation of the environment and integrates multi-agent simulation with roadmap-based path planning. We demonstrate the utility of this approach for a variety of scenarios including pursuit-evasion on terrains, in multi-level buildings, and in crowds.

Keywords: Pursuit-Evasion, Multi-Agent Simulation, Roadmap-based Motion Planning.

1 Introduction

The chase between a pursuer and an evader is an exciting display of dynamic interaction and competition between two different groups, each trying to outmaneuver the other in a time-sensitive manner. When this is further extended to include teams of pursuers, this problem combines cooperative problem solving with fast-paced competition in scenarios that are often seen in the real world. Both of these aspects have a wide range of applications, and are one reason for the prevalence of the pursuit/evasion problem in such a wide range of disciplines.

While much research has been done in the area, it has become difficult to unify or even compare these different techniques. For example there is no easy way to compare a purely theoretical exact solution to a graph-based pursuit/evasion problem or to a heuristic real world robotic example. Each discipline category of the pursuit/evasion problem has a set of restrictions and assumptions about the environment or the agents involved to make it more manageable. For example, in computational geometry the environment is often limited to polygonal shapes. In the case of graph-based approaches, the agents are limited to only moving on the nodes and the edges. While providing a means for developing a more complete solution, these assumptions limit the scope of the problem and

[⋆] This research supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266, NSF/DNDO award 2008-DN-077-ARI018-02, by the DOE NNSA under the Predictive Science Academic Alliances Program by grantDE-FC52-08NA28616, by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, HP, Oracle/Sun and by King Abdullah University of Science and Technology (KAUST) Award KUS-C1-016-04.

consequently the applicability of each technique. As a result, the study of the pursuit/evasion game has been fractured.

The existing approaches were designed to solve a very limited problem very well [17, 8, 10], but they are unable to create or duplicate a wide range of interesting pursuit/evasion motions. Each of these approaches does provide an interesting and useful perspective on the problem, but individually they remain too limited. We extend many existing approaches in an attempt to have a pursuit/evasion problem applicable to a much wider range of scenarios than are normally considered.

We propose agent based pursuit-evasion games where agents are equipped with a roadmap, encoding the free space in the environment and used for navigation. The roadmap is versatile in that it can act as abstract representation of potentially complex environments (generated in a number of ways) and facilitate communication by marking of nodes and edges. Agents are also equipped with a set of behaviors which define the actions they will take given their knowledge of the environment. The agents and behaviors are versatile and tunable in that the kind of agent being studied and the behaviors being applied can be adjusted with a set of parameters that define them. Our representation of the problem allows us to study problems in many unexplored areas which include pursuit-evasion games in crowds, terrains and multi-level buildings.

2 Related Work

In this section we describe work that is relevant to the pursuit evasion problem that we consider. A classical, purely graph-based approach [17] looks at the problem of pursuit-evasion on a graph where agents move on edges between nodes on a graph where the pursuer has the goal of occupying the same node as the evader. The problem has been looked at in polygonal environments [8], with much interest in finding exact bounds on the number of pursuers needed [16]. An approach based on exploring undiscovered portions of the frontier is described in [24].

Roadmap-based pursuit-evasion, where the pursuer and evader share a roadmap and play different versions of the pursuit-evasion game is described in [10]. These games include determining: 1) if the pursuer will eventually collide with the evader, 2) if the pursuer can collide with the evader before the evader reaches a goal location and 3) if the pursuer can collide with the evader before the evader collides with the pursuer in a dog fight scenario. In this work, alternate definitions of captures are give than many previous traditional approaches.

In [2–4], the benefits of integrating roadmap-based path planning techniques with flocking techniques were explored. A variety of group behaviors were simulated including exploring and covering utilizing an underlying roadmap. This paper builds on that work by adding group and role structures, making the framework more modular and with a focus on roadmap-based pursuit and evasion behaviors.

A number of other forms of the problem have been considered. A graph-based approach to the pursuit-evasion problem is given in [15] where agents use either blocking or sweeping actions by teams of robots to detect all intruders in the environment. Large teams of robots have been considered to detect intruders with communication allowed between other agents within some range and with no map of the environment [13]. The problem of agent's having limited sensing information by building gap navigation trees with gap sensors is considered in [20]. The idea of using probabilistic shadow information spaces for targets which moving out of a pursuing agent's field-of-view has also been considered [25]. Other forms of collaboration between searching agent's has been considered where agents utilize frontier information [5] and to use the frontier information with communication to switch between roles [6].

The level of visibility between agents plays a role in what the agent's detect in the environment and what kinds of environments can be handled. The problem of restricting or limiting the amount of visibility is considered in [9] with the field of view being variable in [7]. Teams of robots with different kinds of vehicles and sensing information is looked at in [12]. The problem of pursuit-evasion on height maps, which restricts agent visibility is described in [14].

Crowds can also vastly affect the look of virtual environments where many agents may be moving in an environment [22, 21] or where animated characters may be following simple rules to simulate natural behaviors [18]. By considering crowds, the kinds of pursuit-evasion games that can be played becomes much greater, especially when considering the idea that the crowd can block visibility. Sophisticated, path planning techniques have been created for planning the motion of agents in crowded areas [22], some with specific applications of pursuit-evasion scenarios [19].

3 Framework Overview

Many of the previously proposed approaches are often very restrictive on the assumptions and the environments that can be handled. Examples include focusing on polygonal environments, strictly on a graph, or planar environments. Our roadmap-based approach allows us to handle many classical pursuit-evasion problems along with many that have not been considered in the literature. This includes agents moving in 3D environments such as on a terrain, in multi-level environments or in areas with crowds. Here we describe our overall system and approach to this problem.

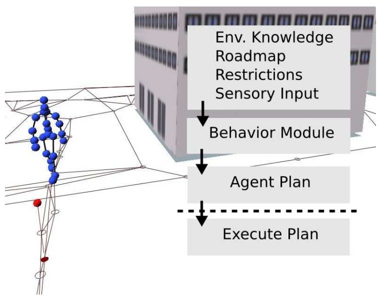
3.1 Problem Definition

The traditional pursuit/evasion problem that we consider consists of one set of agents, the pursuers H, attempting to capture another set of agents, the evaders E. An evader $e \subset E$ is considered captured if a non zero set of pursuers $h \subset H$ fulfills some predefined distance requirement. The pursuers and evaders may have different levels of environmental knowledge, sensing abilities and capabilities, which effect the overall time to capture. The pursuit we study is a complete chase which consists of searching for a target agent and maintaining visibility until a capture can occur. The evading agents take adversarial actions which involves fleeing from pursuing agents that have been detected and improving on hiding locations when undetected.

3.2 Approach

In an attempt to explore a range of the entire pursuit/evasion spectrum we have developed a unique approach that allows for quick and easy development of strategies for different versions of the problem. Our approach gives the user full control over a number of parameters including agent and behavioral properties. This ability allows us to explore a wide spectrum and provides the basis for a framework that provides a great deal of flexibility and control to the user over interesting pursuit/evasion motions and simulations.

Our approach uses a real-time simulation that allows for the movement of agents using the roadmap, complex pursuit/evasion strategies, and interesting agent interactions with both the environment and other agents. We have designed and implemented a simulation infrastructure for storing and manipulating the following information: agents, behaviors, the environment, groupings, and relationships between groups.



The outline of our simulation loop is given in Algorithm 1. It is important to note that we use a general concept of individual agent, calling them a group (with no subgroups). In this way our framework can be more general, especially when creating behaviors since a behavior created for individual agents can easily be applied to a grouping of agents, with some additional logic used to ensure grouping restrictions are maintained.

The behavior that the agent is equipped with will determine how the agent reacts throughout the simulation. These behaviors determine the actions that the agent or groups of agents take. At each time step, all groups update their state based on the last plan that was either generated or updated by the behavior rule. The state is then resolved which includes with other agents and the environment (i.e. preventing collision) as well handling any interactions that may have

Algorithm 1 General Simulation Loop

```
Input: simulator $sim$, environment $env$
1: $groups_{all}$ = sim.getAllGroups()
2: for $g \in groups_{all}$ do
3:     $g$ →applyBehaviorRule($env$)
4: end for
5: for $g \in groups_{all}$ do
6:     updateState($g$)
7: end for
8: ResolveStateWithAgents_Environment($groups_{all}$,$env$)
9: Evaluation
```

occurred between groups. This includes sending any communication messages that may have been generated throughout the behavior process or roadmap reweighting. Since all interactions are delayed and only processed at the end of each time step, the environment remains constant in relation to all the agents during the individual time step. By processing the interactions in this manner, we ensure that the order in which the agents are processed does not affect their behavior.

The environments types that we have traditionally focused on are 2D environments consisting of polygonal obstacles. In this work we extend our roadmap-based approach to 3D environments where agents can move on surfaces which alter the agent's height component along with affecting visibility. This extension allows us to handle 3D environments consisting of terrain and multi-level environments such as buildings. A terrain is a surface consisting of polygons in 3-dimensions that may represent hills and valleys whereas a building may be represented as a connected set of these surfaces which the agent may move on.

3.3 Roadmaps

Roadmaps are graph representations of an environment that encode feasible, collision-free paths through the environment. Our approach allows these graphs to be generated in variety of ways. For traditional 2D environments, we used probabilistic roadmaps [11], which are generated by first sampling nodes in the free space of the environment and creating paths between nodes that satisfy certain constraints (e.g. collision-free connections). A simple local planner is used to connect nearby nodes. The network of nodes and paths form a graph that the agent is able to query in order to find a path from its current position to an end goal. Depending on the agents being simulated, we allow enhancements to the PRM approach which improve where and how sampling is done – for example near obstacle boundaries [1] or near the medial axis of the environment [23].

Agents navigating in environments consisting of multi-level surfaces such as terrain or buildings need the ability to map these spaces as well. We allow the agents to move on the surfaces by sampling nodes on each surface. Connections are allowed between nodes on the same surface in which the straight line along that surface, projected to a 2-dimensional plane, remains completely within the project polygon. Connections are then made between surfaces that are connected based on an input configuration which allows us to map multi-level surfaces and terrains along with connections between surfaces and the default 2D surface.

Roadmaps can also encode global information which may be useful to groups of agents, such as by storing information about an area to avoid, or by associating certain zones in the environment with certain groups of agents. Additionally, agents can then perform searches on the roadmap based on actions and observations that other agents they are cooperating with have made. These graphs/roadmaps are an essential component for our pursuit/evasion scenarios. Other benefits of the roadmap include the low computational cost of querying them once they are created and repairing them when the environment has changed. Rather than having to check for collisions during every timestep (which would be very computation-intensive), agents can query a valid roadmap quickly for paths guaranteed to be both available and collision-free.

The roadmaps are used to generate paths through the free areas of the environment that will guide an agent from some start to goal location. These paths can be adhered to strictly or act as a guide and optimized based on agent criteria depending on the type of problem being considered. In many of the examples we study, we allow the agent's to use the paths from the roadmap as merely a guide through the environment. Agent path optimization parameters allow the agent to improve the path to more quickly reach a goal location requiring that the optimized path remains collision free. As the agent is navigating through the environment following a path generated, another parameter determines the distance required for a subgoal to be considered reached. These parameters play an important role in determining the amount an agent adheres to the roadmap and the time it takes an agent to reach a goal in the environment.

3.4 Visibility Between Agents

The agent's ability to detect other agents in the environment can be affected by a number of aspects. In simple 2D problems, the agent's visibility to other agents may only be restricted by the obstacles in the environment. Agent capabilities can also determine the amount of the environment that can be sensed by setting a maximum view radius and angle. Visibility can be further restricted by considering other agents that may block the view of other agents in the environment, i.e. the 3D representation of all agents used as potential obstacles in visibility checks. When surfaces are included in environments, these surfaces may also block visibility from one agent to another. We have included each of these aspects in our visibility checks which allows us to handle complex surfaces and crowd examples.

4 Pursuit Behavior

Our general pursuit strategy has four stages: location of the target, creation of a pursuit plan, acting upon the plan, and then evaluating pursuit status. A wide range of pursuit strategies can be employed by enabling certain customizations of the general pursuing strategy. The most basic pursuit strategy will chase a target, once located, in a direct path until it either captures the target or the target is no longer visible.

4.1 Search Behaviors

The searching behaviors that agents use in order to locate a target can greatly effect the pursuing agents effectiveness. We have implemented a number of roadmap-based searching behaviors which allows groups of agents to utilize the roadmap in order to effectively cover an environment [3]. The goal of the searching behaviors is for the agents in the group to visit as much of the environment as possible. For these behaviors, agents use the roadmap to obtain pathways to free areas of the environment. The effectiveness of the search behavior has an underlying dependence on the quality of the roadmap used by the agents. Agents can either locally try to find nodes that have been visited the least or search for some random area in the environment.

4.2 Enhancements

The basic pursuit strategy consists of an agent chasing a target toward the target's current location. Many improvements to this basic pursuit strategy can be enabled which potentially improve a group of pursuing agent's success rate. Pursuing agents can head off an evading agent by planning their path to the target ahead of the evading agent; we refer to this as a heading behavior. Pursuing agents can also avoid over re-planning by only creating a new pursuit plan when the evading agent has moved far enough away from the previously planned pursuit route. This can prevent erratic motion of the pursuing agent.

Enabling communication between pursuing agents is another aspect that can improve pursuing agent's effectiveness. This could allow agents that may not detect an evading agent to be made aware of their location enabling a pursuit plan to be generated. Communication between agents can also allow for one agent, the leader, to request other agents to follow that agent when searching the environment. This level of communication allows for potentially more coordination between pursuing agents which is necessary in some pursuit scenarios and something we are interested in studying in the future.

5 Evading Behavior

Our evasion strategy attempts to reduce the likelihood of being or staying visible to opposing agents. In this behavior, an agent generates hiding locations within a certain range in the environment. These points are then evaluated, or scored. Different scoring functions are used depending on whether or not pursuers are present. A new hiding location can then be selected if one is found such that the score of the new location achieves some predefined percentage increase over the score of the current hiding location being used. The new goal and path are updated if a point is found that sufficiently improves the score.

5.1 Scoring Hiding Locations

In the case of an evading agent being undetected by pursuing agents, the goal of the evading agent is to decrease it's likelihood of being detected. The score for each hiding location generated is determined by the visibility to all the other hiding locations generated. A hiding location with a low visibility is preferred over those with high visibility, i.e. a location that is blocked by objects in the environment.

When an evading agent is detected, five criteria are used for scoring hiding locations:

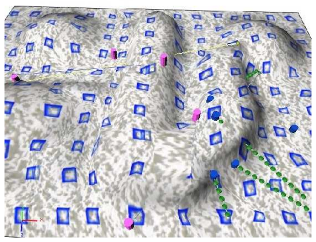
- **Distance to Pursuers:** This value will determine how much an agent prefers locations that are further away from the visible agents. The distance a hiding location is from all visible pursuing agents can be accumulated and then scaled by the evading agent's view radius.
- **Direction to Pursuers:** The direction to a hiding location is computed and scored such that directions away from pursuing agents are preferred.
- **Distance To Boundary:** The distance to the boundary of a hiding location can be factored in to the final score by weighting the factor when the potential hiding location is within some predefined distance to the boundary. This factor will allow agents to prefer hiding locations away from the boundary and reducing the likelihood of being trapped by the boundary.
- **Visibility Restricted Surfaces:** The weight of this factor can be determined by how much a hiding location is obscured by surfaces in the environment. In this way, agents would prefer hiding locations that are not visible to pursuing agents.
- **Visibility Restricted by Other Agents:** Other agents can be used when scoring hiding locations. A hiding location can be weighted such that an agent prefers a location that has other agents in between pursuing agent locations and the hiding location.

Each of these factors can have weights associated to these factors so that evading agents can be tuned to prefer certain kinds of evasion tactics. These tunable factors allow for our adversarial evaders to exhibit potentially very different behaviors and are different from many approaches that assume the evading agents has a set evasion strategy.

6 Experiments

We present results for pursuit/evasion scenarios that have not received as much attention as the traditional problems. These involve more complex environments where the visibility of the agents is very much restricted by the environment. We also show examples where some interesting interaction takes place between the

Table 1. Pursuit/Evasion on a terrain with hills and valleys obscuring visibility in the environment.



Scenario	Num Capt.	Time To Capture	Min/Max Time Chasing	Min/Max Time Hidden	Prop. Time Hidden
basic	1	3686.7	(1902,3189)	(1538,3461)	0.5261
heading	3.25	1363.5	(1056,2792)	(409,2556)	0.3044
send target	4	1028	(1133,2520)	(523,3980)	0.4169

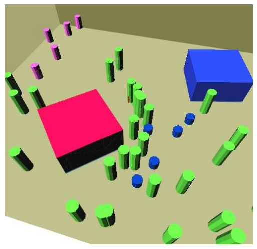
groups of agents being simulated. In the following, we show results in environments consisting of a terrains, crowds, and a multi-level example. We also show an example of interacting behaviors where agents play a game of tag with varying pursuit and evasion behaviors. Results show the average number of captures, the average time to capture, minimum and maximum time spent chasing among the pursuing agents, minimum and maximum time hidden for the evading agents and the proportion of time hidden. These values give insight into how involved each agent was in the pursuit process both in the chasing aspect and the searching of the environment. The maximum number of time steps used throughout any simulation is 5000.

7 Terrain

The terrain environment, shown in Table 1, consists of multiple hills and valleys with dimensions of 195 by 150 units and a maximum height 18.5 units. This provides numerous hiding locations for the evading agents. In this experiment 5 pursuing and 5 evading agents react to one another in the pursuit evasion game until the pursuing agents capture the evading agents. The pursuit behaviors tested in this environment are: basic pursuit, heading off the evading agents, and sharing targets between pursing agents which are also heading off. Pursuing agents have a maximum velocity of 4, view radius of 40 and height of 6 units. Evading agents have a maximum velocity of 6, view radius of 50 and height of 3. In this scenario, the advantage lies with the evading agents where they are faster and can sense more of the environment than the pursuing agents.

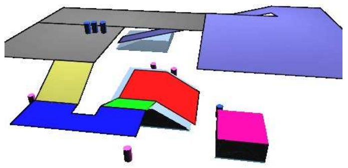
In Table 1, it can be seen that using only basic pursuit results in fairly poor performance since the evading agents can easily outmaneuver the pursuers. When pursuit agents attempt to head off the evading agents, the performance improves overall resulting in more captures and lower time to capture. While adding the flank behavior does not improve results over using a heading off behavior, sharing a target among the pursuing agents greatly improves the pursuit agents effectiveness. Agents utilizing the most effective enhancements(heading off and sharing targets) allows agents to restrict the number of evasive actions.

Table 2. Pursuit/Evasion in a crowd of other agents (5 pursuing, 5 evading, 30 in crowd). Pursuing agents shown in Top Left corner of image, evading agents shorter, blue cylinders at center and crowd are tall, green cylinders.



Scenario	Num Capt.	Time To Capture	Min/Max Time Chasing	Min/Max Time Hidden	Prop. Time Hidden
basic	3	1013.2	(19,376)	(1152,4727)	0.5889
heading, share target	5	448.4	(126,543)	(418,1744)	0.4104

Table 3. Pursuit/Evasion in a multi-level building example.



Scenario	Num Capt.	Time To Capture	Min/Max Time Chasing	Min/Max Time Hidden	Prop. Time Hidden
basic	5	784.8	(1103,2453)	(728,1349)	0.2428
send target	3	1236	(284,1267)	(1457,3921)	0.5082

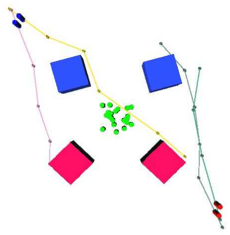
8 Crowd

In an environment with a crowd, the evading agents can use the crowd members to score hiding locations as they can obscure the visibility of pursuing agents. While the test environment is very basic, including the crowd in the problem adds another level of complexity. The environment dimensions are 110 by 100 units with the members of the crowd tall enough to obscure visibility. The pursuit behaviors tested are: a) basic pursuit, and b) using a heading off and shared target behavior. The basic pursuit again doesn't perform as well as using a heading off behavior along with a shared target. The most enhanced behavior results in the best performance with the highest number of captures and lowest average time to capture. This behavior among pursuing agents also alerts other agents to a target that may end up hiding in the crowd, but the subsequent searching in the crowd often resulted in a capture.

9 Multi-Level

The multi-level environment is a very complex environment consisting of two main floors connected to each other in two ways. One connection is a long ramp,

Table 4. Pursuit/Evasion applied in a game of tag. Two pursuing agents start in each type of behavior (basic, heading) and attempt to capture an evading agent. An evading agent that is captured becomes part of the pursuing agent's team, executes the same behavior and can be used to capture. The final number of agents in each pursuing agent behavior is shown.



	Pursuit 1	Pursuit 2	Num. Left Evading	Time Complete
basic/basic	12.2	12.2	0.6	4708
heading/basic	17.8	6.8	0.4	4780
heading/heading	12.4	12.4	0.2	3606

in the background in the environment shown in Table 3. The other connection consists of multiple surfaces ramping to the second floor. Two basic pursuit behaviors are tested a basic pursuit and pursuit with a shared target. It is interesting to note that in this example, sharing a target among pursuing agents does not result in better performance. In this environment, agents independently searching the environment increase the chance that the pursuing agents will trap agents on the upper floor. This reduces the potential for the evading agents to find valid evasive actions. Agents that are independently searching, also reduce the chance of an evading agent to stay in a hiding location for very long.

10 Tag Game

The game of tag played in this scenario is a very simple extension of the standard pursuit/evasion game. In this scenario, two sets of pursuing agents with the same capabilities start out with predefined pursuit behaviors. The evading agents in the environment have the goal of avoiding both sets of pursuing agents. Once an evading agent is captured, it becomes part of the capturing agent's team and begins executing the same pursuit behavior. In Table 4, pursuit behavior comparisons are shown for basic vs. basic pursuit, heading vs. basic pursuit, and heading vs. heading pursuit behaviors. We initialize each set of pursuing agents with two pursuers. The numbers shown in Table 4 reflect the final number of agents performing the behaviors, average number of agents left evading and the average time to complete the game.

When each type of pursuit behavior is tested against the other, on average the same number of agents end up in each pursuit group. When comparing heading against basic pursuit behaviors, it can be seen that many more agents end up in the heading behavior group which shows that using a heading behavior is much more effective in this scenario. Another interesting aspect is that when both pursuit behaviors use a heading behavior, the average time until the game is complete is much lower which shows just how effect this pursuit behavior is.

11 Conclusion

References

1. N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Robotics: The Algorithmic Perspective*, pages 155–168, Natick, MA, 1998. A.K. Peters. Proc. Third Workshop on Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.
2. O. B. Bayazit, J.-M. Lien, and N. M. Amato. Better flocking behaviors using rule-based roadmaps. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 95–111, Dec 2002.
3. O. B. Bayazit, J.-M. Lien, and N. M. Amato. Better group behaviors in complex environments using global roadmaps. In *Artif. Life*, pages 362–370, Dec 2002.
4. O. B. Bayazit, J.-M. Lien, and N. M. Amato. Roadmap-based flocking for complex environments. In *Proc. Pacific Graphics*, pages 104–113, Oct 2002.
5. W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 476–481, 2000.
6. J. W. Durham, A. Franchi, and F. Bullo. Distributed pursuit-evasion with limited-visibility sensors via frontier-based exploration. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3562–3568, 2010.
7. B. P. Gerkey, S. Thrun, and G. Gordon. Visibility-based pursuit-evasion with limited field of view. *Int. J. Robot. Res.*, 25(4):299–315, 2006.
8. Leonidas J. Guibas, Jean claude Latombe, Steven M. Lavalle, David Lin, and Rajeev Motwani. Visibility-based pursuit-evasion in a polygonal environment. In *International Journal of Computational Geometry and Applications*, pages 17–30. Springer-Verlag, 1997.
9. V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion with limited visibility. In *Proc. ACM-SIAM Symposium on Discrete Algorithms*, pages 1060–1069, 2004.
10. V. Isler, D. Sun, and S. Sastry. Roadmap based pursuit-evasion and collision avoidance. In *Proc. Robotics: Sci. Sys. (RSS)*, 2005.
11. L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.
12. H.J. Kim, R. Vidal, D.H. Shim, O. Shakernia, and S. Sastry. A hierarchical approach to probabilistic pursuit-evasion games with unmanned ground and aerial vehicles. In *Proc. IEEE Conf. on Decision and Control*, pages 634–639, 2001.
13. A. Kolling and S. Carpin. Multi-robot pursuit-evasion without maps. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3045–3051, 2010.
14. A. Kolling, A. Kleiner, M. Lewis, and K. Sycara. Solving pursuit-evasion problems on height maps. In *IEEE International Conference on Robotics and Automation (ICRA 2010) Workshop: Search and Pursuit/Evasion in the Physical World: Efficiency, Scalability, and Guarantees*, 2010.
15. Andreas Kolling and Stefano Carpin. Pursuit-evasion on trees by robot teams. *Trans. Rob.*, 26(1):32–47, 2010.
16. Steven M. Lavalle, David Lin, Leonidas J. Guibas, Jean claude Latombe, and Rajeev Motwani. Finding an unpredictable target in a workspace with obstacles. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 737–742, 1997.

17. T. D. Parsons. Pursuit-evasion in a graph. In *Theory and Applications of Graphs*, volume 642 of *Lecture Notes in Mathematics*, pages 426–441. Springer, 1978.
18. C. W. Reynolds. Flocks, herds, and schools: A distributed behaviroal model. In *Computer Graphics*, pages 25–34, 1987.
19. A. Sud, E. Andersen, S. Curtis, L. Ming, and D. Manocha. Real-time path planning for virtual agents in dynamic environments. In *Virtual Reality Conference*, March 2007.
20. B. Tovar, R. Murrieta-Cid, and S. M. LaValle. Distance-optimal navigation in an unknown environment without sensing distances. *IEEE Transactions on Robotics*, 23(3):506–518, 2007.
21. Adrien Treuille, Seth Cooper, and Zoran Popovi. Continuum crowds. *ACM Trans. Graph.*, 25(3):1160–1168, 2006.
22. Jur van den Berg, Sachin Patil, Jason Sewall, Dinesh Manocha, and Ming Lin. Interactive navigation of multiple agents in crowded environments. In *Proc. Symposium on Interactive 3D Graphics and Games - I3D'08*, 2008.
23. S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1024–1031, 1999.
24. Brian Yamauchi. Frontier-based exploration using multiple robots. In *International Conference on Autonomous Agents (Agents '98)*, pages 47–53, 1998.
25. J. Yu and S. M. LaValle. Probabilistic shadow information spaces. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3543–3549, 2010.