Department of Computer Science
George Mason University
Technical Report Series

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/
703-993-1530

Scalable and Robust Shepherding Svia Deformable Shapes

Joseph F. Harrison jharri1@cs.gmu.edu
Christopher Vo cvo1@cs.gmu.edu
Jyh-Ming Lien lien@cs.gmu.edu

Technical Report GMU-CS-TR-2010-4

Abstract

In this paper, we present a new motion planning strategy for *shepherding* in environments containing obstacles. This instance of the *group motion control problem* is applicable to a wide variety of real life scenarios, such as animal herding, civil crowd control, and oil-spill cleanup. However, the problem is challenging in terms of scalability and robustness because it is dynamic, highly underactuated, and involves multi-agent coordination. Our previous work showed that high-level probabilistic motion planning algorithms combined with simple shepherding behaviors can be beneficial in situations where low-level behaviors alone are insufficient. However, inconsistent results suggested a need for a method that performs well across a wider range of environments. In this paper, we present a new method, called DEFORM, in which shepherds view the flock as an abstracted deformable shape. We show that our method is more robust than our previous approach and that it scales more effectively to larger teams of shepherds and larger flocks. We also show DEFORM to be surprisingly robust despite increasing randomness in the motion of the flock.

1 Introduction

Group motion control is the problem of moving a group of agents in coordination. One instance of this problem is *shepherding*, in which the objective is to herd a group of agents (e.g., a flock of sheep, crowd of people, etc.) using one or more “shepherd” agents (e.g., shepherds, riot police, etc.). The objective of shepherding is typically to guide the group to a goal, though other variants exist (e.g., escorting or protection, in which a “parent” tries to maintain separation between their “child” and a “stranger” [1]). A solution to the shepherding problem is typically given as a sequence of movements the shepherds can perform to guide the flock to the goal. This sequence may be the output of a high-level motion planner, the result of agent-based behaviors working independently or in coordination to achieve high-level commands, or some combination thereof.

Shepherding is applicable to a wide variety of real life scenarios, such as animal herding [2, 3], civil crowd control [4, 5], and micromanipulation [6]. Shepherding in these scenarios is very challenging (in terms of *scalability* and *robustness*) since it involves the underactuated control of a large number of dynamic agents whose trajectories may be unpre-

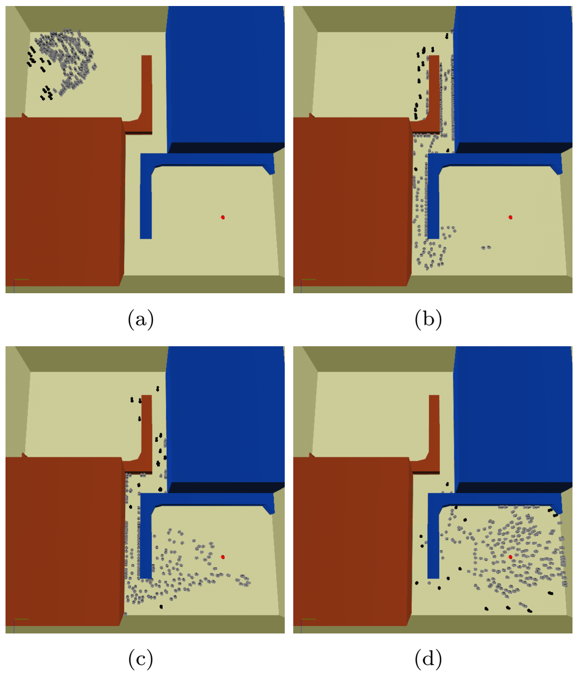

(a) (b) (c) (d)

Figure 1: A team of 15 shepherds successfully herds a flock with 200 members from the starting region in the top-left corner to the goal in the lower-right. Along the way, the flock must be coaxed through a narrow corridor. The flock are drawn in gray and the shepherds are drawn in black.

dictable. While several works in robotics and computer animation have modeled the shepherding problem, none of them addresses shepherding for large groups (with tens to hundreds of members), shepherding in environments with obstacles, or how to handle uncertainty in the motion of the group.

Efficient group representation is necessary for any algorithm to be scalable and robust to uncertainty. Some naïve methods such as [7] and [8] represent the flock as individual agents which severely limits their scalability to large flocks. In this paper we present a simple approach called Deform which represents the flock as a *deformable blob* that can split and merge. This representation is more efficient than individual-based representations and approximates the flock more accurately than simple representations such as discs or bounding boxes [9]. We will define the deformable blob and describe the strategies to manipulate the deformable blob in Section 4.

Despite its simplicity, Deform broadly outperforms our previous method (Magb [9]), especially with larger flocks. Fig. 1 shows an example simulation where 15 shepherds successfully move a flock of 200 agents through a narrow passage to reach the goal. Deform also shows impressive robustness vs. Magb despite increasing randomness in flock behavior. Detailed experimental results will be presented in Section 5.

2 Related Work

We are interested in controlling large groups of active agents by providing external stimuli. There are several works along the vein of understanding the emergent behavior of simulated crowds when significant changes are made to the environment, such as adding movement barriers or additional agents. For example, Brenner et al. studied the effect of adding barriers to influence the movement of crowds in disaster scenarios for RoboCup [4]. Schubert and Suzić [5] used a genetic algorithm with agent simulation to determine optimal barrier deployments for controlling rioting crowds. Some other works have modeled the effect of adding agents with attractive or repulsive social forces [10] and agents with different roles (such as “leader” agents [11]). Yeh et al. also experimented with composite agents [1] which can exhibit different behaviors such as “guidance”, using proxy agents as temporary obstacles.

Some experiments have also been performed to understand how to control living organisms using robotic agents. For example, Halloy et al. [12] showed that robotic agents can influence the collective behavior of a group of cockroaches through local interactions. Ogawa et al. [6] demonstrated the ability to track and manipulate Paramecium caudatum cells in a chamber using an electrical field. Some experiments have herded cows using “smart collars” which act as virtual fences [2, 3]. When a cow apprapoaches virtual fence, the collar produces a noise which causes the cow to move away.

Explicit modeling of shepherding behaviors has also been studied in robotics and computer animation. In robotics, Schultz et al. [7] used a genetic algorithm to learn behaviors for a shepherding robot. Vaughan et al. [8] simulated and constructed a robot that shepherds a flock of geese in a circular environment. In computer animation, Funge et al. [13] simulated an interesting shepherd-like behavior in which a T. Rex chases raptors out of its territory. Potter et al. [14] studied a herding behavior using three shepherds and a single sheep in a simple environment. However, none of the aforementioned methods deals with large flocks or obstacle-filled environments.

In most existing approaches to shepherding, a bounding circle is used to model the flock. This is sufficient for relatively sparse workspaces but is excessively restrictive in environments with many obstacles or narrow corridors, or when the flock size is large. Using deformable object to model a coherent group is not new. Notably, Kamphuis and Overmars [15] used a hinged-box to represent a group of agents for navigating through an obstacle-filled environment. However, the deformable object in our work is modeled as a blob that can split and merge, and, to the best of our knowledge, our work is the first using deformable representation for group motion control.

When multiple shepherds work together to move a flock, their movements must be coordinated to be effective. One common approach is to move the shepherds in formation [16]. Similar formations are commonly used by police and military personnel to disperse, contain, or block crowds and prevent riots [17]. In our own earlier work [18] we showed that shepherd formations can be used to effectively control a flock.

In our most recent work on shepherding, [9], we presented a behavior-based shepherding method which uses the medial axis of the workspace as a roadmap, selecting intermediate milestones on its way to the goal. There, we referred to the method as "simulation only." In this paper, we use this method as a baseline for comparison, and for clarity, we refer to it hereafter as the "Medial Axis Graph-Based" (MAGB) approach.

Our work in [9] also used sample-based methods (RRT [19], EST [20], META-GRAPH) to find a sequence of intermediate milestones through free-space (not restricted to the medial-axis), and connected them with low-level shepherding behaviors. We found these approaches more effective than MAGB for certain types of environments (e.g., those requiring multiple changes of direction toward and away from the goal), but less effective on environments with multiple paths to the goal. The planning-based methods were also much less efficient, and when held to a fixed budget of simulation steps, were often unable to reach the goal.

3 Preliminaries

3.1 Definitions

A *flock* is a group of agents that move with some level of cohesion through the environment and away from *shepherds* which attempt to guide the flock toward a goal. We use F to denote the flock and S to refer collectively to the shepherds.

The configuration of an agent (shepherd or flock member) is represented by its position and velocity. Therefore, a group control problem with n flock members and m shepherds in 2-d will have a configuration space C in $4(n+m)$ dimensions. We refer to the joint configuration of S at time step t as $C_S(t)$ and F as $C_F(t)$. To simplify our notation, we will drop t whenever it does not cause any ambiguity. A *valid* configuration is one in which no agents are in collision with obstacles or other agents. Because n is usually much larger than m, the problem of shepherding is highly underactuated. Since this work focuses on controlling large

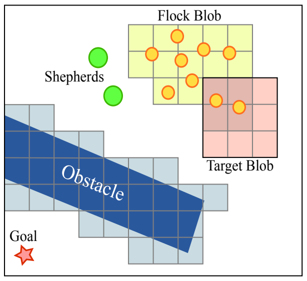


Figure 2: We discretize the workspace to create an occupancy grid of obstacles and free space. The flock blob is the set of grid cells occupied by the flock; the target blob is the where the shepherds would like to move the flock en route to the goal.

flocks ($n > 10$), C is usually very high dimensional. A new representation, such as DEFORM, that can significantly reduce the dimensionality while still accurately approximate C is needed.

We use the term *target* to denote an intermediate configuration toward which the shepherds attempt to steer the flock, and *steering point* to denote a point toward which a shepherd moves itself in order to do the steering.

The term *blob* in this paper refers to a subset of cells in the discretized grid of the environment. The *flock blob* is the area currently occupied by the flock and is denoted B_F. The target area to which the shepherds try to guide the flock is called the *target blob* and denoted B_T. The concept of blobs should be taken generally, as DEFORM may also be implemented with polygons, contours, alpha-shapes, etc. While blobs are often contiguous, they do not need to be.

We define the group-control problem as follows: Given $C_S(0)$ and $B_F(0)$, find a sequence of valid configurations $C_S(t)$ for $0 < t \leq 1$ so that all $B_F(t)$ are valid and $B_F(1) \in GC$, where GC is a set of user specified goal configurations.

3.2 Flock Behavior

The motion of our flock is similar to Reynolds' *Boids* model [21], in which each agent in the simulation steers itself according to three simple rules: *separation*, steer away to avoid colliding with other agents, *alignment*, steer toward the average heading of neighbors, and *cohesion*, steer toward the average position of neighbors. The *neighbors* of an agent are those that are located within its view, which is a fan-shaped region in front of the agent. The fan is defined by a viewing angle and distance. In our work, we consider agents with a viewing angle of 360° so their view depends only on proximity. We add the following rules to determine the motion of our flock: *avoidance* (steering away from nearby shepherds), *damping* (reducing speed over time when other forces are absent), and *entropy* (adjusting the direction randomly, defined in Eq. 1, and is usually set to zero).

4 Our Method: DEFORM

In this section, we describe DEFORM in detail and the work performed beforehand.

4.1 Overview of the Algorithm

DEFORM attempts to move the shepherds such that they guide the flock from the start configuration to the goal g. It does so by continually updating the flock blob (B_F), target blob (B_T), and steering points (s_{steer}). Algorithm 1 outlines the steps necessary.

Algorithm 1 DEFORM(F, S, g)

```
Input: $F$ (flock), $S$ (shepherds), and $g$ (goal)
while $g$ is not reached do
    Compute the flock blob $B_F(t)$ of $F$ at time $t$
    Determine the next target blob $B_T(t)$
    Find $s_{steer} \in C_S$ to morph $B_F$ toward $B_T$
    Assign $s_{steer}$ to $S$
    Update simulation
```

4.2 Computing the Grid

We discretize the polygon-based environment to create an occupancy grid with cells as wide as the diameter of a flock member. We mark the cells as either *free* or *in-collision* depending on whether they contain any part of an obstacle (see Fig. 2). For later use, we calculate a gradient outward from the goal, which provides the geodesic distance from each cell to the goal. Advantages of using grid-based representation also include efficiency and applicability to video games and mobile robots, which often store the environmental data in bitmaps.

4.3 Flock Blobs (B_F)

Given a flock configuration C_F, the corresponding flock blob B_F is the set of all grid cells occupied by members of the flock (see Fig. 3(a)). Since the grid size matches the diameter of the flock members, a one-member flock would have a flock blob with as few as 1 or as many as 4 cells. A large, tightly packed

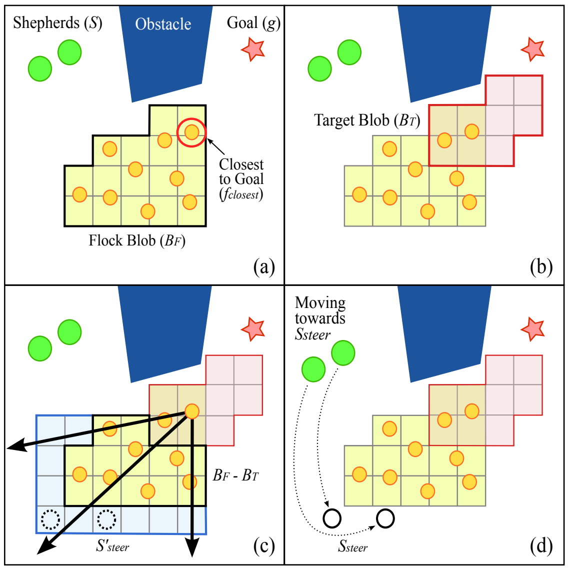


Figure 3: The process of choosing steering points: (a) calculate the flock blob and $f_{closest}$, (b) calculate the target blob, (c) take the boolean difference between flock blob and target blob and calculate potential steering points, (d) select steering points on the opposite side from the goal.

flock can fit within a blob with fewer cells than the size of the flock.

Note that a flock blob may contain multiple non-contiguous parts if the flock has split.

4.4 Target Blobs (B_T)

Given the current flock blob B_F, the target blob B_T is calculated as follows. First, let $f_{closest}$ be the member of the flock closest (in geodesic distance) to the goal. Next, grow B_T using free cells, starting with the 8-connected set and expanding in concentric rings around the cell containing $f_{closest}$. Stop when B_T contains as many cells as there are members of the flock. See Fig. 3(b) for an example.

This approach maximizes the thickness of the target blob, though different metrics could be chosen instead. We discuss some of them in Section 6.

4.5 Shepherds' Steering Points (s_{steer})

Given a flock blob B_F and a target blob B_T, we compute the set of potential steering points s'_{steer} and its subset $s_{steer} \subset s'_{steer}$ which get assigned to the shepherds. We begin by computing the boolean difference between B_F and B_T. Next, we select the points for s'_{steer}. We say a cell p is in s'_{steer} if (1) p is neighboring (using 8-connectivity) to at least a cell $c \in B_F - B_T$, (2) $p \notin (B_F \cup B_T)$, and (3) the geodesic distance from p to B_T is greater than from c to B_T.

Next we choose which n points in s'_{steer} will be assigned to the n shepherds. If s'_{steer} contains fewer than n points, each will be given to a shepherd and the remaining shepherds will stay stationary. If there are more than n points in s'_{steer}, we partition s'_{steer} into n pie wedges centered at $f_{closest}$. In each pie wedge, the point in s'_{steer} farthest from the global goal is selected as a steering point. Fig. 3(c) shows an example.

After s_{steer} is computed, the steering points are assigned to the shepherds by forming a bipartite graph whose nodes are the points in s_{steer} and the shepherd positions, and whose edge weights are geodesic distances between them. The steering point assignment is then solved using a bipartite matching algorithm.

Once the steering points are matched to shepherds, the shepherds move toward their steering points during the next simulation step, as shown in Fig. 3(d).

5 Experiments

We used simulation to test the performance of DEFORM and compare it to our previous results using MAGB [18]. Each simulation begins with a starting and goal region for the flock. The flock is randomly scattered throughout the starting region and the shepherds are randomly placed around the perimeter of the flock. Providing different seeds for the random number generator gives us different initial conditions for each run. The shepherds and flock are modeled as 2-d discs for the collision detection.

We compared DEFORM against MAGB across 6 different test environments, shown in Fig. 4. For this paper, each experimental sample consists of 30 runs, and claims of statistical significance are based on 95% confidence. Despite the simplicity of Deform, in our experiments, we have observed that Deform broadly outperforms Magb in terms of scalability—the ability to steadily maintain high success rates despite an increasing simulation sizes (see details in Section 5.2). The Deform method also shows impressive robustness vs. Magb in terms of maintaining 100% success rates despite increasing randomness in agent behavior (see details in Section 5.3).

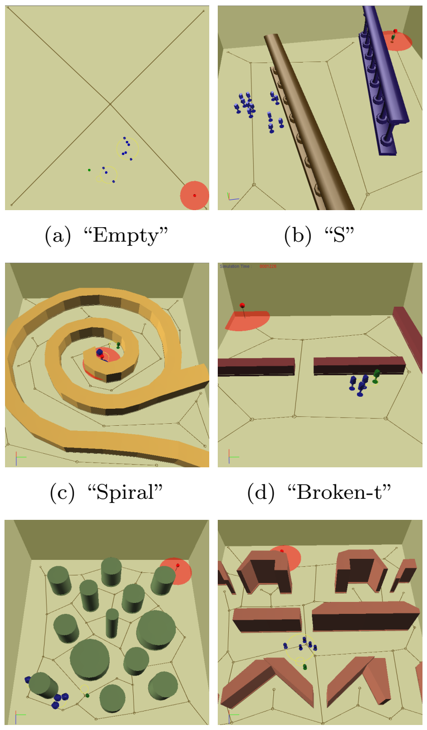

(a) “Empty” (b) “S” (c) “Spiral” (d) “Broken-t” (e) “Pillars” (f) “Jagged”

Figure 4: Environments used in our experiments.

5.1 Environments

We created several different environments to highlight different shepherding challenges (see Fig. 4). The “Empty” environment allows us to evaluate how well a given shepherding approach can move the flock in the absence of obstacles. While it may seem trivially easy, it can be more difficult to move a flock through an open area than a down a narrow corridor. The “S” environment demonstrates a situation with simple local minima which require a shepherd to push the flock away from the goal temporarily. This environment is relatively easy for a motion planning algorithm, compared to the others. The “Spiral” environment requires repeated motions that move the flock closer to, then further from, the goal. This map is particularly difficult for sampling-based planning methods as it requires samples in each of several specific regions before it will be able to connect a path. The “Broken-t” environment presents a narrow passage in the middle of the map as well as another along the wall. The “Pillars” environment offers many different paths to the goal, which can be difficult for tree-based planning methods. The final environment, called “Jagged”, also offers many different paths to the goal, but includes several areas where members of the flock may get stuck.

5.2 Experimental Results: Scalability

We presume that increasing the size of the flock should impact both the effectiveness and performance of the shepherding algorithm, and that more shepherds will be needed to move larger flocks to the goals in a timely manner. The *scalability* of these behaviors is defined by the ability for them to effectively herd increasingly large flocks to the goal within the given simulation time budget.

To test our hypotheses, we ran experiments with varying number of shepherds (2 and 4) and varying flock sizes (20 and 40) on each of the 6 test environments. The results of these experiments are shown in Figure 5. In this figure, note that across most environments, as the simulation size increases from a 2/20 to 4/40, Magb significantly degrades in success rate, while Deform shows no significant degradation in success rate as the simulation size increases. To further investigate the scalability in the finer resolutions, we ran experiments with varying number of shepherds (1, 2, 3, and 4) and varying flock sizes (5, 10, 15, 20, and 25) on the “Broken-t” environments. We computed *success rates* as the proportions of sample runs where the algorithm successfully moved the

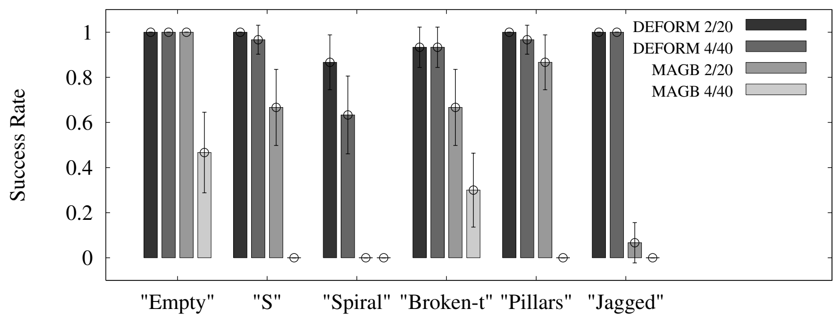


Figure 5: Success rates on all environments using Deform and Magb. “2/20” means 2 shepherds and a flock of size 20. “4/40” means 4 shepherds and a flock of size 40. Error bars indicate 95% confidence over 30 runs.

flock to the goal. Table 1 shows example results for the “Broken-t” environment. For small flock sizes, Deform and Magb perform similarly. However, as the size of the flock increases, Magb shows significant decay in performance - it is clear that more shepherds are needed to control larger flocks using the Magb behavior. On the other hand, Deform manages to achieve excellent results throughout (above 90% success rate) with no negative trend in performance up to 25 flock members.

We also ran experiments to test larger shepherd and flock sizes on the “S” environment. The results, shown in Fig. 6, show that Deform once again outperforms Magb across the board, but especially with larger flock sizes.

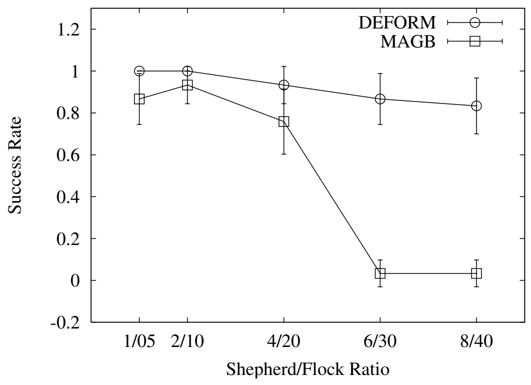


Figure 6: Success rate on “S” Environment as flock size increases. Ratio is fixed at 1 shepherd per 5 sheep. Error bars are shown at 95% confidence over 30 runs.

5.3 Experimental Results: Robustness

We define *robustness* as the ability for the shepherds to control the flock in spite of unpredictable flock behavior. We modeled the unpredictable behavior by linearly mixing each flock agent’s behavior with a random vector. We control this randomness by increasing and decreasing the magnitude of the random vector. More specifically, the randomness r is defined as:

$$r = \frac{|F_{rand}|}{|F_{rand}| + |F_{flock}|} \ , \tag{1}$$

where F_{rand} and F_{flock} are the random force and the flocking force, respectively. Fig. 7 illustrates the results (with 95% confidence bars) of success rate versus increasing randomness. In these figures, randomness is shown as a ratio from 0.0 to 0.7 where 0.0 represents fully deterministic behavior and 0.7 represents the maximum randomness attempted. In general, Deform performs significantly better than Magb with increasing flock randomness. An interesting effect is shown in Fig. 7 (top), where there is a bump in performance with increasing flock random-

Table 1: Success Rates for Deform and Magb with Varying Shepherd and Flock Sizes on "Broken-t"

		Deform				Magb			
		# Shepherds				# Shepherds			
		1	2	3	4	1	2	3	4
Flock Size	5	0.93	1.00	0.90	0.83	0.90	1.00	0.93	0.97
	10	1.00	0.93	0.93	0.93	0.56	0.97	1.00	0.97
	15	1.00	1.00	1.00	0.93	0.47	0.80	0.77	0.83
	20	1.00	0.97	1.00	0.90	0.07	0.67	0.43	0.83
	25	1.00	1.00	0.97	0.97	0.00	0.40	0.33	0.43

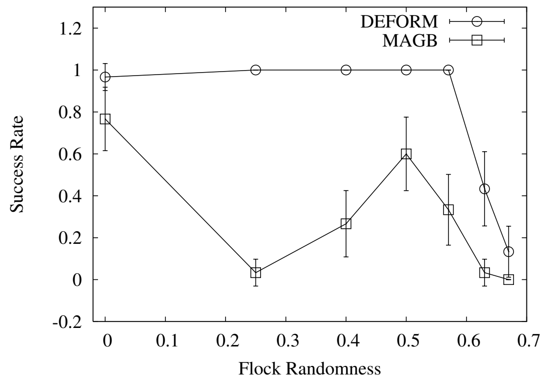
ness for the Magb behavior. We believe this occurs because the random perturbations help the flock pass through a narrow corridor like salt from a shaker.

6 Conclusion and Future Work

This paper introduces a new and powerful representation to the shepherding problem. Using a discretized, deformable blob to abstract the position of the flock members, our new Deform method performs shepherding more effectively than our previous best method [9], especially in terms of handling larger flock sizes and unpredictable flock behavior. Whereas our previous method relied on computing the medial axis of the workspace, the new method requires only an occupancy grid. This makes our new method more applicable to robotics scenarios where a medial axis is not available (i.e. when the workspace geometry is not known *a priori*).

There are some areas which deserve more attention. In the future, we'd like to try growing target blobs that favor metrics besides just "thickest". For example, it may be desired for target blobs to grow away from obstacles, towards planned waypoints, or along existing roadmap information. We are also further exploring the interaction between high level planning and these behaviors. For example, we would also like to apply the new deformable blob abstraction to the high level planning methods for shepherding that we have presented in [9], which combined shepherding behaviors with probabilistic motion planning algorithms such as Est [20], Rrt [19], and our own, Meta-Graph.

As we mentioned in Section 3, Deform could be



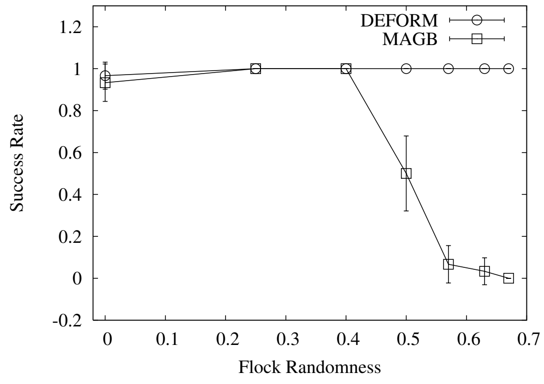


Figure 7: (**top**) Success rate on "Broken-t" as flock motion becomes more random. (**bottom**) Success rate on "S". The numbers on the x-axis indicate the ratio of randomness to deterministic flocking motion. Error bars are shown at 95% confidence over 30 runs. In both experiments, two shepherds are used to herd 20 flock members.

implemented to represent flock blobs using other geometric abstractions such as polygons, contours, or α-*shapes* [22, 23]. α-*shapes* are particularly interesting in this regard since the value of α could be based on the flock's sensing range to prevent the shepherds from disturbing the flock unnecessarily. The Iterative Closest Point (ICP) algorithm [24] could then be used to compute successive transformations of the flock. However, to be efficient, a new variation would be needed that exploits the temporal and spatial coherence of the flock's motion.

References

[1] H. Yeh, S. Curtis, S. Patil, J. van den Berg, D. Manocha, and M. Lin, "Composite agents," in *Proceedings of Eurographics / ACM SIGGRAPH Symposium on Computer Animation*, M. Gross and D. James, Eds., 2008.

[2] Z. Butler, P. Corke, R. Peterson, and D. Rus, "Virtual fences for controlling cows," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 5, April-1 May 2004, pp. 4429–4436 Vol.5.

[3] D. M. Anderson, "Virtual fencing–past, present and future," *The Rangeland Journal*, vol. 29, pp. 65–78, 2007.

[4] 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.

[5] J. Schubert and R. Suzic, "Decision support for crowd control: Using genetic algorithms with simulation to learn control strategies," in *Military Communications Conference, 2007. MILCOM 2007. IEEE*, Oct. 2007, pp. 1–7.

[6] N. Ogawa, H. Oku, K. Hashimoto, and M. Ishikawa, "Microrobotic visual control of motile cells using high-speed tracking system," *Robotics, IEEE Transactions on*, vol. 21, no. 4, pp. 704–712, Aug. 2005.

[7] 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.

[8] 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.

[9] C. Vo, J. F. Harrison, and J.-M. Lien, "Behavior-based motion planning for group control," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), St. Louis Missouri. To Appear.*, 2009.

[10] 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.

[11] 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.

[12] J. Halloy, G. Sempo, G. Caprari, C. Rivault, M. Asadpour, F. Tache, I. Said, V. Durier, S. Canonge, J. M. Ame, C. Detrain, N. Correll, A. Martinoli, F. Mondada, R. Siegwart, and J. L. Deneubourg, "Social Integration of Robots into Groups of Cockroaches to Control Self-Organized Choices," *Science*, vol. 318, no. 5853, pp. 1155–1158, 2007.

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

[14] 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.

[15] M. P. for Coherent Groups of Entities, "Motion planning for coherent groups of entities," in *IEEE International Conference on Robotics and Automation*, 2004, pp. 3815 – 3822.

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

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

[18] 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.

[19] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An Efficient Approach to Single-Query Path Planning," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 995–1001.

[20] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," *Int. J. Comput. Geom. & Appl.*, pp. 2719–2726, 1997.

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

[22] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 551–559, 1983.

[23] H. Edelsbrunner and E. P. Mücke, "Three-dimensional alpha shapes," *ACM Trans. Graph.*, vol. 13, no. 1, pp. 43–72, Jan. 1994.

[24] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.