Roadmap-Based Flocking for Complex Environments[*]

O. Burchan Bayazit
Texas A&M University
burchanb@cs.tamu.edu

Jyh-Ming Lien
Texas A&M University
neilien@cs.tamu.edu

Nancy M. Amato
Texas A&M University
amato@cs.tamu.edu

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. Generally, such work considers behaviors that can be determined independently by each flock member solely by observing its local environment, e.g., the speed and direction of its neighboring flock members. Since flock members are not assumed to have global information about the environment, only very simple navigation and planning techniques have been considered for such flocks.

In this work, we investigate how the addition of global information in the form of a roadmap of the environment enables more sophisticated flocking behaviors. In particular, we study and propose new techniques for three distinct group behaviors: homing, exploring and shepherding. These behaviors exploit global knowledge of the environment and utilize knowledge gathered by all flock members. This knowledge is communicated by allowing individual flock members to dynamically update the shared roadmap to reflect (un)desirable routes or regions. We present experimental results showing how the judicious use of simple roadmaps of the environment enables more complex behaviors to be obtained at minimal cost. Animations of these behaviors can be viewed at `http://parasol.tamu.edu`.

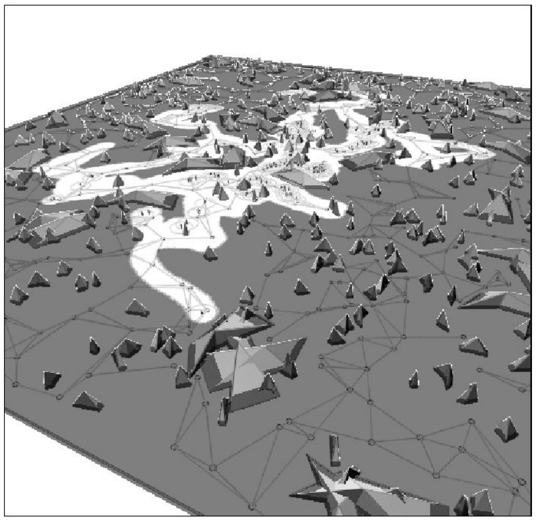


Figure 1. Global navigation information can assist coordinated group behaviors, such as flocking or mine sweeping (shown here), in complex environments.

1 Introduction

Coordinated group movement can be observed in many species. For example, birds fly in flocks, fish swim in schools, and sheep move as a herd. Simulating such behaviors requires techniques for generating the motion of the individual entities within the flock and techniques for directing the global movement of the flock. There exist good techniques for modeling individual behavior within a flock, such as Reynolds' *boids* [17]. These techniques have been coupled with simple methods for guiding global flock movement, such as attractive potential fields centered at a goal location, to achieve realistic group movement in simple environments with few external obstacles, such as birds in the air or fish in the sea. However, existing flocking methods do not perform well if complex navigation is required, such as in cities, through crowded rooms, or over rough terrain.

Path planning algorithms developed in the robotics community are capable of navigation in complex environments [11]. In particular we note the *roadmap methods* which can quickly answer many diverse path planning queries in the same environment using a map, typically constructed during preprocessing, containing a network of representative feasible paths in the environment. In essence, these maps function similarly to driving maps in that one plans a route

[*]Authors' address: PARASOL Lab, Dept. of Computer Science, Texas A&M University, College Station, TX 77843-3112. This research supported in part by NSF CAREER Award CCR-9624315, NSF Grants IIS-9619850, ACI-9872126, EIA-9975018, EIA-0103742, EIA-9805823, ACI-0113971, CCR-0113974, EIA-9810937, EIA-0079874, and by the Texas Higher Education Coordinating Board grant ARP-036327-017. Bayazit is supported in part by the Turkish Ministry of Education.

by first locating their initial and final positions and then selecting a route connecting them from the roads and highways shown on the map. While many good path planning algorithms exist, they have traditionally only been used to plan paths for a single moving object (the 'robot'). That is, they have not been customized to support coordinated group behavior.

1.1 Our contribution

In this work, we explore the benefits of integrating flocking techniques with roadmap-based path planning methods. We find that the global navigation information provided by the roadmaps can also be exploited to support more sophisticated group behaviors than possible using traditional (local) flocking methods. In particular, we consider three basic behaviors: homing, exploring, and shepherding. Our new techniques can be applied to an entire flock, to individual flock members, or to an external agent that may influence the flock (e.g., a sheep dog).

In the *homing* behavior, the goal is to move the entire flock from a starting position to a goal position. The individual flock members follow Reynolds' *'boid'* dynamics, and the flock's global motion is planned using the roadmap of the environment.

We study two types of *exploring* behavior: *covering* and *goal searching*. In the *covering* behavior, individual flock members explore the environment attempting to visit all points of the scene. This behavior is useful, e.g., for mine sweeping. In the *goal searching behavior*, individual members first explore the environment searching for a goal whose position is not known *a priori*, and then, after the goal is located, all the flock members should move to that goal. This can be used, for example, to model foraging behavior in ants. This second exploring behavior illustrates another feature of our roadmap-based methods. After the goal's location is found, it is communicated to the rest of the flock by modifying the shared roadmap, and in particular by updating the edge weights appropriately. Thus, in addition to providing a convenient data structure for storing and searching for paths, the roadmap can also be used as a communication mechanism among flock members.

The final behavior we study is *shepherding*, in which the flock is steered by an external agent (a dog). It associates a repulsive potential with the shepherd coupled with a fairly standard flocking model for the herd. This behavior has applications in robot collaboration or guarding problems.

Outline. In the next section we give a summary of related work. In Section 3, we briefly describe the *probabilistic roadmap (*PRM*)* motion planning methods used to construct the roadmaps we use in our new flocking techniques. The three different flocking behaviors we consider are discussed in Section 4. Section 5 contains experimental results and we present our conclusions and discuss future work in Section 6.

2 Related Work

Reynolds' influential flocking simulation [17] established the feasibility of modeling such a system. His work showed that flocking is a dramatic example of emergent behavior: where global behavior can arise from the interaction of simple local rules. Each individual member of the flock (boid), has a simple rule set stating that it should move with its neighbors. This concept has been used successfully by researchers both in computer graphics and robotics. Tu and Terzopoulos [23] used flocking behaviors with intention generators to simulate a school of fish in artificial life. An intention generator is similar to a finite state machine that generates emergent behaviors based on current or past states. Later, they implemented a search over possible situations expressed in formal logic [7]. They also demonstrated shepherding behavior in which a T-Rex shepherds groups of raptors out of its territory. Brogan and Hodgins [5] investigated group behavior with significant dynamics, such as human-like bicycle riders. Sun et al. [22] achieve swarm behaviors based on a biological immune system. Balch and Hybinette [2] propose a behavior-based solution to the robot formation-keeping problem. Fukuda et al. [6] describe group behavior for a Micro Autonomous Robotics System. Mataric [14] classifies a basic set of group behaviors which can be used to create more complex behaviors including flocking and herding. Saiwaki et al. [20] use a chaos model to simulate a moving crowd. An interesting approach by Vaughan et al. [24] used a robotic external agent to steer a flock of real geese.

Parker's investigation [16] of the use of global knowledge to model more complicated behavior supports our approach. She concluded that global knowledge should be used to provide general guidance for the longer-term actions of an agent, whereas local knowledge influences the more short-term, reactive actions. She also suggested that local information should be used to ground global knowledge in the current situation. This allows agents to remain focused on the overall goals of their group while reacting to the dynamics of their current situations.

Research on flocking behavior has been successfully applied to the entertainment area. For example, the behavior of individual characters in several animated films is based on flocking behavior. There is also an interest in flocking behavior in the game industry where the behavior of artificial characters is defined by flocking rules [18]. For example, real-time strategy (RTS) games usually involve moving a group of creature-like objects to accomplish tasks, like gathering resources and attacking opponents. Motion planning is required to find paths for these creatures to reach the user specified goal. The most popular motion planning method used in the game industry is A^* search [19] on a grid-based representation of the environment.

Although there is little research on path planning for flocks, much work has been done on the problem of planning for multiple robots. Motion planning methods for multiple robots fall into two categories: centralized and decoupled. Centralized methods consider all robots as one entity whose degree of freedom (DOF) is the summation of the DOF of all robots. Decoupled methods first find a path for each robot separately and then resolve conflicts between these paths. Although centralized approaches can be complete and decoupled methods usually are not, decoupled methods are usually must faster and more practical. In [12], each group of crowds is guided by a leader and the paths of the leaders are generated using a decoupled approach. Unfortunately, followers can be trapped in local minima of their leader's potential field.

Flocking behavior can also be used for optimization. The ant colony optimization (ACO) [13] meta-heuristic is a group of algorithms for discrete optimization. The method is inspired by the behavior of ant colonies. Ants find shortest paths between food sources and their hill using pheromones. Ants smell pheromone and tend to select the route with higher pheromone concentration. Because shorter paths gather pheromone faster than longer paths, the ants eventually converge on the shortest path. Dorigo et al. [13] exploit this ant-like behavior to optimize solutions for NP-Complete problems, such as the traveling salesman problem, network routing, quadratic assignment, and job scheduling. In our work, the flock's ability to explore comes from combining a roadmap of the environment with the ACO concept. Using ACO to adaptively adjust roadmap edge weights enables us to represent dynamic environmental information.

3 Roadmap-Based Path Planning with PRMs

Given a description of the environment and a movable object (the 'robot'), the motion planning problem is to find a feasible path that takes the movable object from a given start to a given goal configuration [11]. Since there is strong evidence that any complete planner (one that is guaranteed to find a solution, or determine that none exists) requires time exponential in the number of degrees of freedom (DOF) of the movable object [11], attention has focused on randomized or probabilistic methods.

As mentioned in Section 1, our approach utilizes a roadmap encoding representative feasible paths in the environment. While noting that our techniques could use any roadmap, our current implementation is based on the probabilistic roadmap (PRM) approach to motion planning [9]. Briefly, PRMs work by sampling points 'randomly' from the robot's configuration space (C-space), and retaining those that satisfy certain feasibility requirements (e.g., they must correspond to collision-free configurations of the movable object). Then, these points are connected to form a graph, or roadmap, using some simple planning method to connect 'nearby' points. During query processing, the start and goal are connected to the roadmap and a path connecting their connection points is extracted from the roadmap using standard graph search techniques (see Figure 2).

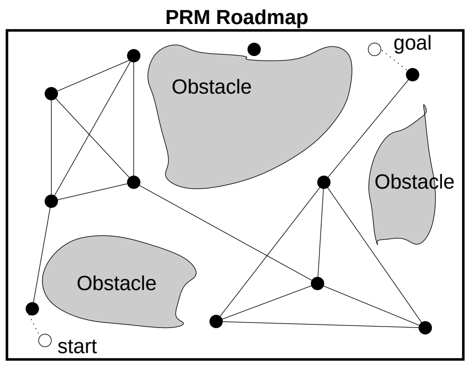


Figure 2. Querying a PRM roadmap (C-space).

We use a particular variant of the PRM called the Medial-Axis PRM, or MAPRM[1]. In MAPRM, instead of generating the nodes uniformly at random in C-space, they are generated on or near the medial-axis of C-Space. MAPRM is particularly well suited to flocking behavior since roadmap nodes tend to maximize clearance from obstacles. Note that although the initial roadmap is found for the static environment, the roadmap, or a path extracted from it, can be modified according to dynamic changes in the environment, e.g., a new roadmap could be built from scratch [8], the existing roadmap can be modified [4, 15, 21], or a path containing collisions (an approximate path) can be modified to fit the new requirements [3].

4 Group Behaviors

4.1 Homing behavior

Homing behavior consists of two sub-models, one representing the individual behavior of flock members and the other influencing the global behavior of the flock. *"Boid"* dynamics [17] sufficiently model individual behavior in most cases. In this model, individual members should: (i) avoid collision with neighboring flockmates, (ii) match velocity with them, and (iii) stay close to their neighbors. The neighborhood is defined by a distance, and an individual member of the flock is steered by angle and directional vectors satisfying the above criteria. These individual behaviors can be seen in Figure 3.

Global behavior is usually simulated using potential field methods by adding two directional vectors [10]: one toward a goal and one away from nearby obstacles. However, this method may easily be trapped in a local minimum in an environment crowded with obstacles. A method commonly used in computer games that require motion of a group of objects is a grid-based A^* search [19]. In this approach, the environment is discretized to small grid cells and the search for the flock's path is based on expanding toward the most promising neighbor of already visited positions. Although its result is optimal (shortest) and it is usually quite fast, it does have several drawbacks. The necessity of finding a completely new path for each new goal reduces the efficiency of this approach and increases the computation time for complex environments.

In contrast, roadmap-based path planning methods do not have such drawbacks. They work on the global scale and once the roadmap is generated, finding new paths is fast and efficient. In our approach we use MAPRM to find a path for the flock. One of the advantages of MAPRM is that the paths we find maximize clearances from obstacles. Once a path is found, individual flock members follow the path. The path is discretized to subgoals based on an individual flock member's sensor range. Each member keeps track of subgoals and as soon as a subgoal comes within the sensory range the next subgoal becomes the steering direction for the global goal. Note that, although all of the members follow the same path, they may be in the different parts of the path (or try to reach different subgoals) due to individual and environmental interactions. Another approach would be to have all members find their own paths. But since they are close to each other, they would find the same path. Boid rules prevent collisions between the individual members and a repulsive field from the the obstacles handles collisions with the environment.

With other interacting forces from neighboring flock members and obstacles, steering toward the subgoal has the lowest priority, so individual members still move together while moving toward goal. This results in a flocking toward the goal and avoids getting trapped in local minima.

ALGORITHM FOR HOMING

```
01.for (each individual flock member)
02.   if (goal is in its view range)
03.      stay around goal
04.   else if  (current subgoal is in its view range)
05.      set next subgoal as the target
06.   else
07.      steer toward the target
08.   endif
09.endfor
```

4.2 Exploring behavior

There are several variants of exploring. For example, flocks may have good *a priori* data about the environment and they may be looking for objects whose locations are unknown, or the environment may be unknown and the objective of the exploring behavior is to gather information about the environment. For our work, we assume the environment is known. We consider two different exploring behaviors: (i) covering the environment, and (ii) searching for a goal and moving toward the goal once it is found. To achieve these behaviors, we use a roadmap graph with adaptive edge weights. In this approach, each individual member behaves independently from its flock mates and uses the roadmap to wander around. Specifically, they follow roadmap edges and there are no predefined paths. If they reach a roadmap node with several roadmap edges, they choose a roadmap edge to follow based on the weight of the edge. The edge weights represent how relevant the edge is to the current task, either covering the environment or searching for and reaching the goal.

4.2.1 Covering the environment

In this behavior, we want some member of our flock to have covered every location in the environment. We assume we start with a roadmap covering all relevant portions of the environment. The goal is to have some flock member visit every edge and vertex of the roadmap. Initially, edges all have weight one. Flock members attempt to move along roadmap edges. As they traverse a roadmap edge they increase its weight. This is similar to ant pheromones which increase as more ants follow the same path. Since our goal is to explore the environment, the individual flock members are biased toward relatively unexplored areas of the roadmap. This is achieved by having them select roadmap edges with smaller weights with some higher probability. The algorithm is shown below.

ALGORITHM FOR COVERING THE ENVIRONMENT

```
01.for (each flock member )
02.   while (not all nodes visited)
03.      if (not in the roadmap)
03.         move to the closest roadmap node
04.      if (current node has no outgoing edge)
05.         pop stack until a new branch is found
04.      else
02.         probabilistically pick a lower-weight edge
03.         increase edge weight
09.         push this node onto the stack
10.   endwhile
11.endfor
```

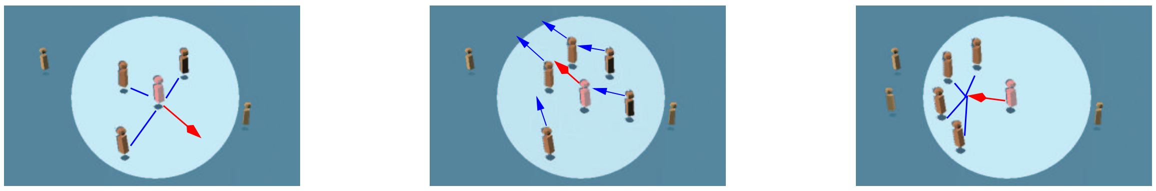


Figure 3. Individual member behavior for flocks. (a) Separation: avoid crowding neighbors. (b) Alignment: match velocity of neighbors. (c) Cohesion: stay close to neighbors. The red line represents the steering direction.

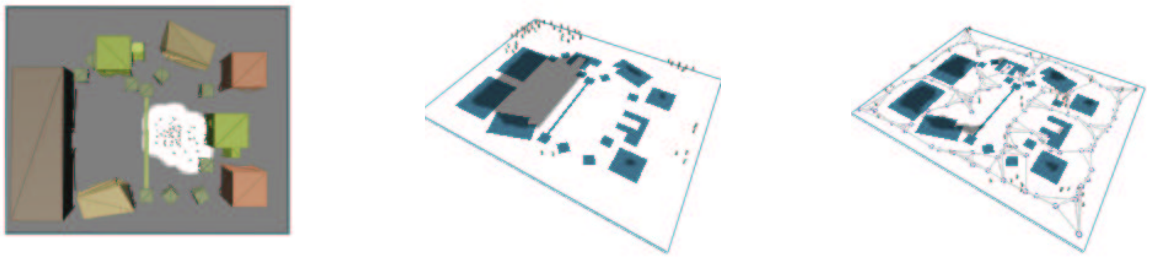


Figure 4. Covering behavior. The dark area represents unexplored regions. (a) Initial configuration of flock. (b) Normal flocking system. (c) Flocking system with roadmap-based method.

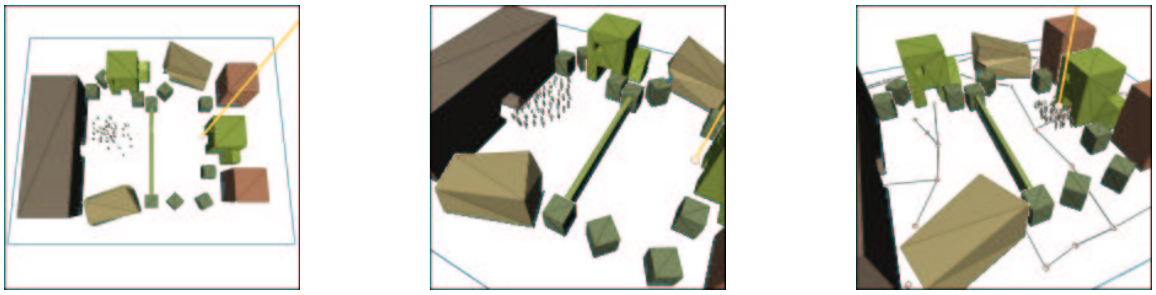


Figure 5. Searching for a goal behavior. There are 80 boids in the scene and the light tower on the right is the goal. (a) Initial configuration of flock. (b) Normal flocking system. (c) Flocking system with roadmap method.

4.2.2 Searching for an unknown goal

Our goal searching behavior is similar to *ant colony optimization (ACO)*. Although the individual flock members know the environment, they don't know the location of the goal. If an individual reaches a location where the goal is within sensor range, all other members try to reach the goal. Like the previous case, we implemented this behavior using adaptive roadmap edge weights. The weight of an edge shows how promising a path segment is. Again, the member chooses an edge to leave a roadmap node with some probability based on the edge's weight. As an individual traverses a path in the roadmap, it remembers the route it has taken. Then, when it reaches a goal, it increases the weight of the edges on the route it took. If the individual reaches a roadmap node without any outgoing connections (i.e., with only one edge) or a node already contained in the current path (i.e., a cycle), the weight of the edges it followed will be decreased.

We also investigated the case where the goal is allowed to move. Searching for moving goals is similar to searching for non-moving goals. Two stages are classified in searching for moving goals: finding the goals and staying near goals. Finding moving goals is exactly the same as searching for non-moving goals. However, since goals are moving, flock members need to keep the goal in sight and, also, update global information to tell other members which direction to go. Unlike non-moving goals, edge weights will quickly become out of date as goals move away from their

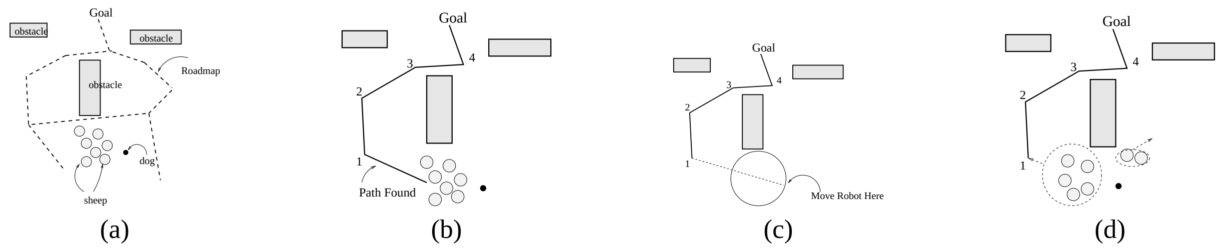


Figure 6. Shepherding: sheep are represented by large circles and the dog by a small dark circle. (a) Roadmap, (b) path selected by dog, (c) dog's steering location, (d) flock is separated.

original positions. To solve this problem, we interpret all "solutions" as suboptimal and use evaporation as in *ACO* to prevent flock members from being trapped in these "local minimum."

Algorithms for both approaches are summarized below.

ALGORITHM FOR NON-MOVING GOALS

```
01.for (each flock member)
02.   if (it finds goal)
03.      increase edge weight according to its memory
04.   else if (it finds a dead end)
05.      pop stack until a new branch is found
06.      decrease weight according to popped node
07.   else
08.      select a neighboring node of the current node
09.      push this neighbor node onto the stack
10.   endif
11.endfor
```

ALGORITHM FOR MOVING GOALS

```
01.for (each flock member)
02.   if (goal is not in view range)
03.      Find goal as in Alg. for Non-Moving Goals
04.   else
05.      find closest node to the goal
06.      update weight of incoming edges of this node
07.   endif
08.endfor
09.decrease every node weight
```

4.3 Shepherding Behavior

In the previous sections we have observed two distinct flocking behaviors. In the first case, the flock members were moving toward a goal together, i.e., as a flock. The motion was planned for the flock. In the second case, the flock members were exploring and planning their motions individually. In a sense, the flock had control of the motion in the first case and individual flock members had control in the second case. In our third scenario, neither the flock nor the individuals have control of the motion. Instead, an outside agent guides or shepherds them. In the simulation shown in Figure 6(a), the external agent is a dog whose objective is to move the flock of sheep toward the goal. The only motion control for the flock is to move away from the dog. A similar implementation has been done in [24] where a robot was programmed to move geese toward a goal position. We would like to implement a similar algorithm where a subgoal will be a roadmap node found in the path. Until the subgoal is reached, the robot will move toward that goal and then will choose the next roadmap node on the path as the next subgoal (see Figure 6(b)).

To move the flock toward the goal, the dog steers the flock from behind (Fig. 6(c)). If any subgroup separates from the flock, it is the dog's job to move the subgroup back to the flock (Fig. 6(d)).

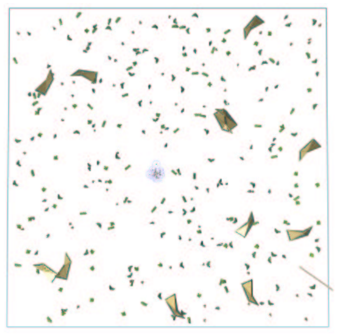
ALGORITHM FOR SHEPHERDING

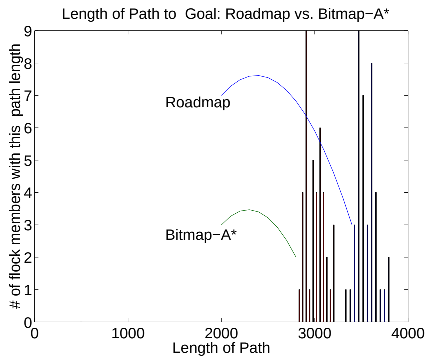
```
01.Find a path in roadmap
02.while (goal not reached )
03.   Select the next node on the path as subgoal
04.   while (subgoal not reached)
05.      Move to rear of flock on the far side of the subgoal
06.      if flock separate
07.         Move the subgroup that is farthest from subgoal
               toward other subgroups
08.      endif
09.   endwhile
10.endwhile
```

5 Experimental Results

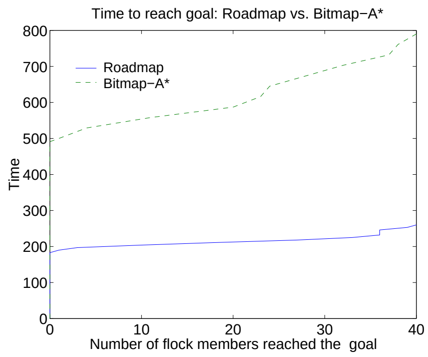
In this section we evaluate our roadmap-based techniques for the homing, exploring, and shepherding behaviors that were described in Section 4. Movies illustrating the experiments as well as the behaviors in three-dimensional space with rigid or deformable objects can be found on our webpage (http://parasol.tamu.edu).

Our experiments are designed to compare our roadmap-based techniques with more traditional approaches for simulating flocking behavior and to study the improvements


(a)



(b)



(c)

Figure 7. Homing behavior: (a) Environment for homing experiments. (b) The number of flock members reaching goals with respect to the length of the paths they took. (c) The number of flock members reaching goals over time. Although the grid-based A^* behavior finds shorter paths, the flock spends less time to reach the goals with the roadmap-based behavior.

possible by incorporating global information about the environment as encoded in a roadmap.

To study the efficiency of our exploring techniques, we also compare our roadmap-based techniques with 'ideal' variants which have complete knowledge of the environment and the current status of the search. For example, in the goal searching behavior, the location of the goal is known at all times in the ideal variant.

All of our experiments were run on a Linux system with Athlon 1.33 processor and 256MB memory.

5.1 Homing Behavior

For the homing behavior, our roadmap-based technique is compared with a basic flocking behavior using a potential field and a grid-based A^* search behavior.

The environment is a square with sides measuring 420 meters (see Figure 7(a)). It contains a total of 301 randomly placed obstacles (six types of obstacles are used). At any given time there is one goal, and when all flock members reach it, a new goal is randomly generated; this process continues until eight goals have been generated and reached. The experiment involves 40 flock members, which are initially placed according to a Gaussian distribution around the center of the square environment. The simulation is updated every 100 ms.

For the grid-based A^* behavior, a bitmap of the environment of 914×914 cells is constructed; the length of a side of each square cell is equal to the diameter of a flock member. Cells are classified as free cells and collision cells. Paths are found in this bitmap using A^* search. For the roadmap-based behavior, the roadmap is built using the MAPRM method (Section 3) to generate 400 roadmap nodes and we attempt to connect each node to its 4 nearest neighbors.

Table 1 shows that, without global information, only a few flock members reach the goal and most are trapped in local minima. On the other hand, when global navigation information is utilized, either with the grid-based A^* method or our roadmap-based method, all flock members reach the goal.

HOMING BEHAVIOR: BASIC V.S. ROADMAP

METHOD	#flockmates reaching the goal
Basic	10
grid-based A^*	40
Roadmap-based	40

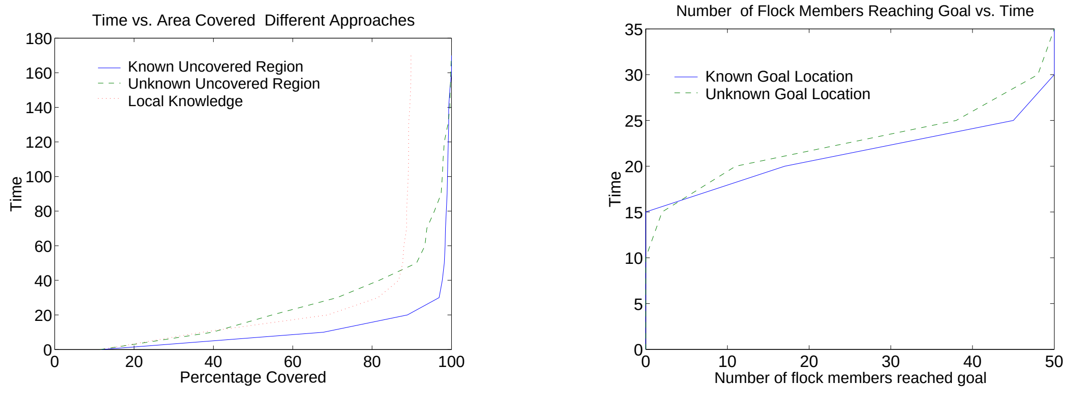
Table 1. Homing behavior. This table shows how many of the 40 flock members reach their home within 30 seconds using the basic flocking behavior, the grid-based A^* behavior, and the roadmap-based behavior.

HOMING BEHAVIOR: ROADMAP V.S. GRID-BASED A^*

BEHAVIOR METHOD	init time	find path time	local minima #	local minima escape (s)
roadmap-based	0.88	0.652	255	22.99
grid-based A^*	6.02	5.757	2005	1035.43

Table 2. Homing behavior. This table shows the time for initialization, the average time to find a path, and the total time spent by all flockmates escaping local minima.

In Table 2 we show the time spent searching for paths, the number of local minima encountered along all paths, and the total time spent escaping from local minima. This offers some insight into the methods studied, as can be seen more clearly in Figure 7. Although the flock takes a shorter path with the grid-based A^* search than with the roadmap-



(a) (b)

Figure 8. Covering behavior: (a) the percentage of the environment covered in terms of time, (b) the number of flock members reaching the goal area in terms of time.

based method (Figure 7(b)), the flock reaches the final goal faster with the roadmap-based method (Figure 7(c)). As A^* search is known to be fast and to find shortest paths, this example illustrates that our roadmap-based method indeed is a competitor for grid-based A^* methods – while the paths found are a bit longer, they are found faster.

5.2 Exploring behavior

We next consider the two types of exploring behaviors described in Section 4.2 – covering the environment and searching for a goal. For both behaviors, we compare our roadmap-based methods with very basic random walk methods and with 'ideal' variants of the roadmap-based methods which have complete knowledge of the search status at all times. For example, *a priori* knowledge of the location of the goal. The former comparison enables us to establish the value of the global information stored in the roadmap, while the latter comparison enables us to determine how far our methods are from achieving optimal performance.

5.2.1 Covering the environment

Space covering is tested on the environment shown in Figure 4, which requires flock members to pass through narrow passages to access undiscovered areas. In this experiment, we compare basic flocking behavior, roadmap-based behavior, and an ideal variant of the roadmap-based behavior that has dynamic knowledge of the undiscovered regions.

The environment (80×100) is populated with 16 obstacles (6 types of obstacles) and in total 24% of the environment is occupied by obstacles. 50 flock members are simulated and states are updated every 100ms. A bitmap is built to record discovered/undiscovered information. A bitmap cell is discovered when it is inside the sensory range of any flock member. We set the radius of the sensory circle as 5m. For the roadmap, 120 nodes are sampled and connections are attempted to each node's 4 nearest neighbors.

The roadmap-based covering behavior is described in Section 4.2.

The basic behavior uses only local information, and is essentially a random walk through the environment. It shows that the lack of global knowledge results in some areas never being discovered, especially those nearly surrounded by obstacles.

The behavior with perfect knowledge of the undiscovered locations uses the roadmap to find paths from a flockmate's current position to the closest unexplored spot. Although such knowledge would not be available in this covering application, this variant gives us an idea of how fast an environment can be covered in the best case.

As seen in Figure 8(a), the perfect behavior rapidly covered almost 91% of the environment in the first 30 seconds. The roadmap-based behavior, using indirect communication (adaptive edge weights) takes about three times as long (90 seconds), to reach a similar coverage point of 91.6%. Nevertheless, like the perfect behavior, the roadmap-based behavior found most reachable areas. In contrast, the basic flocking behavior had difficulty covering more than 80% of the environment. However, it is interesting to note that the basic flocking behavior found more undiscovered areas than the roadmap-based approach in the first 40 seconds; this is due to the basic behavior which tends to bounce around and discover 'easy' areas very quickly.

5.2.2 Searching for a goal

In this experiment, the roadmap-based behavior is compared with a simple flocking behavior that has only local information about the environment and no knowledge of the goal position, and with an ideal variant of the roadmap-based behavior that has *a priori* knowledge of the position of the goal. The environmental is the same as that used in the covering experiment (See Figure 5).

We are interested in how many flock members reach the goal and how fast they get there. As previously mentioned, the behavior with complete knowledge is used to establish a best case (lowerbound) for the simulation efficiency, and the basic behavior using only local information is used to illustrate the importance of global knowledge. The results of some experiments are shown in Figure 8. The flocks using the basic behavior do not discover any goals within 35 seconds, and in particular, none of the flock members discover the narrow passage out of the confined region in which they start. Overall, the roadmap-based behavior is competitive with the ideal roadmap-based behavior – only taking 5 seconds longer than the method in which the position of the goal is known *a priori*. In addition, it is surprising to note that two of the flock members in the roadmap-based method reach the goal earlier than any of their flockmates in the ideal roadmap-based behavior. While we expect the roadmap-based method to continue to perform well in more complex environments, we expect its efficiency relative to the ideal method to decline somewhat.

5.3 Shepherding behavior

In the shepherding experiment, the flock consists of 30 sheep and the sheep dog is an external agent. The experiment starts the flock in a random location and the objective is to move it to a randomly selected goal. When the sheep reach the goal, the experiment is repeated again by selecting a new starting position and goal position at random.

We compare our method, as described in Section 4.3, to a grid-based A^* method. The basic method using local information was not considered due to its observed inadequacies in the previous experiments. The environment is the same and the experiments are similar as for the homing experiments (Figure 7). In our grid-based A^* implementation, the search for the path to the goal and the dog's path to the steering position use A^* search on a bitmap with 914×914 cells. The roadmap method used the MAPRM method with 400 roadmap nodes, with each node connected to its 4 nearest neighbors.

Our results, shown in Table 3, include initialization times, the number of simulation steps required to reach the goal, and the number of local minima encountered. All values reported are averages over 40 experiments. We see that

SHEPHERDING BEHAVIOR: ROADMAP V.S. A^*

METHOD	Init time	#steps to goal	#local minima
Roadmap-based	0.88	2348.17	7.8
A^*-based	6.02	10612.08	32.2

Table 3. Shepherding behavior. This table shows time for initialization, the average number of simulation steps required to reach the goal, and the average number of local minima encountered.

fewer local minima are encountered here than in the previous behaviors. This is influenced by the fact that MAPRM tries to generate paths that have high clearance from obstacles and because as the dog moves to the steering position, it effects the individual members so it increases the entropy of the system, resulting in relatively more randomness. Thus, even though some members are stuck in local minima, the dog would come and retrieve them. The table shows that similar to homing behavior, our shepherding with the roadmap approach performed better than the grid-based A^* search.

6 Conclusion

In this paper, we have shown that complex group behaviors can be generated if some global information of the environment is available. The global knowledge used is a roadmap of the environment. The information it contains, such as topological information and adaptive edge weights, enables the flock to achieve behaviors that cannot be modeled with local information alone. Moreover, since in many cases global knowledge involves high communication costs between individuals, indirect communication though dynamic updates of the roadmap's edge weights provides a less expensive means of obtaining global information.

Our simulation results for the types of behaviors studied show that the performance of the roadmap-based behavior is very close to an ideal behavior that has complete knowledge. Our future work will focus on shepherding and searching for moving goals, as in pursuit/evasion games.

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 *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 155–168, 1998.

[2] T. Balch and M. Hybinette. Social potentials for scalable multirobot formations. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 73–80, 2000.

[3] O. B. Bayazit, G. Song, and N. M. Amato. Enhancing randomized motion planners: Exploring with haptic hints. *Autonomous Robots, Special Issue on Personal Robotics*, 10(2):163–174, 2001. Preliminary version appeared in ICRA 2000, pp. 529–536.

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

[5] D. C. Brogan and J. K. Hodgins. Group behaviors for systems with significant dynamics. In *Autonomous Robots*, pages 137–153, 1997.

[6] T. Fukuda, H. Mizoguchi, K. Sekiyama, and F. Arai. Group behavior control for MARS (micro autonomous robotic system). In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1550–1555, 1999.

[7] J. Funge, X. Tu, and D. Terzopoulos. Cognitive modeling: Knowledge, reasoning and planning for interlligent characters. In *Computer Graphics*, pages 29–38, 1999.

[8] D. Hsu, R. Kindel, J-C. Latombe, and S. Rock. Randomized Kinodynamic Motion Planning with Moving Obstacles. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages SA1–SA18, 2000.

[9] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[10] O. Khatib. Real–time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.*, 5(1):90–98, 1986.

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

[12] T. Y. Li, Y. J. Jeng, and S. I. Chang. Simulating virtual human crowds with a leader-follower model. In *Proceedings of 2001 Computer Animation Conference*, 2001.

[13] Dorigo M., G. Di Caro, and L. M. Gambardella. Ant algorithms for discrete optimization. In *Artificial Life*, pages 137–172, 1999.

[14] M. J. Mataric. *Interaction and Intelligent Behavior*. PhD thesis, MIT EECS, 1994.

[15] C. L. Nielsen and L. E. Kavraki. A two lovel fuzzy prm for manipulation planning. *IEEE/RSJ International Conference on Intelligent Robotics and Systems*, 2000.

[16] L. E. Parker. Designing control laws for cooperative agent teams. In *IEEE International Conference on Robotics and Automation*, pages 582–587, 1993.

[17] C. W. Reynolds. Flocks, herds, and schools: A distributed behaviroal model. In *Computer Graphics*, pages 25–34, 1987.

[18] C. W. Reynolds. Steering behaviors for autonomous characters. In *Game Developers Conference*, 1999.

[19] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1th edition, 1994.

[20] N. Saiwaki, T. Komatsu, T. Yoshida, and S. Nishida. Automatic generation of moving crowd using chaos model. In *IEEE Int. Conference on System, Man and Cybernetics*, pages 3715–3721, 1997.

[21] G. Song, S. L. Miller, and N. M. Amato. Customizing PRM roadmaps at query time. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1500–1505, 2001.

[22] S.-J. Sun and D.-W Lee K.-B. Sim. Artificial immune-based swarm behaviors of distributed autonomous robotic systems. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3993–3998, 2001.

[23] X. Tu and D. Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. In *Computer Graphics*, pages 24–29, 1994.

[24] 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.