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

Samuel Rodriguez, Jory Denny, Juan Burgos, Aditya Mahadevan, Kasra Manavi,
Luke Murray, Anton Kodochygov, Takis Zourntos, and Nancy M. Amato
Parasol Lab., Dept. of Computer Science and Engineering, Texas A&M Univ., College Station, Texas, 77843-3112, USA
{*sor8786,jorydenny,ramietesaima,aditya.mahadevan,kmmanavi,tromboneman,akodochygov,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. Our approach can be applied to more realistic scenarios than are typically studied in most previous work, including agents moving in 3D environments such as terrains, multi-story buildings, and dynamic environments. We also support more realistic three-dimensional visibility computations that allow evading agents to hide in crowds or behind hills. We demonstrate the utility of this approach on mobile robots and in simulation for a variety of scenarios including pursuit-evasion and tag on terrains, in multi-level buildings, and in crowds.**

I. INTRODUCTION

The chase between a pursuer and an evader is an exciting display of dynamic interaction and competition between two 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 a wide range of disciplines.

While a great deal of work has been done in the area, each 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 the case of graph-based approaches, the agents are limited to moving on the nodes and the edges [1]. In geometry based approaches, the environment is often limited to being polygonal [2]. In fact, the types of visibility restrictions that agents can handle is often one of the main limiting factors [3], [4]. While providing a means for developing a more complete solution, these assumptions limit the scope of the problem and consequently the applicability of the techniques.

The main contribution of this work is our versatile approach for agent-based pursuit-evasion games that extend the range of applications that we are able to handle. We are able to extend the scenarios to 3D-scenarios that require more complex visibility checks that consider obstacles, surfaces, and agents in the environment. This 3D extension is critical to be able to handle the more realistic situations we are interested in studying. The only work of which we are aware that uses 3D visibility during pursuit evasion is an approach using height maps on terrains [5].

In our framework, agents are equipped with a roadmap (used for navigation), a set of capabilities and behaviors which define the actions they will take given their knowledge of the environment. The agents and behaviors are versatile and tunable so that the kind of agent and the behaviors being applied can be adjusted with a set of parameters that define them. This general representation of the problem allows us to use the same approach for pursuit-evasion in very simple environments as well as a range of other scenarios that have received much less attention. We demonstrate our approach on mobile robots and in simulated scenarios involving pursuit-evasion and tag on terrains, in multi-story buildings and in crowds. Movies showing additional results can be found on our webpage: http://parasol.tamu.edu/groups/amatogroup/research/flock/.

II. 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 [1] looks at the problem of pursuit-evasion on a graph in which agents move on edges between nodes in the graph and where the pursuer has the goal of occupying the same node as the evader. The problem has been looked at in polygonal environments [2], with much interest in finding exact bounds on the number of pursuers needed [6]. An approach based on exploring undiscovered portions of the frontier is described in [7].

Roadmap-based pursuit-evasion, where the pursuer and evader share a roadmap and play different versions of the pursuit-evasion game is described in [8]. These games include determining: 1) if the pursuer can 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, a wider range of capture conditions are supported than in most previous approaches.

In [9], [10], [11], the benefits of integrating roadmap-based path planning techniques with flocking techniques were explored. A variety of group behaviors, including exploring and covering, were simulated utilizing an underlying roadmap.

*This research supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266 by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).

This paper builds on that work 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 [12] 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]. One form of the problem ([14]) involves a team of agents attempting to maximize the average number of targets that are observed by at least one of the team members. Agents deal with limited sensing information by building gap navigation trees with gap sensors in [15]. The idea of using probabilistic shadow information spaces for targets which move out of a pursuing agent's field-of-view has also been considered [16]. Other forms of collaboration between searching agents has been considered where agents utilize frontier information [17] with communication to switch between roles [18].

The level of visibility between agents plays a role in what the agents 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 [3] with the field of view being variable in [4]. In [19] a team of agents with limited sensory abilities attempt to capture a single evading agent using sweep formations. Teams of robots with different kinds of vehicles and sensing information is looked at in [20]. The problem of pursuit-evasion on height maps, which restrict agent visibility is described in [5].

Crowds of agents moving in an environment [21], [22], [23] present many interesting challenges for pursuit-evasion, including large numbers of moving obstacles and visibility calculations which include the crowd. Sophisticated, path planning techniques have been created for planning the motion of agents in crowded areas [23], some with specific applications of pursuit-evasion scenarios [24].

III. FRAMEWORK OVERVIEW

Many of the previously proposed approaches make restrictive assumptions. Examples include focusing on polygonal environments, operating 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.

A. 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 $d \geq 0$. The pursuers and evaders may have different levels of environmental knowledge, sensing abilities and capabilities, which affect 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 involve fleeing from the pursuing agents that have been detected and improving on hiding locations when undetected.

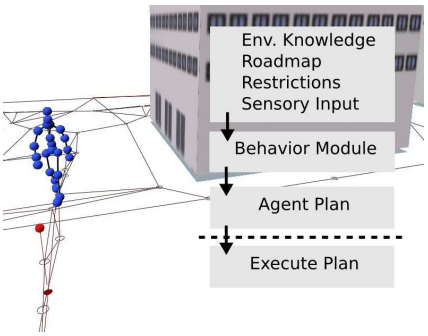


Fig. 1. Agent-Behavior Overview

B. Approach

In an attempt to explore a range of the entire pursuit/evasion spectrum we have developed an 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 of problems and provides the basis for a framework that provides a great deal of flexibility and control to the user over interesting pursuit/evasion games and simulations.

Our approach uses a real-time simulation which consists of the movement of agents (utilizing 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.

Algorithm 1 General Simulation Loop

Input: simulator sim, environment env

1: $groups_{all}$ = sim.getAllGroups()
2: **for** $g \in groups_{all}$ **do**
3: $g \rightarrow$applyBehaviorRule(env)
4: **end for**
5: **for** $g \in groups_{all}$ **do**
6: updateState(g)
7: **end for**
8: ResolveCollisions($groups_{all}$,env)
9: Evaluation

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 of collisions with other agents and the environment as well as handling any interactions that may have occurred between groups. This includes sending any communication messages that may have been generated throughout the behavior process or roadmap re-weighting. 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 environment 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 alters the agent's height component) and affects their 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.

C. 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 [25], 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 [26] or near the medial axis of the environment [27].

Agents navigating in environments consisting of multi-level surfaces (terrains 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 projected 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 time step (which would be very computation-intensive), agents can query a valid roadmap quickly for paths guaranteed to be both available and collision-free.

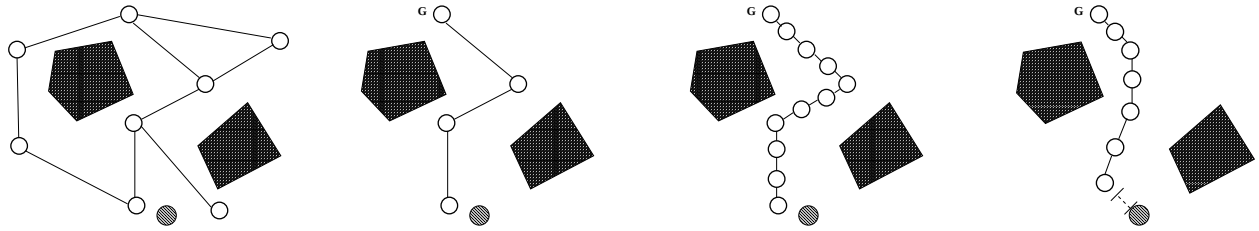
The roadmaps are used to generate paths through the free areas of the environment that will *guide* an agent from some start to goal location. These paths can be adhered to strictly or act as a guide that is optimized based on agent specific criteria. In many of the examples we study, we allow the agents 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 or to modify it to allow for greater clearance. An example of this process is shown in Figure 2. As the agent is navigating through the environment following a path, another parameter determines the distance required for a subgoal to be considered reached. These parameters play an important role in determining how closely an agent adheres to the roadmap and the time it takes an agent to reach a goal in the environment.

D. Visibility between agents

The agent's ability to detect other agents in the environment can be affected by a number of factors. 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 that agents may block one another's view in the environment, i.e., the 3D representation of all agents are 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.

IV. PURSUIT BEHAVIOR

Our general pursuit strategy, shown in Algorithm 2, 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. As an example, the most basic pursuit strategy will chase a target,



(a) Roadmap Example (b) Path Extracted (c) Path Reduced (d) Optimized Path

Fig. 2. This figure illustrates the usage of roadmaps and paths used during navigation. (a) An example environment shown with two polygonal obstacles (shaded), representative roadmap and agent utilizing roadmap (shaded circle). (b) A path extracted from the agent's location to a goal location G. (c) A path reduced to a pre-defined resolution. (d) The reduced path is optimized to allow the agent to reach the goal faster. An agent considers a sub-goal along the path as reach when within a predefined distance. Steps (c) and (d) can be repeated in order to obtain a more optimized path.

once located, in a direct path until it either captures the target or the target is no longer visible.

Algorithm 2 General Pursuing Algorithm

```
1: if a target exists then
2:     Determine which potential target to pursue
3:     Create a pursuit plan
4:     Selection individual pursuit roles
5:     Execute plan and pursue the target
6:     Determine status — terminate on success or failure
7: else
8:     search for target
9: end if
```

A target may be available to an agent through either direct observation or through communication with nearby agents (communication being a capability that may or may not be available among agents). Determining which potential target to pursue involves selecting the target agent from all the available agents which is most likely to result in a successful capture. While we typically use distance as the criteria for selecting a target to pursue, other agent factors could be used such as size, health, or stamina. In creating a pursuit plan, agents may take into account behavioral factors, described further in the enhancements sections. The plan created should take the agent from its current location to a location that will improve its chances of capturing the evading agent. The plan is executed until it is no longer valid (i.e., the target agent is no longer within range of the final goal of the plan or the target is no longer available).

A. Search behaviors

The searching behaviors that an agent uses to locate a target can greatly influence the pursuing agent's effectiveness. We have implemented a number of roadmap-based searching behaviors which allow groups of agents to utilize the roadmap to effectively cover an environment [10]. 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 least visited or search for some random area in the environment. Since the searching aspect is not the focus of this paper, we do not go into detail on these searching behaviors.

B. 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 the success rate of a group of pursuing agents. Pursuing agents can "head off" an evading agent by planning their path to intercept it; 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.

Another enhancement that pursuing agents can be given is the ability to flank an evading agent. Pursuing agents attempt to flank by spreading out behind the evading agent while chasing. This restricts the evaders motions when an evasive action is need (for example near an obstacle or boundary). Our flanking enhancement allows agents in a pursuit group to plan their path to flank positions around a target; this corresponds to selecting individual roles from Algorithm 2.

Enabling communication between pursuing agents can improve a pursuing agent's effectiveness. For example, communication can alert a pursuing agent to the location of an evading agent. Communication between agents can also allow for one agent, the leader, to request other agents to follow that agent when searching the environment. This type of communication allows for potentially more coordination between pursuing agents which is necessary in some pursuit scenarios.

V. EVADING BEHAVIOR

Our general evasion strategy, shown in Algorithm 3, 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 positions 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 location is found that sufficiently improves the score.

Algorithm 3 General Evasion Algorithm

```
1: if Evasive plan is needed then
2:     Determine strategy for evasion
3:     Begin execution of the strategy
4:     Adapt strategy to current situation
5: else
6:     attempt to improve hiding location
7: end if
```

A. Scoring Hiding Locations

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

When an evading agent detects one or more pursuing agents, five criteria are used for scoring hiding locations:

- **Distance to Pursuers (α):** 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 (β):** The direction to a hiding location is computed and scored such that directions away from pursuing agents are preferred.
- **Distance To Boundary (γ):** 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 reduce the likelihood of being trapped by the boundary.
- **Visibility Restricted Surfaces (δ):** The weight of this factor can be determined by how much a hiding location is obscured by surfaces in the environment.
- **Visibility Restricted by Other Agents (ϵ):** 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.

An evasion strategy can be defined to be a weighted set of these factors, $< \alpha, \beta, \gamma, \delta, \epsilon >$ so that evading agents can be tuned to prefer certain kinds of evasion strategies. 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 have a set evasion strategy. As an example, for two evasion strategies, S1 and S2, with all factors being equal except $\gamma_1 \ll \gamma_2$ would result in S2 avoiding the boundary more than S1.

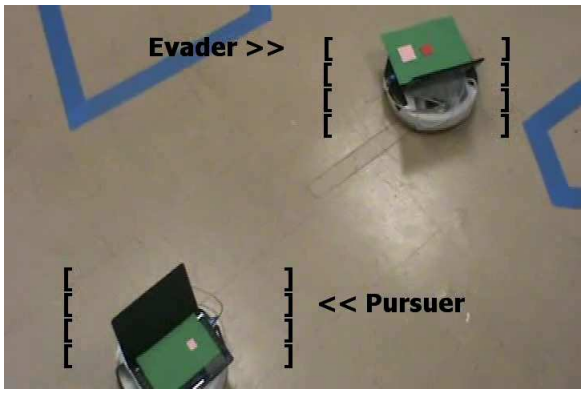


Fig. 3. Real Environment

VI. APPLICATION ON ROBOTS

While our roadmap based approach is able to handle very complex environments, it can also handle environments consisting of actual robots. We have applied our simulated framework on a set of iCreate robots, an example of which is shown in Figure 3. Overhead camera data is used to obtain information about the environment which is sent to simulated sensors (i.e., robot placements and orientations). The simulated sensors process this information to give each agent knowledge about what is known to it in the environment. An agent uses this information in the behavioral process when determining the next actions to take.

The framework for this application makes use of a server, which collects important position and orientation information of each of the iCreates. Although an overhead camera is used the simulation had to be adapted to handle errors never before met in our framework. From here information is sent over a wireless network to the iCreates, which are running the simulation framework. The camera data however is only seen by the virtual sensor suite which will translate the data into useful information for the behaviors to use. Once the behavior framework plans the robot's path, this data is translated to actual translation and rotations for the robots. The iCreate robots were limited to first rotational movement followed by translational movement for simplicity.

The virtual sensor suite contains the most vital sensing capabilities for the behavioral framework. The sensors we decided to simulate were as follows:

- **Localization:** This sensor will update the positional values for the agent. This is the first introduction of error from the vision server, and alters the movement plans of the agents.
- **Compass:** The compass provides orientation information for the agents. This is where the majority of error occurs in our system. The vision server can only approximate the orientation of the robot. This also affects the local movement of the agents.
- **Vision ID:** This collect the ID's of the individual agents, so that the correct information about visibility of the

agent can be calculated. This sensor determines which agents are within the view radius of the agent.

- **Communication:** This is a message passing sensor, which will connect to a separate message forwarding service so that all agents can receive and filter the messages.

The execution of the framework for a simple one-on-one pursuit evasion with the iCreates is shown in the video. Additional examples are provided on our webpages: http://parasol.tamu.edu/groups/amatogroup/research/flock/.

VII. EXPERIMENTS

We present results for pursuit/evasion scenarios that have not received as much attention as traditional problems. These involve more complex environments where the visibility of the agents is restricted by the environment. We also show examples where some interesting interaction takes place between the groups of agents being simulated. In the following, we show results in terrains, crowds, and a multi-level environment. 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 proportion/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.

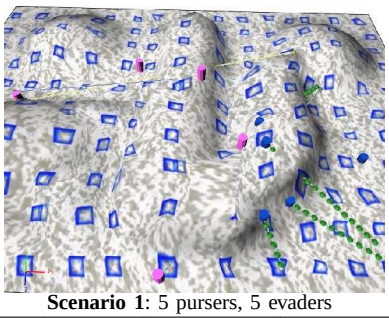
VIII. TERRAIN

The terrain environment, shown in Table I, 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 the first scenario, 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, using a flank behavior and with 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, a slight advantage is given to the evading agents where they are faster and can sense more of the environment than the pursuing agents.

In Table I, for Scenario 1, it can be seen that using only basic pursuit results in fairly poor performance since the evading agents can easily outmaneuver the pursuers. In this table, the proportion of captures is an average of the proportion of captures per run. When pursuing agents attempt to head off the evading agents, the performance improves overall resulting in more captures and lower time to capture. In this case, adding the flank behavior does not improve results over using a heading off behavior. Also, sharing a target among the pursuing agents greatly improves the pursuing agents' effectiveness.

TABLE I

PURSUIT/EVASION ON A TERRAIN WITH HILLS AND VALLEYS OBSCURING VISIBILITY IN THE ENVIRONMENT.



Scenario 1: 5 pursers, 5 evaders

Pursuit Type	Prop. Capt.	Time To Capture	Min/Max Time Chasing	Min/Max Time Hidden	Prop. Time Hidden
basic	0.2	3686.7	(1902,3189)	(1538,3461)	0.5261
heading	0.65	1363.5	(1056,2792)	(409,2556)	0.3044
flank	0.4	1302	(634,2379)	(532,3186)	0.3192
send target	0.8	1028	(1133,2520)	(523,3980)	0.4169

Scenario 2: 4 pursers, 1 evader

Pursuit Type	Prop. Capt.	Time To Capture	Min/Max Time Chasing	Min/Max Time Hidden	Prop. Time Hidden
basic	0.2	2376	(470,937)	(3059)	0.6352
heading	0.2	3680	(413,1116)	(2954)	0.6029
flank	0.4	3018	(541,859)	(3283)	0.7368

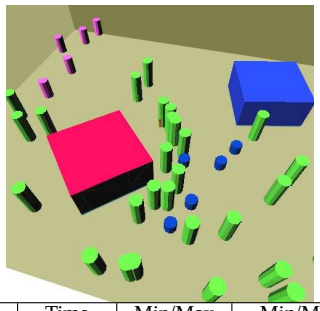
In the second scenario, 4 pursuers attempt to capture a single evading agent. A much greater advantage is given to the evading agent by giving it a faster maximum velocity. Pursuing agents have a maximum velocity of 2, view radius of 80 and height of 6 units. Evading agents have a maximum velocity of 9, view radius of 80 and height of 3. In the table, the proportion of captures refers to an average of successful captures within the time limit over all runs. In this scenario, both sets of agents have similar sensing abilities, being able to detect a large portion of the environment that is not blocked by obstacles/surfaces. Both basic and heading behaviors have fairly low captures with the flank behavior outperforming both. This is an example where coordination and communication between agents is needed given the evading agent's superior maneuverability. In each scenario, utilizing the some effective enhancements allows agents to restrict the number of evasive actions.

IX. CROWD

In an environment with a crowd, the evading agents can hide among crowd members. 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, b) using a flank behavior and c) using a heading off with shared target behavior. The basic pursuit again performs the worst with agents that perform a flank behavior being

TABLE II

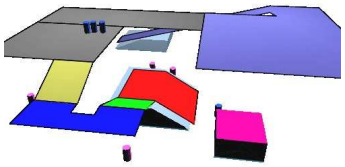
PURSUIT/EVASION IN A CROWD OF OTHER AGENTS (5 PURSUING, 5 EVADING, 30 IN CROWD). PURSUING AGENTS SHOWN IN TOP LEFT CORNER OF IMAGE, EVADING AGENTS SHORTER, BLUE CYLINDERS AT CENTER AND CROWD ARE TALL, GREEN CYLINDERS.



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
flank	4.5	770.5	(144,458)	(911,3959)	0.4418
heading, share target	5	448.4	(126,543)	(418,1744)	0.4104

TABLE III

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

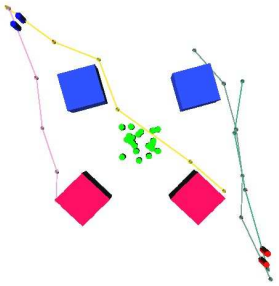
even more effective. As in the previous example, using a heading off behavior along with a shared target performs the best. Both enhanced behaviors result in the much better performance than basic pursuit with the highest number of captures and lower average time to capture. These behaviors among pursuing 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.

X. MULTI-LEVEL ENVIRONMENT

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, in the background in the environment shown in Table III. The other connection consists of multiple surfaces ramping up to the second floor. Two basic pursuit behaviors are tested a basic pursuit and

TABLE IV

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/heading	12.4	12.4	0.2	3606
heading/basic	17.8	6.8	0.4	4780
flank/basic	15.0	10.0	0.0	3794

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.

XI. 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 IV, pursuit behavior comparisons are shown for basic vs. basic pursuit, heading vs. heading pursuit, heading vs. basic, and flank vs. basic pursuit behaviors. We initialize each set of pursuing agents with two pursuers. The numbers shown in Table IV 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 (all agents are caught) is much lower, also showing the effectiveness of this pursuit behavior.

While using a heading off behavior results in the most captures, there are cases where agents are left evading. When agents are equipped with a flank behavior more agents end up performing a flank behavior than basic pursuit. These examples show the flank behavior results in better performance overall. This can be seen in leaving no agents evading and a low time in completion of the game. This is in part due to agents performing a basic pursuit being assisted by the agents performing the flank behavior. While some agents may be flanking an evading agent, basic pursuit agents are able to pursue and take advantage of the flanking agents restricting evading agent's potential evasive actions.

XII. CONCLUSION

This paper describes an agent- and roadmap-based approach to pursuit-evasion which facilitates the study of heuristic pursuit-evasion in interesting scenarios including crowds, terrains, and multi-level environments. The presented results demonstrate this this approach can be applied on mobile robots and in simulated environments which have more complex visibility and reachability constraints than are typically studied in pursuit-evasion games.

Acknowledgements

The authors would like to acknowledge Gavin Guy, Plamen Ivanov, Andrew Logan for their assistance with the experimental testbed.

REFERENCES

[1] T. D. Parsons, "Pursuit-evasion in a graph," in *Theory and Applications of Graphs*, ser. Lecture Notes in Mathematics, vol. 642. Springer, 1978, pp. 426–441.

[2] L. J. Guibas, J. claude Latombe, S. M. Lavalle, D. Lin, and R. Motwani, "Visibility-based pursuit-evasion in a polygonal environment," in *International Journal of Computational Geometry and Applications*. Springer-Verlag, 1997, pp. 17–30.

[3] V. Isler, S. Kannan, and S. Khanna, "Randomized pursuit-evasion with limited visibility," in *Proc. ACM-SIAM Symposium on Discrete Algorithms*, 2004, pp. 1060–1069.

[4] B. P. Gerkey, S. Thrun, and G. Gordon, "Visibility-based pursuit-evasion with limited field of view," *Int. J. Robot. Res.*, vol. 25, no. 4, pp. 299–315, 2006.

[5] 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.

[6] S. M. Lavalle, D. Lin, L. J. Guibas, J. claude Latombe, and R. Motwani, "Finding an unpredictable target in a workspace with obstacles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1997, pp. 737–742.

[7] B. Yamauchi, "Frontier-based exploration using multiple robots," in *International Conference on Autonomous Agents (Agents '98)*, 1998, pp. 47–53.

[8] V. Isler, D. Sun, and S. Sastry, "Roadmap based pursuit-evasion and collision avoidance," in *Proc. Robotics: Sci. Sys. (RSS)*, 2005.

[9] 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)*, Dec 2002, pp. 95–111.

[10] O. Bayazit, J. M. Lien, and N. M. Amato, "Better group behaviors in complex environments using global roadmaps," in *Artif. Life*, Dec 2002, pp. 362–370.

[11] O. Bayazit, J. Lien, and N. M. Amato, "Roadmap-based flocking for complex environments," in *Proc. Pacific Graphics*, Oct 2002, pp. 104–113.

[12] A. Kolling and S. Carpin, "Pursuit-evasion on trees by robot teams," *Trans. Rob.*, vol. 26, no. 1, pp. 32–47, 2010.

[13] A. Kolling and S. Carpin, "Multi robot pursuit evasion without maps," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2010, pp. 3045–3051.

[14] A. Kolling and S. Carpin, "Cooperative observation of multiple moving targets: an algorithm and its formalization," *Int. J. Rob. Res.*, vol. 26, pp. 935–953, September 2007.

[15] B. Tovar, R. Murrieta-Cid, and S. M. LaValle, "Distance-optimal navigation in an unknown environment without sensing distances," *IEEE Transactions on Robotics*, vol. 23, no. 3, pp. 506–518, 2007.

[16] J. Yu and S. M. LaValle, "Probabilistic shadow information spaces," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2010, pp. 3543–3549.

[17] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 476–481.

[18] 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)*, 2010, pp. 3562–3568.

[19] S. Bopardikar, F. Bullo, and J. Hespanha, "Cooperative pursuit with sensing limitations," in *American Control Conference (ACC)*, July 2007, pp. 935–953.

[20] H. Kim, R. Vidal, D. 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*, 2001, pp. 634–639.

[21] C. W. Reynolds, "Flocks, herds, and schools: A distributed behaviroal model," in *Computer Graphics*, 1987, pp. 25–34.

[22] A. Treuille, S. Cooper, and Z. Popovi, "Continuum crowds," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1160–1168, 2006.

[23] J. van den Berg, S. Patil, J. Sewall, D. Manocha, and M. Lin, "Interactive navigation of multiple agents in crowded environments," in *Proc. Symposium on Interactive 3D Graphics and Games - I3D'08*, 2008.

[24] 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.

[25] 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.*, vol. 12, no. 4, pp. 566–580, August 1996.

[26] 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*. Natick, MA: A.K. Peters, 1998, pp. 155–168, proc. Third Workshop on Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.

[27] 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)*, vol. 2, 1999, pp. 1024–1031.