The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Behavior-Based Motion Planning for Group Control

Christopher Vo Joseph F. Harrison Jyh-Ming Lien

Abstract— **Despite the large body of work in both motion planning and multi-agent simulation, little work has focused on the problem of planning motion for groups of robots using external "controller" agents. We call this problem the *group control problem*. This problem is complex because it is highly underactuated, dynamic, and requires multi-agent cooperation.**

In this paper, we present a variety of new motion planning algorithms based on EST, RRT, and PRM methods for shepherds to guide flocks of robots through obstacle-filled environments. We show using simulation on several environments that under certain circumstances, motion planning can find paths that are too complicated for naïve "simulation only" approaches. However, inconsistent results indicate that this problem is still in need of additional study.

I. Introduction

An interesting yet mostly unexplored instance of multi-robot motion planning is the *group control* (or at a larger scale, the *swarm control*) problem. The objective of this problem is to control and navigate a group of agents (such as a crowd of people or a flock of sheep) using external "controller" agents (such as police officers or shepherd dogs). The solution provided by the planner is a sequence of actions for the controller agents that will guide the group to a goal. The group control problem is challenging because it is highly underactuated, involves complex dynamics, and requires multi-agent cooperation.

The group control problem has many important applications in security (e.g., simulation of disaster scenarios and responses [1], [2], [3]); in civil crowd control (e.g., planning evacuation routes for sporting or spectator events [4]); in pollution control (e.g., collecting oil spills [5]); in agriculture (e.g., sheep herding [6]); in transportation safety (e.g., preventing bird strikes [7]); in education and training (e.g., providing immersive museum exhibits and training systems); and entertainment (e.g., interactive games, virtual crowds in cinema).

Despite the importance of the group control problem, methods for creating group control strategies are still poorly understood. A recent study [8] has highlighted the lack of research into effective strategies for crowd control and the catastrophic consequences that may result from flawed strategies. Riots and fatalities due to poorly controlled crowds are a frequent reminder of the need for better strategies. In late 2008, 249 people died and more than 425 were injured at a Hindu temple in India when a bomb threat caused a stampede. Also in 2008, a man in a New York Wal-Mart was trampled to death by a crowd of eager holiday shoppers.

All authors are with Department of Computer Science, George Mason University, 4400 University Drive MSN 4A5, Fairfax, VA 22030 USA, {cvo1, jharri1, jmlien}@cs.gmu.edu

Motion planning algorithms and data structures such as Rapidly Exploring Random Trees (RRTs), Expansive-Spaces Trees (ESTs), and Probabilistic Roadmaps (PRMs) have proven to be elegant and powerful ways to find paths in high-dimensional configuration spaces. They have already been applied successfully to a wide variety of complex scenarios including robots with various kinematic, dynamic, or non-holonomic constraints [9]; manipulation planning [10], [11]; and multi-agent planning [12], [13].

The track record of these probabilistic methods, particularly on high-dimensional configuration spaces, makes them attractive for planning in the group control problem. In this paper, we propose several new motion planners for the group control problem based on RRT, EST, and PRM methods. We test these algorithms on a variety of obstacle-filled environments, and compare them to a "simulation-only" approach which uses only simple behaviors with the help of the medial axis of the workspace. In general, we find that the exploration offered by motion planning sometimes improves the success rate, particularly in environments where the the medial axis passes through narrow passages. However, in cases where there are many viable paths, our planning methods do not as yet show enough improvement to justify the additional computation time required and in some cases are less likely to reach the goal.

II. Related Work

The group motion control problem is one that merges the simulation and modeling of multi-agent dynamics with cooperation, planning, and control. There is a large body of work on the dynamics and modeling of human crowds, but there is still very little work on swarm motion control via agent interactions, and no work has been proposed to systematically study group motion control as a whole.

Works that explore the *crowd control problem* have used a variety of simple approaches such as introducing new objects or agents into the environment. For example, the effect of adding barriers into disaster scenario environments was studied in [2]. Other works have attempted to model the effect of adding agents with various social forces. The effect of adding robots with attractive social forces was examined in [14], and crowd dynamics in the presence of "leader" individuals was modeled in [15].

There are several works in robotics and computer animation related to modeling behaviors such as shepherding. For example, Schultz et al. [16] applied a genetic algorithm to learn rules for a shepherd robot. Its objective was to control the movement of other robots (sheep) that react by moving away from the shepherd. Vaughan et al. [17] simulated and constructed a robot that shepherds a flock of geese in a circular environment. In computer animation, Funge et al. [18] simulated an interesting shepherding behavior in which a T. rex chases raptors out of its territory. Potter et al. [19] studied a herding behavior using three shepherds and a single sheep in a simple environment. None of the aforementioned methods have shown the ability to guide flocks in environments with obstacles.

Swarm control can also be viewed as a cooperative problem. The survey from Parker [20] provides an overview of multi-robot systems. From the perspective of multi robot systems, the task of crowd control requires *inherent cooperation*, in which the success of a robot in the team depends on the actions of other robots. Inherent tasks (such as crowd control) are distinguished from non-inherent tasks (such as covering) in that they cannot easily be decomposed into independent sub-tasks and thus are generally more difficult. Multiple robots may also move in formation [21] to accomplish a given task. In our previous research on shepherding behaviors [22], we observe that formations can be used effectively to control the motion of the flock. Similar observations have also been found in sociological studies of crowd control [23]. Recently, Shell and Matarić [1] have used multiple robots to deploy and assist with the evacuation of pedestrians.

Swarm control is also related to competitive activities such as pursuit and evasion behaviors, and sports such as soccer. A simplified version of the pursuit and evasion problem which considers only one pursuer and one evader has been studied for decades using methods such as game theory [24], co-evolutionary algorithms [25], and neural networks [26] (see survey [27] for detail).

Controlling groups of agents may also be viewed as a type of robotic manipulation problem. Several researchers have attempted to use multiple robots to cooperatively manipulate or move passive objects, for example pushing a disc [28], or a box [29], or kicking a ball [30]. A passive object will move only if external force is applied to it. On the other hand, crowd control attempts to manipulate the motion of active objects which have the ability to move on their own. This usually makes active objects more difficult to control. So far, no methods have been proposed to manipulate multiple active objects using multiple robots.

There is also a significant amount of work in planning for multi-robot (or multi-agent) navigation [31], [32], [13]. More recently, Sung et al. [33] proposed a system that simulates goal-directed crowds using motion graph and constraint solver for several types of constraints. Lau and Kuffner [34] proposed a behavior planning algorithm that performs a global search of the Finite State Machine. Sud et al [35] used first- and second-order Voronoi diagrams and van den Berg et al. [36] used the idea of a "reciprocal velocity obstacle" for real-time multi-agent planning.

III. Preliminaries

A. Shepherd Behaviors

In this section, we define some of the terms and concepts that are used later in the paper. A *shepherd* is an external agent that influences the movement of the flock. A *flock* is a collection of agents that try to steer away from the shepherd. The shepherd's task is to steer the flock to desired locations. In addition to steering the flock, the shepherd attempts to keep the flock in a cohesive *group*, where each member is in visible range of at least one other member of the group. Separation of the flock into multiple groups may be caused by repulsive forces exerted by obstacles, or by shepherds.

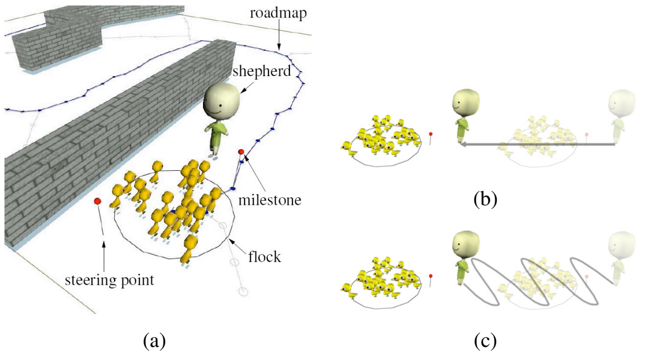


Fig. 1. (a) An environment and associated terms. Steering the flock using (b) a straight line (c) a side-to-side motion.

A *configuration* is the full specification of the position and velocity of flock members and shepherds. Therefore, a group control problem with n flock members and m shepherds in a 2-d workspace will have a configuration space in $4(n+m)$ dimensions. A *roadmap* is an abstract representation of the feasible configuration space in a given environment, given as a directed graph $G = (V, E)$, where each node in V represents a valid configuration, and each directed edge $(p, q) \in E$ denotes that it is possible for the flock to travel from configuration p to configuration q. We use the term *milestone* to denote any intermediate position that the shepherd attempts to *steer* the flock towards, and we use the term *steering point* to denote any position to which the shepherd moves himself in order to influence the movement of the flock; see Figure 1(a).

We define a *shepherd's locomotion* as the manner in which a shepherd moves to control the movement of a flock. The shepherd's locomotion remains an invariant in different shepherding behaviors and dramatically affects the quality of simulation. We divide the shepherd's locomotion into two sub-problems: *approaching* and *steering*. In approaching locomotion, the shepherd attempts to get close to the flock. In steering locomotion, the shepherd attempts to push to flock forward. Two examples of the shepherd's steering locomotion can be seen in Figures 1(b) and (c). We will use the shepherd's locomotion to move the flock forward in the local planner discussed next.

B. Local Planning

In traditional probabilistic motion planning methods, a *local planner* is used to connect pairs of nearby configurations. Examples of traditional local planners include: the straight-line planner (which simply connects configurations through direct interpolation); the rotate-at-s planner [37]; and the A^*

planner. The local planner used in our planning method is a simulation defined by a set of behavior rules for how each of the agents will move.

More specifically, our local planner first finds a path in workspace from the *center* of the flock to the goal position. In our implementation, the path is extracted from the medial axis of the free workspace. (Examples of these workspace medial axes can be found in Fig. 3.) Then, the shepherd pushes the flock along the path by performing a sequence of locomotions determined by on the current state of the flock. As with most traditional local planners, this local planner may fail to lead the flock to a given milestone. For example, the shepherds may not have enough room to push the flock, or a passage may be too narrow for the flock to pass through. The SIMULATE procedure in Algorithm 1 sketches an overview of the local roadmap.

IV. Our Methods

In our motion planning method, the input is a non-toroidal workspace W, an initial configuration q_{init}, and a goal configuration q_{goal}. If a solution is found, it is given by the planner as a path through the feasible configuration space $\mathcal{C}$-free of valid configurations that guides the flock from q_{init} to q_{goal}. These planners incrementally build the roadmap. The connectivity of nodes of the roadmap is determined by the local planner described in III-B.

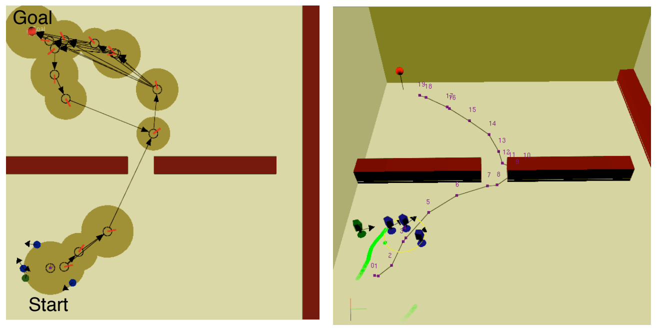
A. RRT-*based planners*

Our RRT-based planner (see Algorithm 1) constructs the roadmap G by repeatedly attempting to extend G towards new randomly generated configurations in W. For RRT, the procedure SELECT-INTERMEDIATE GOAL chooses a random configuration g in the workspace, and the procedure SELECT-NODE-TO-EXPAND chooses the node already in G that is closest to g.

Algorithm 1 Tree-Based Planner

```
procedure TREE-BUILD($q_{init}$)
    $G.init(q_{init})$
    for $k = 1$ to $K$ do
        $g \leftarrow$ SELECT-INTERMEDIATE-GOAL
        $p \leftarrow$ SELECT-NODE-TO-EXPAND
        $r \leftarrow$ SIMULATE($G$,$p$,$g$,$simsteps$)
        if $r = Success$ then
            $G.add\_edge(p, g)$
    return $G$

procedure SIMULATE($G$,$p$,$g$,$simsteps$)
    for $i = 1$ to $simsteps$ do
        for shepherd $s$ in $S$ do
            Use workspace roadmap to find a path to $g$
            Select a milestone $m$ on the path.
            Calculate new target $t_{new}$ for $s$
            Move $s$ towards $t_{new}$
        if flock is close to $g$ then
            return $Success$
```



(a) Meta Graph (b) Path

Fig. 2. (a) Broken T environment with a meta graph. The dark areas in the figure are obstacles. Each node in a meta graph is shown as an oriented disc. (b) A path found by the fuzzy meta-graph planner.

B. EST-*based planners*

Our EST planner is similar to the RRT planner presented in Algorithm 1, with a few key differences. Instead of expanding the tree towards a randomly generated configuration, EST operates by first choosing an existing node p in G to expand based on a probability distribution π_G (in the procedure SELECT-NODE-TO-EXPAND). Then, within a *neighborhood* of p, it selects an intermediate milestone g to expand to (SELECT-INTERMEDIATE-GOAL). A key design decision in the implementation of EST algorithms is the choice of π_G to prevent excessively dense sampling of configurations in the workspace. We have implemented several variations of the EST method which vary only in the selection of π_G:

- BASICEST – The node to expand is selected randomly; that is, π_G is uniformly distributed.
- NAIVEEST – The distribution π_G is weighted so that nodes with fewer neighbors are more likely to be selected over nodes with more neighbors.
- MINEST – The node with the fewest neighbors is chosen every time.

C. PRM-*based planner*

In our PRM based method, we build a *meta graph* containing the set of possible navigation routes in a *fuzzy* way. Each node in a meta graph defines a set of flock configurations (excluding the shepherd positions) that are conforming to the properties of the node. As in the other methods, edges in the meta graph approach are directed. The weight of an edge in a meta graph presents the probability of successful herding from the start node to the end node of the directed edge. An visual depiction of a meta graph is shown in Figure 2. In a similar fashion to fuzzy or lazy PRM approaches [38], [39], paths from the meta graph are extracted and evaluated until a path is found, or until no path can be found to connect the start and the goal configurations in the meta graph.

Each node in a meta graph represents a *meta configuration*. Each meta configuration C is as an oriented disc in a four dimensional space, and is this composed of four values: position ($p_C = (x_C, y_C)$), orientation (θ_C) and radius (r_C). Each meta configuration intuitively defines a set of *conforming* *flocks*. We say a flock is conforming to a meta configuration C if the minimum enclosing circle of the flock is enclosed by C and the angle between the mean velocity of the flock and θ_C is less than a user defined threshold.

We say that a meta configuration is *free* if the disc does not intersect an obstacle. To generate one free meta configuration, we first assign a random position and orientation, whose ranges are fixed.

To assign a random radius, we compute the smallest enclosing disc as follows: First, we consider a group containing N_f flock members where each individual member is enclosed in a radius R_f circle. The compact area of a group is the smallest circle that one can put all group members into. Ideally, shepherds should control the flock so that all members are inside its compact area.

This is an inverse version of the computationally difficult *packing circles in a circle problem* [40]. The objective of this problem is to find the maximum radius of K equally-sized smaller circles that can be packed into a unit circle. Since we do not have to compute the exact value of the radius R_c of the compact area, we approximate it by considering a bounding square of N_f flock members. In this case, the compact area will be the minimum circle enclosing that square. That is, we approximate R_c as:

$$R_c = R_f\sqrt{2N_f} \tag{1}$$

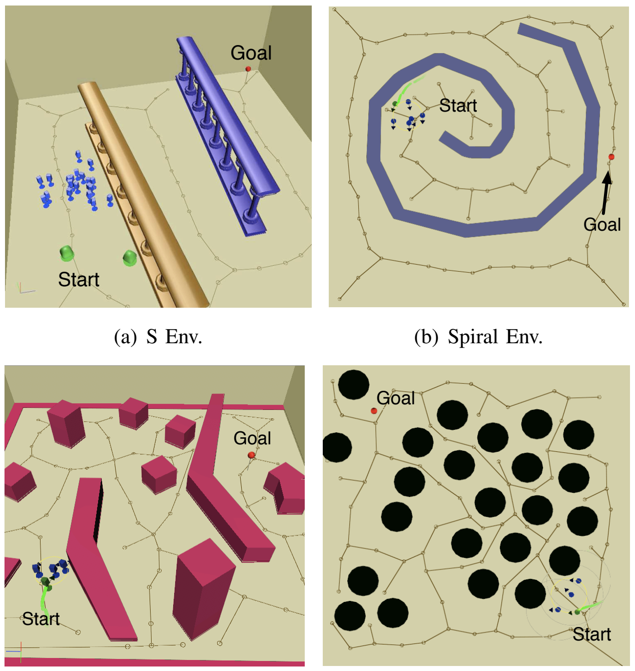
The radius of our meta configuration is set to be a random value between R_c and $3R_c$.

The weight of an edge that connects meta configurations C_1 and C_2 represents the probability that a flock conforming to C_1 can be guided by the shepherds so that the flock is conforming to C_2. This can be estimated in several ways. For example, we can estimate this probability by generating a set of random flocks conforming to C_1, and, for each flock, we find a path to p_{C_2} using one of the tree-based planners described above. While this approach is accurate, it is inefficient due to the requirement of running many (usually thousands or even tens of thousands) of simulations. The approach we use estimates the probability by using the differences in position and orientation between C_1 and C_2. The distance is defined as the follows:

$$\text{dist}(C_1, C_2) = \begin{cases} \infty & \text{if } \theta_{C_1,V} \geq \tau, \\ \infty & \text{if } \theta_{C_2,V} \geq \tau, \\ \frac{|\theta_{C_1} - \theta_{C_2}|}{\pi} + \frac{\text{dist}(p_{C_1}, p_{C_2})}{D} & \text{otherwise,} \end{cases}$$

where $\theta_{C_i,V}$ is the angle between V and C_i's facing direction, $V = p_{C_2} - p_{C_1}$, $\text{dist}(p_{C_1}, p_{C_2})$ is the Euclidean distance between p_{C_1} and p_{C_2} and D is the diagonal distance of the bounding box. In each iteration of the query phase, a path is extracted from the meta graph. To check the feasibility of a consecutive pair of meta configurations in the path, we use simulation to determine if the final configuration of the flock conforms to the end meta configuration.

Note that this meta-graph planner is not probabilistically complete. That is, there are some situations where our planner cannot find the answer even if the path exists – such as in problems that require the flock to form a long line. This is because we confine our radius of our meta configurations and disallow the discs from the meta configurations from intersecting workspace obstacles.



(a) S Env. (b) Spiral Env.

(c) Maze 1 Env. (d) Maze 2 Env.

Fig. 3. Environments used for experiments. The workspace medial axes are shown in all figures. We use these medial axes as the workspace roadmap for local planning.

V. Experimental Results

We tested the proposed motion planners on the five environments shown in Fig. 2 and Fig 3. In the Spiral environment (Fig. 2), the start configuration is at the center and the goal is outside the outer most wall of a spiral-shaped obstacle. In the S environment (Fig. 3(a)), the start and end configurations are at the ends of an S-shaped passage. In the Broken T environment (Fig. 3(b)), the free space is separated by three bars, with the middle bar equidistant to the bars on the side. We also tested the planners on two maze environments. The first maze (Fig. 3(c)) is similar to the S environment with additional road blocks in the S-shaped passage. The second maze (Fig. 3(d)) is composed of several cylinders – a larger cluster at the upper-right corner and a smaller cluster at the lower-left corner.

We compared our planners to a *simulation-only* approach, which only uses the simulator to steer the flock with the help from the medial axis of the workspace. This simulation-only approach is the same method used in [22], [41]. In our test suite, we have 12 variants of EST-based planners: MinEst, NaiveEst, and BasicEst with neighborhood sizes of 3, 5, 7 and 9. In our test suite we also have one Rrt-based planner, and one fuzzy meta-graph planner.

In Table I, we show success rates for each of the algorithms. Each success rate in the table is an average over 50

TABLE I

PROPORTION OF SUCCESSFUL RUNS.

Method	S	Broken T	Spiral	Maze 1	Maze 2
Simulation Only	0.72	0.58	0.22	0.58	**0.82**
MinEST (nhood=3)	0.44	0.48	0.44	**0.66**	0.68
MinEST (nhood=5)	0.48	0.64	0.12	0.38	0.52
MinEST (nhood=7)	0.5	0.56	0.08	0.22	0.4
MinEST (nhood=9)	0.6	0.48	0	0.2	0.3
NaiveEST (nhood=3)	0.26	0.46	0	0	0.18
NaiveEST (nhood=5)	0.4	0.58	0	0.02	0.12
NaiveEST (nhood=7)	0.44	0.82	0	0.02	0.28
NaiveEST (nhood=9)	0.56	0.84	0	0.02	0.32
BasicEST (nhood=3)	0.24	0.36	0	0.02	0.22
BasicEST (nhood=5)	0.46	0.62	0	0.04	0.32
BasicEST (nhood=7)	0.48	0.84	0	0.02	0.24
BasicEST (nhood=9)	0.7	0.84	0	0.08	0.28
rbrmRRT	0.72	0.3	0	0.02	0.06
dbrmRRT	0.74	0.56	0	0.06	0.22
Fuzzy Meta-Graph	**0.96**	0.68	**0.68**	0.1	0.24

runs, and all comparisons are statistically significant at the 95% confidence level.

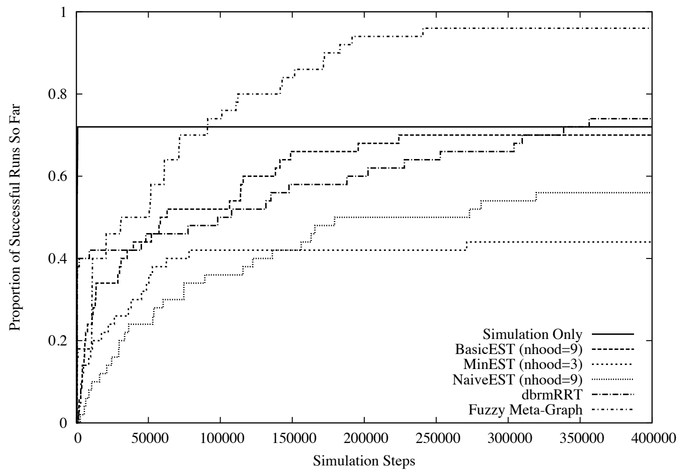
We find that planners do well in the Broken T, S and Spiral environments. We also find that planners do poorly on the maze environments. We believe the reason for this is that we do not allow the planners to search for long enough. For example, the fuzzy meta-graph planner will need to exhaustively check all the shorter but more difficult paths before it can try the longer path with larger clearance. This is clearly shown in the S and maze 1 environments. The main difference between maze 1 and S environment are the introduction of islands in the S shaped passage.

In the S environment, the fuzzy meta-graph planner clearly out-performs simulation-only. In the maze 1 environment, the performance of simulation methods degraded only slightly, whereas the success rate of the fuzzy meta-graph planner drops significantly.

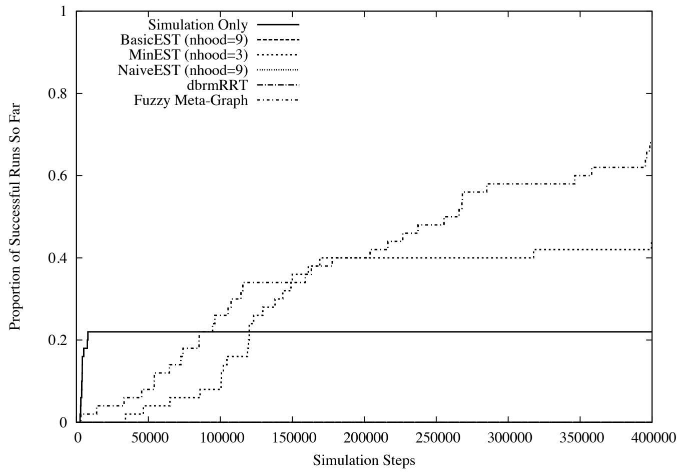
One important observation we had is that motion planning helps improve the success rate, albeit slowly. In Fig. 4, we show the success rate of the planning approaches on a sample of initial configurations. This plot confirms the hypothesis that simulation-only approaches may often get stuck, whereas many of the planning approaches eventually find solutions given a larger simstep budget. This is particularly clear in the S and spiral environments as shown in Fig. 4(a) and Fig. 4(b) . The planners have lower success rates before 80K simulation steps but end at much a higher success rate (60% vs. 18%). Even in the maze environments, in which the planners perform poorly, in Fig. 4(c), we can still see the continuing improvement throughout the entire experiment.

VI. CONCLUSION

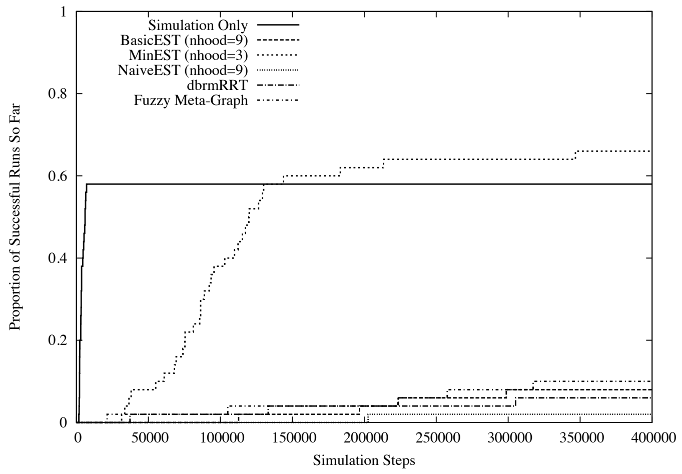
The combination of motion planning techniques and behavior-based simulation allows us to study the effects of planning on domains such as group and swarm control. Using simple extensions to existing EST and RRT methods, we have demonstrated that planning can potentially be beneficial to explore more solutions in scenarios where simulation alone is not adequate. However, there are still cases (such as environments cluttered with islands) where our planning techniques perform poorly, and more research is needed to understand and address those issues.



(a) S Env.



(b) Spiral env.



(c) Maze 1 env.

Fig. 4. Plots showing the proportion of successful runs so far versus the number of simulation steps. Note the plateau for the simulation-only method (the bold black line) which shows that for many of the samples, simulation-only gets stuck and does not make progress.

One avenue for future study would be to apply different metrics for evaluating the quality of a planned path. These metrics may include the number of times the flock separates, or the amount of time spent shepherding.

Acknowledgment

This work has benefited from the contributions of Stephen Donnelly, who created the experimental environments in our test suite. We also thank Brian Hrolenok, Sean Luke, and Keith Sullivan for their valuable input and discussion.

References

[1] D. A. Shell and M. J. Matarić, "Directional audio beacon deployment: an assistive multi-robot application," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2004, pp. 2588–2594.

[2] M. Brenner, N. Wijermans, T. Nussle, and B. de Boer, "Simulating and controlling civilian crowds in robocup rescue," in *Proceedings of RoboCup 2005: Robot Soccer World Cup IX*, 2005.

[3] R. L. Hughes, "A continuum theory for the flow of pedestrians," *Transportation Research Part B: Methodological*, vol. 36, no. 6, Jul 2002.

[4] ——, "The flow of human crowds," *Annual Review of Fluid Mechanics*, vol. 35, no. 1, p. 169, 2003.

[5] M. F. Fingas, *The basics of oil spill cleanup*, 2nd ed. Lewis Publishers, 2001.

[6] A. C. Schultz and W. Adams, "Continuous localization using evidence grids," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 4, 1998, pp. 2833–2839.

[7] 2002, endangered Wildlife Trust. EWT airport safety project. http://www.ewt.org.za/.

[8] J. M. Kenny, C. McPhail, D. N. Farrer, D. Odenthal, S. Heal, J. Taylor, S. Ijames, and P. Waddington, "Crowd behavior, crowd control, and the use of non-lethal weapons," Penn State Applied Research Laboratory, Tech. Rep. A274644, January 2001.

[9] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots under obstacle and dynamic balance constraints," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2001, pp. 692–698.

[10] K. Miyazawa, Y. Maeda, and T. Arai, "Planning of graspless manipulation based on rapidly-exploring random trees," in *(ISATP 2005) The 6th IEEE International Symposium on Assembly and Task Planning From Nano to Macro Assembly and Manufacturing 2005*, 2005, p. 7.

[11] J. C. Thierry Siméon, Jean-Paul Laumond and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *The International Journal of Robotics Research*, vol. 23, no. 7-8, p. 729, 2004.

[12] F. Morini, B. Yersin, J. Maim, and D. Thalmann, "Real-time scalable motion planning for crowds," in *Proceedings of the International Conference on Cyberworlds*, Hannover, Germany, 2007, pp. 24–26.

[13] A. Kamphuis and M. H. Overmars, "Finding paths for coherent groups using clearance," in *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. New York, NY, USA: ACM Press, 2004, pp. 19–28.

[14] J. Kirkland and A. Maciejewski, "A simulation of attempts to influence crowd dynamics," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, 2003, pp. 4328–4333.

[15] F. Aubé and R. Shield, "Modeling the effect of leadership on crowd flow dynamics." in *ACRI*, ser. Lecture Notes in Computer Science, P. M. A. Sloot, B. Chopard, and A. G. Hoekstra, Eds., vol. 3305. Springer, 2004, pp. 601–621.

[16] A. C. Schultz, J. J. Grefenstette, and W. Adams, "Robo-shepherd: Learning complex robotic behaviors," in *In Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*. ASME Press, 1996, pp. 763–768.

[17] R. T. Vaughan, N. Sumpter, J. Henderson, A. Frost, and S. Cameron, "Experiments in automatic flock control," *J. Robot. and Autonom. Sys.*, vol. 31, pp. 109–117, 2000.

[18] J. Funge, X. Tu, and D. Terzopoulos, "Cognitive modeling: Knowledge, reasoning and planning for intelligent characters," in *Computer Graphics*, 1999, pp. 29–38.

[19] M. A. Potter, L. Meeden, and A. C. Schultz, "Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists," in *IJCAI*, December 2001, pp. 1337–1343.

[20] L. E. Parker, "Current research in multi-robot systems," *Journal of Artificial Life and Robotics*, vol. 7, 2003.

[21] T. Balch and R. Arkin, "Behavior-based formation control for multi-robot teams," *IEEE Trans. Robot. Automat.*, vol. 14, no. 6, pp. 926–939, 1998.

[22] J.-M. Lien, O. B. Bayazit, R.-T. Sowell, S. Rodriguez, and N. M. Amato, "Shepherding behaviors," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, April 2004, pp. 4159–4164.

[23] R. Applegate, *Riot control: materiel and techniques*. Stackpole Books, 1969.

[24] R. Isaacs, *Differential games: A mathematical theory with applications to warfare and pursuit and optimization*. John Wiley, 1965.

[25] C. W. Reynolds, "Competition, coevolution and the game of tag," 1994. [Online]. Available: http://www.red3d.com/cwr/papers/1994/alife4.html

[26] D. Cliff and G. F. Miller, "Co-evolution of pursuit and evasion II: Simulation methods and results," in *From animals to animats 4*, P. Maes, M. J. Mataric, J.-A. Meyer, J. B. Pollack, and S. W. Wilson, Eds. Cambridge, MA: MIT Press, 1996, pp. 506–515.

[27] G. Miller and D. Cliff, "Co-evolution of pursuit and evasion I: Biological and game-theoretic foundations," School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK, Tech. Rep. CSRP311, 1994.

[28] D. N. A. van der Stappen and M. Overmars, "Pushing a disk using compliance," *IEEE Trans. Robot. Automat.*, vol. 23, no. 3, pp. 431–442, 2007.

[29] A. Yamashita, T. Arai, J. Ota, and H. Asama, "Motion planning of multiple mobile robots for cooperative manipulation and transportation," *IEEE Trans. Robot. Automat.*, vol. 19, no. 2, pp. 223–237, 2003.

[30] P. Stone and M. Veloso, "A layered approach to learning client behaviors in the RoboCup soccer server," *Applied Artificial Intelligence*, vol. 12, pp. 165–188, 1998.

[31] J.-C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.

[32] T.-Y. Li and H.-C. Chou, "Motion planning for a crowd of robots," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, September 2003, pp. 4215–4221.

[33] M. Sung, L. Kovar, and M. Gleicher, "Fast and accurate goal-directed motion synthesis for crowds," in *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. New York, NY, USA: ACM Press, 2005, pp. 291–300.

[34] M. Lau and J. J. Kuffner, "Behavior planning for character animation," in *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. New York, NY, USA: ACM Press, 2005, pp. 271–280.

[35] A. Sud, E. Andersen, S. Curtis, and D. M. Ming Lin, "Real-time path planning for virtual agents in dynamic environments," in *2007 IEEE Virtual Reality Conference*, 2007, pp. 91–98.

[36] J. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 1928–1935, May 2008.

[37] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, "Choosing good distance metrics and local planners for probabilistic roadmap methods," *IEEE Trans. Robot. Automat.*, vol. 16, no. 4, pp. 442–447, August 2000.

[38] C. L. Nielsen and L. E. Kavraki, "A two level fuzzy PRM for manipulation planning," *IEEE/RSJ International Conference on Intelligent Robotics and Systems*, pp. 1716–1722, 2000.

[39] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 521–528.

[40] R. Graham, B. Luboachevsky, K. Nurmela, and P. Ostergard, "Dense packings of congruent circles in a circle," *Discrete Mat.*, vol. 181, pp. 139–154, 1998.

[41] J.-M. Lien, S. Rodriguez, J.-P. Malric, and N. M. Amato, "Shepherding behaviors with multiple shepherds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, April 2005, pp. 3413–3418.