

Technical Report TR03-009
PARASOL LAB
Department of Computer Science
Texas A&M University

Better Shepherding Behaviors Using Improved Shepherd Locomotion ¹

Ross T. Sowell O. Burchan Bayazit Jyh-Ming Lien Nancy M. Amato
Author Post-Doctorate Mentor Graduate Student Mentor Faculty Mentor

Parasol Lab., Department of Computer Science
Texas A&M University
{sowelrt0,burchanb,neilien,amato}@cs.tamu.edu

108 North Bigby Drive
Columbia, TN 38401

¹This research supported in part by NSF Grants ACI-9872126, EIA-9975018, EIA-0103742, EIA-9805823, ACR-0081510, ACR-0113971, CCR-0113974, EIA-9810937, EIA-0079874. Sowell supported in part by NSF grant IIS-0118097.

Abstract

Flocking behavior is very common in nature, and there have been ongoing research efforts to simulate such behavior in computer animations and robotics applications. Shepherding is one type of flocking behavior in which an outside agent guides or shepherds the members of the flock. Simulating this shepherding behavior is interesting because of its commonality in nature. A sheep dog guiding a flock of sheep, cowboys driving a herd of cattle, a predator hunting a group of its prey, and a flock of geese following a mother goose are all examples of shepherding behaviors found in nature. Currently, little work has been done on simulating these types of behaviors.

In this work, we investigate ways to improve existent methods of simulating single-agent shepherding behaviors. We define the locomotion of a shepherd, and we explore methods of increasing the efficiency of a shepherd in herding a flock from a start to a goal position.

1 Introduction

Simulating the coordinated behavior of a flock has attracted the attention of many researchers because it is commonly observed in nature and has numerous applications in the fields of robotics and computer animation. Shepherding behaviors, specifically, are one class of flocking behaviors in which one or more external agents attempt to control the motion of another group of agents. One example of a shepherding behavior found in agriculture is a sheep dog guiding the movement of a flock of sheep.

Shepherding behaviors have many applications in several fields of research. For example, in robotics, since training a sheep dog may take years and replacing one may be difficult, a robot may be considered as a cheaper alternative. The motion of a group as they are pursued by a hunter is of interest in computer games and computer-animated films. Similarly, a biologist might use a simulation of shepherding behaviors to study a natural behavior such as a group of animals fleeing from their predator or a flock of geese following their mother goose.

Currently, some work has been done to simulate a group whose movement can be controlled by an outside agent, but little work has been done toward simulating more interesting shepherding behaviors. Work has been done for simulating and constructing robots that shepherd a flock of geese in a simple (circular) environment [6]. Since the flock is afraid of the robot, the robot can “push” the flock into a desired destination. In addition, Tu and Terzopoulos have simulated an interesting shepherding behavior in which a T-Rex chases raptors out of its territory [5].

Bayazit et al. [2] use a roadmap to simulate shepherding in more complicated environments, i.e. environments with obstacles. In addition to finding proper positioning for the robot to push the flock in the desired direction, they also consider the re-grouping of flock members since the flock may split into groups due to environmental obstacles and forces exerted from the robot. By using the roadmap, they show that the robot can steer the flock to a destination more easily. Although their method re-groups flocks, they do not try to avoid separating the flock. In particular, when

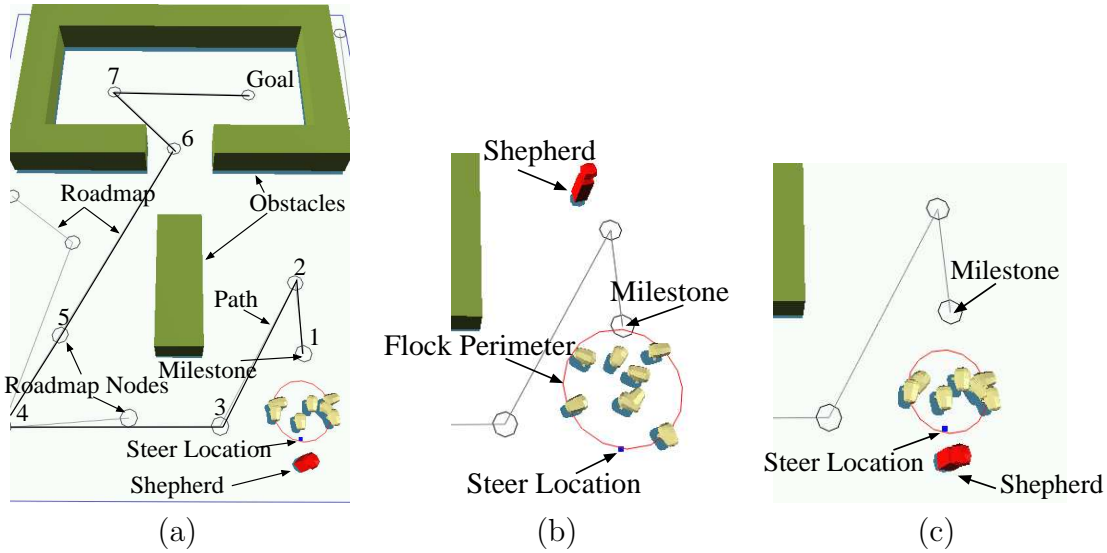


Figure 1: (a) Key elements of the environment. (b) The Approach Problem: How should the shepherd move from his initial position to the steer location? (c) The Steering Problem: Having approached the flock, how should the shepherd move in order to push the flock to the milestone?

the robot is trying to reach a designated position to push the flock, the robot does not consider how its movements to reach the pushing position will effect the flock.

In this paper, we study approaches to produce better locomotion of the shepherd by avoiding flock separation. By doing so, we expect to make the shepherd more efficient at herding a flock from a start to a goal position, since he will be spending less time reuniting separated flock members.

Shepherd’s Locomotion. We define a shepherd’s locomotion as the manner in which the shepherd will move in order to control the movement of a flock. Clearly, if the flock members are afraid of the shepherd, the shepherd can push the flock toward a given goal position by positioning himself on the “side” of the flock furthest from that goal position. However, the manner in which the shepherd should move from his given initial position to that side of the flock and any movements he should make once he is on that side of the flock are not well-defined. The shepherd’s locomotion defines how the shepherd will deal with these issues.

We will discuss possible solutions to the following locomotion sub-problems:

1. Given an initial position away from the flock and a steer location (defined in Section 2.2) near the flock, the shepherd should find a path to the steer location such that as he approaches the flock, the contour of the flock will not change too much, while also trying to keep the trajectory short. This problem is illustrated in Figure 1(b).
2. Given a milestone (defined in Section 2.2), an initial position of the shepherd that is located near the flock and on the “side” of the flock that is furthest from the milestone, and a predefined path to the milestone, the shepherd needs to

push (steer) the flock so that the flock will roughly follow the path from start to goal, without breaking the flock into groups, if possible (see Figure 1(c)).

Once the shepherd is able to control the flock’s movement, while avoiding separating the flock by performing these local motions, we will be able to simulate more interesting behaviors in the given environment.

2 Preliminary

Before continuing, we define some preliminary terms and concepts. The key terms defined are: flocking, flock, flock members, basic flocking system, rules, view-range, time-step, force, total force, shepherding behaviors, shepherd, milestone, roadmap, roadmap node, roadmap edge, path, and adaptive roadmap.

2.1 Basic Flocking Behavior

Flocking is quite simply the coordinated movement of a group of agents. In different applications, this group of agents may represent a flock of sheep, a flock of geese, a school of fish, or a herd of cattle. We will refer to this group of agents collectively as the *flock* and to the individual agents that make up the flock as *flock members*.

For our simulations, we use a *basic flocking system* based largely on Reynolds’s “Boid” dynamics [4] to model the flock. In this model, flock members are assigned a set of simple *rules* that govern their behavior. These rules are as follows:

1. Separation: Avoid colliding with nearby flock members.
2. Cohesion: Stay close to nearby flock members.
3. Alignment: Attempt to match velocity with that of nearby flock members.

Using only these rules, a basic flock which demonstrates simple coordinated movement can be modeled.

Reynolds suggests that more rules may be added to to this list to simulate more interesting behaviors. For example, Bayazit et al. [2] add both a rule of obstacle-repulsion to avoid running into nearby obstacles, and, for shepherding simulations, a rule of shepherd-fear to run away from a nearby shepherd. We also adopt both of these rules for our model.

In addition to a set of rules, each flock member is also given a *view-range* which determines which other flock members, obstacles, and shepherds are “nearby”. This view-range is defined as a sector of a circle specified by a distance (radius) and an angle (central angle). A flock member only “sees”, and therefore reacts to those other flock members, obstacles, and shepherds that lie inside the area of its view-range.

At each *time-step* of our simulation, the set of rules are applied to each member of the flock. The application of each rule causes a corresponding *force*, a vector quantity, to be applied to the flock member. The sum of all these forces yields a *total force* that will determine the movement of the flock member.

Since each flock member’s movement is calculated individually and determined only by those other members of the flock, obstacles, and shepherds that lie in its own view-range, the flock may become separated due to forces caused by obstacles or shepherds. However, the flock generally tends to stay together due to the forces of cohesion and alignment.

2.2 Shepherding Behaviors

Shepherding behaviors are a special type of flocking behavior in which an external agent (agent that is not a flock member) or agents attempt to control the movement of a flock. A *shepherd* is an external agent that influences the movement of the flock. We define a *milestone* as any position toward which the shepherd attempts to steer the flock, and we define a *steer location* to be any position toward which the shepherd moves himself in order to influence the movement of the flock (see Figure 1(c)).

It is important to note that our definition of shepherding behaviors does not limit *shepherding* to describe those behaviors in which an external agent (shepherd) attempts to guide a flock from a start to a goal position. Instead, we define shepherding as a broad term to describe any flocking behavior in which an outside agent influences the movement of a flock. For example, in patrolling, a behavior with application for dogs chasing away birds from dangerous zones in airports, one or more shepherds attempt to prevent flock members from entering a designated area. This is considered a shepherding behavior.

2.3 Roadmaps

A *roadmap* is a graph that represents the connectivity of a given environment for a given robot (see Figure 1(c)). Each *roadmap node* represents a collision-free configuration of the robot, and each *roadmap edge* represents a collision-free path between two roadmap nodes. Many methods have been proposed to automatically generate such roadmaps [3, 1, 7]. Once a roadmap has been generated, a collision-free *path*, a list of connected roadmap nodes (see Figure 1(c)), from a given start configuration to a given goal configuration may be found by simply connecting the start and goal configurations to the roadmap.

A roadmap can be used by a shepherd as a map to steer a flock from a start position to a goal position in a given environment [2]. In this method, a path is first found from the start to the goal position. Then, the shepherd treats each consecutive roadmap node in the path as a milestone, defined in Section 2.2. He steers the flock toward the first milestone (the first roadmap node in the path), and once the flock has reached that milestone, he updates the milestone to be the next roadmap node in the path. Thus, the shepherd herds the flock from a given start position to a given goal position in the environment.

3 Shepherd Locomotion

In this section, we explore ways to have the robotic shepherd more intelligently position himself so that his flock will move from a start position to a goal position in a given environment, while causing the flock to separate as little as possible. We explore and discuss both methods of approaching the flock when the shepherd is away from the flock and methods of steering the flock once the shepherd has approached the flock.

3.1 Approaching the Flock

Given an initial position of the shepherd, an initial position of the flock, and a milestone (defined in Section 2.2) to which the shepherd needs to steer the flock, the shepherd first needs to move to a position near the flock on the “side” of the flock that is furthest from the milestone (see Figure 1(a)). By doing so, he approaches the flock and positions himself in such a way that he “scares” the flock towards the milestone.

We discuss two *approaches* to this problem: a straight-line approach and a “safe-zone” approach.

3.1.1 Straight-line Approach

The simple solution to the problem of approaching the flock is to simply have the shepherd move in a straight line from his initial position to a steer location (defined in Section 2.2) located “behind” the flock at the position on the flock’s perimeter that is furthest from the milestone. This method was employed in [2].

The problem with this method is that the shepherd’s initial position and target goal behind the flock are often such that by travelling in a straight line from his initial position to the steer location, he intersects the flock, disturbing the natural contour of the flock, and often causing the flock to separate into two or more groups. This problem is illustrated in Figure 2.

3.1.2 Safe-zone Approach

We propose a safe-zone approach as one method of avoiding the problem encountered in the straight-line approach. In this method, the shepherd finds a “safe-zone” around the flock. This *safe-zone* is a region around the flock outside of which the shepherd can move freely without disturbing the contour of the flock. By not penetrating this safe-zone when approaching the flock, the shepherd can effectively approach the flock without causing the flock to separate (see Figure 3).

To show how this safe-zone is found, we must first define how we approximate the flock’s perimeter. We use the perimeter of the smallest circle that encloses all flock members as an approximation of the flock’s perimeter. This serves as a good approximation for our simulations. However, other methods of representing the perimeter of the flock exist, e.g. convex hull, and these methods might produce better results in certain situations.

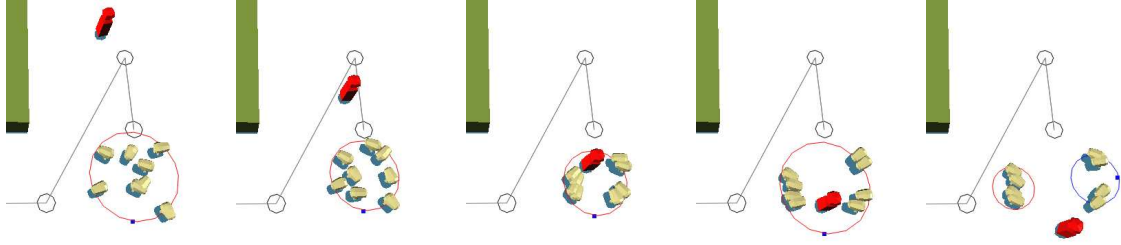


Figure 2: The shepherd intersects the flock as he moves in a straight line to his steer location (marked with a square) behind the flock, causing the flock to separate.

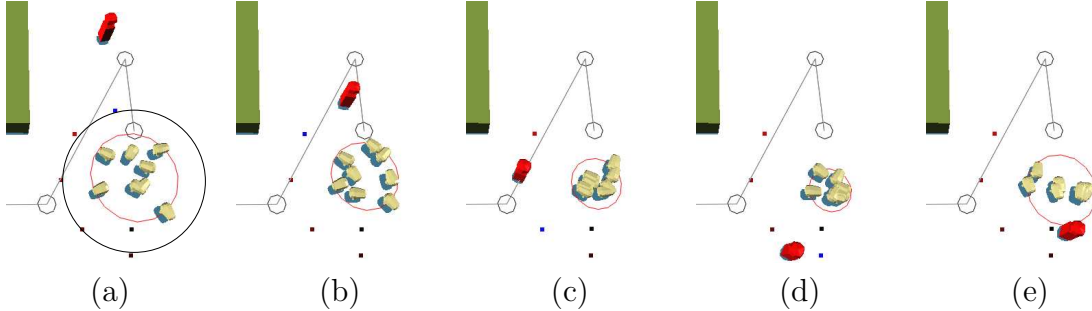


Figure 3: (a) The shepherd and flock are in their initial positions. The outside circle represents the perimeter of the safe-zone. The squares on the perimeter of the safe-zone are intermediate steer locations that the shepherd will follow as he moves around the safe-zone to the steer location behind the flock. (b) The shepherd moves straight towards the steer location behind the flock until he reaches the safe-zone. (c,d) The shepherd takes the shortest path around the perimeter of the safe-zone to the point on the safe-zone's perimeter nearest the target goal behind the flock. (e) The shepherd moves in a straight line to the steer location behind the flock.

Once the perimeter of the flock is known, forming the safe-zone is easy. Simply increase the radius of the circle that defines the flock's perimeter by a "safe" amount. A safe amount is simply an amount large enough so that when the shepherd is positioned outside this region, he does not significantly affect the movement of the flock.

After finding the safe-zone, the shepherd must now move from his initial position to the steer location behind the flock without penetrating the safe-zone. To do this, the shepherd moves in a straight line towards the steer location behind the flock until he reaches the safe-zone (Figure 3(b)). Then, he takes the shortest path around the perimeter of the safe-zone to the point on the safe-zone that is nearest the steer location (Figure 3(c),(d)). Finally, he moves from this point on the safe zone to the steer location in a straight line (Figure 3(e)).

3.2 Steering the Flock

Assuming that the shepherd has approached the flock, i.e. moved from his initial position to a steer location behind the flock, using one of the methods in Section 3.1, the shepherd must now determine how to steer the flock to the milestone (see Figure 1(b)). We discuss two strategies: move straight behind the flock and move from

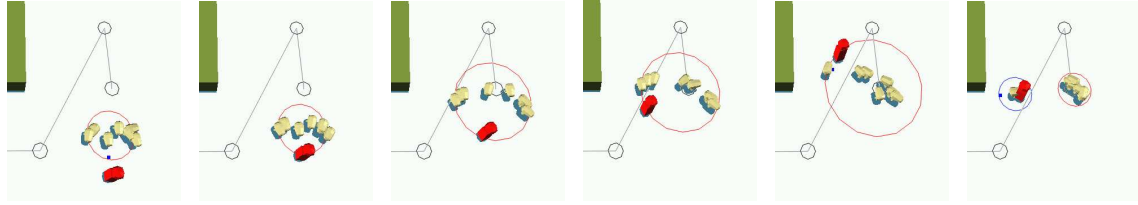


Figure 4: The shepherd guides the flock toward the milestone by always moving to (setting his steer location to be) the position on the perimeter of the flock that is furthest from the milestone. Members on the outside of the flock are pushed further outward, causing the flock to expand and eventually separate.

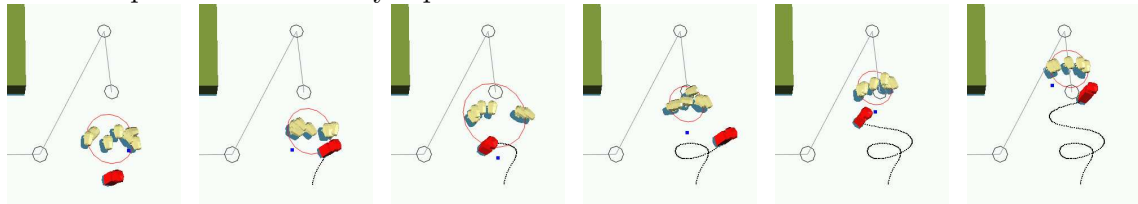


Figure 5: The shepherd guides the flock towards the milestone by moving from side-to-side behind the flock, while advancing towards the milestone. By doing so, he keeps the flock compact, avoiding separation. The shepherd's side-to-side trajectory is represented by the dotted line.

side-to-side behind the flock.

3.2.1 Move Straight Behind the Flock

Presumably, the shepherd has moved from some initial position to a steer location behind the flock, i.e. on the side of the flock furthest from the milestone. By doing so, he has scared the flock towards the milestone. Now, the simplest thing for the shepherd to do is to calculate another steer location, that is calculate the new position on the flock's perimeter that is furthest from the milestone, and move in a straight line to that steer location. Once at that position, he can repeat the process, pushing the flock closer and closer to the milestone. Then, once the flock reaches the milestone, the shepherd can calculate his next steer location based on the next milestone in his path.

This is an effective strategy, and has been used in [2]. However, since the shepherd always moves to the position on the perimeter of the flock that is furthest from the milestone, the flock members near the flock's center tend to be "pushed" very strongly toward the milestone, while the flock members furthest away from the flock's center tend to be pushed further away, thus spreading out the flock and increasing the flock's perimeter. Often, one or more of the flock's members may become separated from the flock as in Figure 4, causing the shepherd to have to regroup the flock.

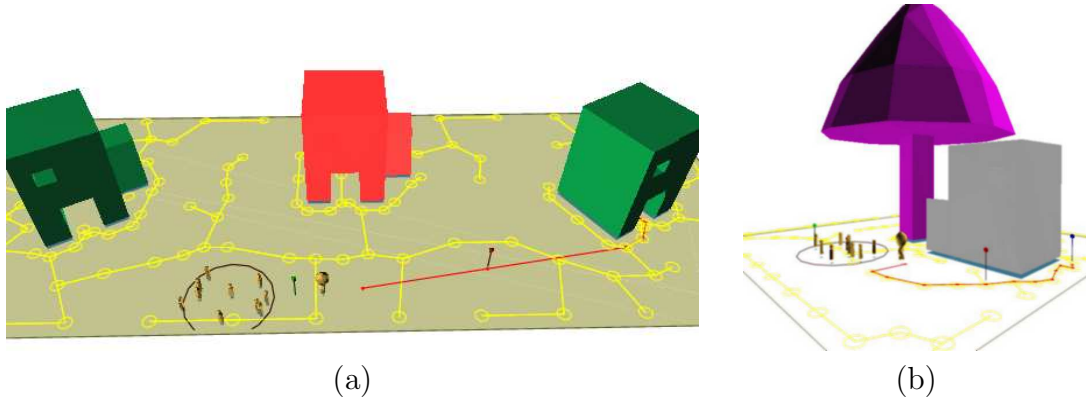


Figure 6: Above are snapshots of the two environments in which we run our experiments. We will refer to (a) as the City Environment and (b) as the Mushroom Environment.

3.2.2 Move From Side-to-Side Behind the Flock

In an effort to avoid the problem encountered by moving straight behind the flock, i.e. on the side of the flock furthest from the milestone, we propose that the shepherd move back-and-forth, from one side of the flock to the other, as he advances behind the flock. Instead of simply always moving to the point on the perimeter of the flock that is furthest from the milestone, the shepherd first moves to the right of this point, then to the left, and then back to the right. This side-to-side trajectory can be seen in Figure 5.

By following this side-to-side motion, the shepherd pushes the flock towards the milestone, while also pushing the flock members furthest away from the flock’s center back towards the center. In this way, the shepherd is able to guide the flock toward the milestone while also keeping the flock members close together, thus reducing the chance of a flock separation.

4 Experimental Results

In this section we evaluate the effectiveness of our proposed methods of shepherd locomotion, i.e., the safe-zone approach and the side-to-side steering, with the old, simple locomotion, i.e., the straight line approach and straight behind steering. We will refer to a shepherd using the straight line approach and straight behind steering as using *simple* locomotion, and we will refer to a shepherd using our safe-zone approach and side-to-side steering as using *better* locomotion.

We create two environments (see Figure 6) and run three trials in each environment. For each trial, the shepherd is given his initial position, an initial position of the flock, and a goal location to which he needs to herd the flock. Each trial is run twice, once with the shepherd using simple locomotion, and once with the shepherd employing the better locomotion.

For each trial, we calculate the number of flock members that actually reached the goal before the end of the simulation (the simulation is given a maximum time limit),

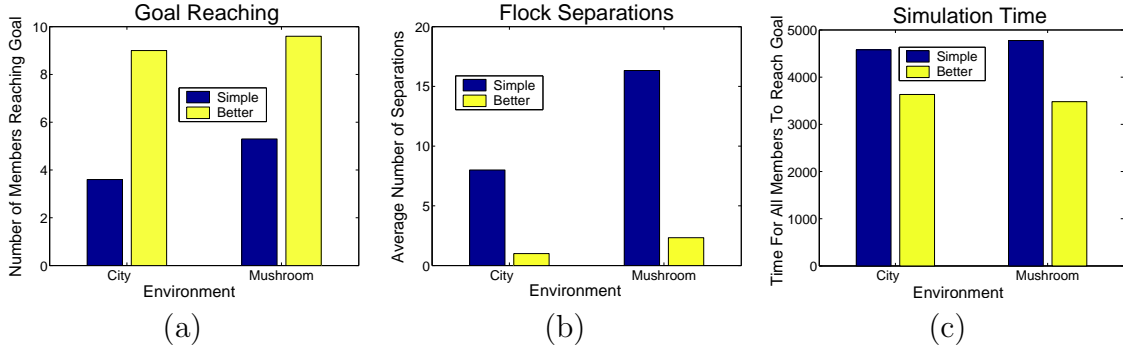


Figure 7: Our results: (a) The average number of flock members (out of a possible ten) that reached the goal before the end of the simulation (see (c)). (b) The average number of times the flock separated during the simulation. (c) The number of time-steps (each time-step is 0.5 seconds) it took for all members to reach the goal, or the maximum time of 5000 time-steps.

the number of times the flock became separated during the simulation, and the time it took for all the members to reach the goal (or the maximum time if the maximum time was reached).

Our results are shown in Figure 7. All experiments were run on a Windows system with a 1 GHz CPU and 512 MB of memory.

As seen in Figure 7(a), the number of members to reach the goal was much higher when the shepherd was employing our better locomotion than when he was using the simple locomotion. This can be attributed to the fact that we were successfully able to reduce the number of flock separations by using the better locomotion instead of the simple (see Figure 7(b)). Since the shepherd is able to keep the flock together more consistently, it is easier for him to move all of the flock members to the goal. Also, because the shepherd, when using the better locomotion, did not have to spend extra time reuniting separated members, the simulaton time for better locomotion was also less (see Figure 7(c)).

As seen from these results, the shepherd is able to move the flock from a given start to a given goal position more efficiently by using the better locomotion techniques of the safe-zone approach and side-to-side steering.

5 Discussion and Conclusion

In this paper, we have shown that a shepherd can herd a flock more efficiently from a given start to a given goal position in a given environment by using more intelligent locomotion techniques, such as the safe-zone approach and side-to-side steering, that discourage flock separations. These are only two examples of better locomotion techniques, and in future work, we plan to explore other possibilities. For example, when the shepherd is guiding the flock along a path that contains a sharp turn, he currently does not anticipate the turn, and the flock often separates. We will explore possible methods for anticipating this turn. In addition, as mentioned in Section 3.1.2, rather than representing the perimeter of the flock by the smallest enclosing circle,

other representations of the flock's perimeter might be considered. We also intend to utilize our shepherd locomotion techniques to simulate more interesting shepherding behaviors, such as patrolling, described briefly in Section 2.2. Finally, we plan to explore how multiple shepherds might work together to better control the movement of a flock.

References

- [1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Robotics: The Algorithmic Perspective*, pages 155–168, Natick, MA, 1998. A.K. Peters. Proceedings of the Third Workshop on the Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.
- [2] O. Burchan Bayazit, Jyh-Ming Lien, and N.M. Amato. Better flocking behaviors in complex environments using global roadmaps. In *Proceedings of the Workshop on Algorithmic Foundations of Robotics (WAFR'02)*, 2002.
- [3] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.
- [4] C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. In *Computer Graphics*, pages 25–34, 1987.
- [5] X. Tu and D. Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. In *Computer Graphics*, pages 24–29, 1994.
- [6] R. T. Vaughan, N. Sumpter, J. Henderson, A. Frost, and S. Cameron. Experiments in automatic flock control. *J. Robot. and Autonom. Sys.*, 31:109–117, 2000.
- [7] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1024–1031, 1999.

Acknowledgements

This research supported in part by NSF Grants ACI-9872126, EIA-9975018, EIA-0103742, EIA-9805823, ACR-0081510, ACR-0113971, CCR-0113974, EIA-9810937, EIA-0079874, IIS-0118097, by the DOE ASCI ASAP program, and by the Texas Higher Education Coordinating Board grant ATP-000512-0261-2001.

Vita

Ross Taylor Sowell was born on October 18, 1982 in Columbia, Tennessee. He graduated from Columbia Central High School in the spring of 2001. Currently, he is a junior Computer Science major at the University of the South in Sewanee, Tennessee. Upon graduation, he plans to pursue a Ph.D. degree in Computer Science.