Interactive Planning for Shepherd Motion

Jyh-Ming Lien and **Emlyn Pratt**
{jmlien, epratt2}@gmu.edu
MASC Group, Computer Science, George Mason University, Fairfax, VA, 22030, USA

Abstract

Shepherding problem asks how the movement of an agent (e.g., predator, policeman, or sheepdog) can push a group of agents (e.g., prey, crowd, or sheep) from one position to another. The shepherding problem has many important and practical applications in security, environmental protection, agriculture, education and more. While much research has focused on simulating coordinated group behaviors, the shepherding problem has not received as much attention as it should be. This work investigate the possibility of addressing the shepherding problem using motion planning strategies. From an algorithmic point of view, the shepherding problem is extremely challenging due to its extremely large state space. It is clear that the existing motion planning methods cannot provide an efficient way to solve the problem. Instead, we propose an approach that incorporates computer-human interaction techniques with algorithmic robotics: an interactive motion planning method. In our experimental results, we have shown that this combination indeed provides performance improvement over the human-only and the computer-only approach. Although the proposed work shows promising initial results from our first prototype system, there are still a long way to go before we can have a better understanding of the interactive method for motion planning and the shepherding problem.

Introduction

Shepherding problem is the problem of group motion control using agent and agent interactions, where there are typically one group (such as mob or sheep) and one or more 'controller' agents (such as police officers or shepherd dogs), whose objective is to control (e.g., navigate) the group. An example of group motion control is demonstrated in Figure 1, where a Border Collie is steering three sheep toward the camera. Shepherding problem is a challenging and fundamental problem in many important applications in security (e.g., simulation of disaster scenarios and responses (Shell and Matarić 2004)), in civil crowd control (e.g., planning exit strategies for sporting events), in environmental protection (e.g., collecting oil spills (Fingas 2001)), in agriculture (e.g., sheep herding (Schultz and Adams 1998)), in transportation safety (e.g., protecting airplanes from birds (EWT 2002)), in education and training (e.g., providing immersive museum exhibits and training systems), and in entertainment (e.g., interactive games).

Figure 1: A Border Collie (Trek) is steering three sheep toward the camera. Image obtained from (Block 2008).

Despite the importance of the shepherding problem, it is still very poorly understood how to generate strategies to obtain better control over a group. For example, a recent study (Kenny et al. 2001) highlighted the fact that incorrect strategies can lead to riots and fatal catastrophes, but strategies of controlling crowds are largely unknown and there is little work in the literature addressing the problems. A large body of work on coordinated group motion focused on only simulation with *low-level interactions*, such as collision avoidance. Some work on multiple-robot systems dealing with the shepherd problem, however, considers only small size robot teams in environments without obstacles (Schultz and Adams 1998; Vaughan et al. 2000).

Main Contribution. The shepherding problem, from the point of view of algorithmic robotics, is very challenging due to its *large state space* and *highly underactuated* nature. Not surprisingly, no existing work can directly address the shepherding problem. In our previous work (Bayazit, Lien, and Amato 2002e; Lien et al. 2004; 2005a), we have developed a *simulation* system capable of steering and navigating a group (the sheep) in an environment with obstacles using a single or multiple 'shepherds'. However, the predefined locomotions and rules of shepherds' strategy to position themselves limit us from applying our work directly to more general problems without defining a larger set and more complex rules.

In order to provide more adaptability and scalability, we propose to incorporate computer-human interaction techniques with the ideas from algorithmic robotics. More specifically, the new technique is an *interactive motion planning* method, which will provide visual hints to the users controlling the shepherds through the provided interface.

More specifically, we use laser pointers as the input devices that control the positions of the shepherds. We then use motion planning methods to provide visual hints to the users (i.e., potential shepherds positions that can lead to better control). We will briefly describe our system in the "System Overview" section. The interactions between the users and the planners will be discussed in the "Interactive Planning" section. In our experimental results, we show that this combination indeed provides performance improvement over the human-only and the computer-only approach. Although our experiments show promising results from our first prototype system, there are still a long way to go before we can have a better understanding of the shepherding problem. One of the most important ways of enhancing our system is to design planners that learn control strategies from user input. More discussion toward this goal is provided in the "Conclusion" section.

Related work

Despite extensive work on crowd simulation, *little work focused on the shepherding problem.* As we will see from the rest of this review, *although there exists some related work, no work has been proposed to study the shepherding problem systematically as a whole*.

Crowd control and shepherding simulation. Studies that attempt to address the 'crowd control' problem usually take very simple approaches by either changing or introducing new objects to the environments, e.g., road block (Kirkland and Maciejewski 2003) or barriers (Brenner et al. 2005), or by exerting influences to the crowd leader (Aubé and Shield 2004). The major drawback of these approaches is that these ad hoc techniques cannot be easily adaptable to the new scenarios and new environments.

Similar to crowd control problem, the shepherding problem considers scenarios in which one group (the shepherds) tries to control the motion of another group (the flock). In robotics, Schultz et al. (Schultz and Adams 1998) applied a genetic algorithm to learn rules for a shepherd robot to control the movement of another robot (sheep). The sheep reacts to the shepherd by moving away from it. Vaughan et al. (Vaughan et al. 2000) simulate and construct a robot that shepherds a flock of geese in a circular environment. In computer animation, Funge et al. (Funge, Tu, and Terzopoulos 1999) have simulated an interesting shepherding behavior in which a T-Rex chases raptors out of its territory. Potter et al. (Potter, Meeden, and Schultz 2001) studied a herding behavior using three shepherds and a single sheep in a simple environment. None of the above mentioned methods are able to navigate in the presence of obstacles.

We use a roadmap to integrate global navigation and shepherding in environments with obstacles (Bayazit, Lien, and Amato 2002a). However, without considering the influence of the shepherd's motion on the flock, the flock is often disturbed and separated and becomes hard to control. We further propose several strategies by which a single or multiple shepherds can position itself more intelligently so that the flock will stay together better (Lien et al. 2004; 2005a).

Shepherding as multi-robot cooperation. We can view the shepherding problem as a multiple robot system. Research in multiple robot systems considers how robots can cooperate to accomplish a task. The survey from Parker (Parker 2003) provides an overview of these systems. From the perspective of multiple robot cooperation, the task of crowd control requires inherent cooperation, in which the success of a robot in the team depends on the actions of other robots. Unlike non-inherent tasks, such as covering, inherent tasks, such as crowd control, cannot be decomposed into sub-tasks that can be solved independently and thus are generally more difficult.

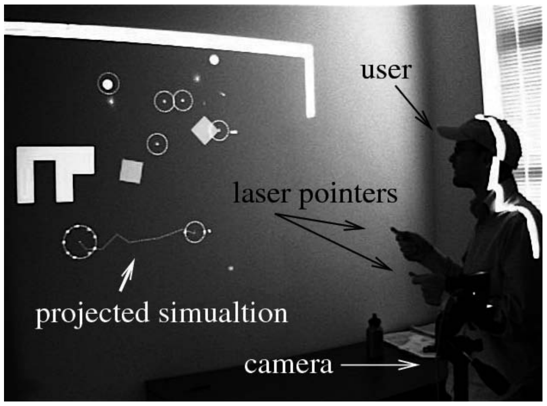
Shepherding as manipulation. The shepherding problem can also be viewed as a type of robotic manipulation task. Several researchers have attempted to use multiple robots to manipulate or move passive objects cooperatively such as pushing a box (Yamashita et al. 2003) and kicking a ball (Stone and Veloso 1998). A passive object will move only if external forces are applied to it. On the other hand, crowd control attempts to manipulate the motion of active objects, which have the ability to change their own movement even without external forces and, thus, are usually more difficult to control. To our knowledge, no methods have been proposed to manipulate multiple active objects using multiple robots. Multiple robots often form a formation, such as a line, a column, or a V shape (Balch and Arkin 1998),, to accomplish a given task. Similar observation is also found in some sociological studies of crowd control (Applegate 1969).

Shepherding as competition. The shepherding problem is also related to competitive activities of two or more groups. Examples include pursuit and evasion behaviors, or sports such as soccer. A simplified version of pursuit and evasion problem that considers only one pursuer and one evader has been studied for decades using methods such as game theory (Isaacs 1965), genetic algorithms (Reynolds 1994), and neural networks (Cliff and Miller 1996) (see (Miller and Cliff 1994) for a good survey).

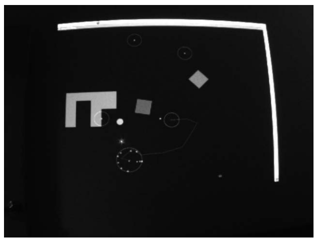
Designing the Interactive System

System Overview

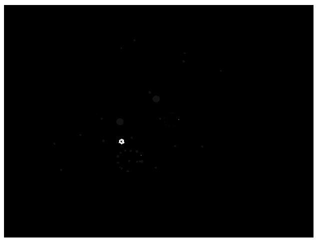
The interactive system involved (1) a projector, which projected the display of the simulation program; (2) a camera, pointed at the projected simulation; (3) a set of laser pointers, emitting at a wavelength of 650 nm (+/- 10); (4) software for tracking the laser pointers; and (5) software for running the shepherding motion planner and simulation. The projector projected a 3 channel (RGB), 1280×1024 pixel image (at 50 Hz). The camera was an IEEE-1394 Dragonfly2 (DR2-COL), produced by Point Grey Research, with a maximum 60 fps, 640×480 3 channel (RGB) image, with 8 bits per channel. Figure 2(a) shows a picture of the actual system.



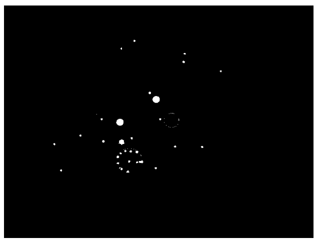
(a)



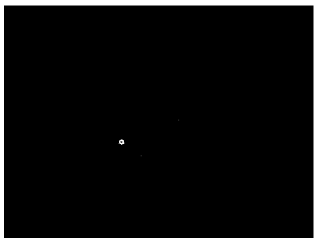
(b) raw image



(d) color filtering



(c) basic subtraction



(e) laser pointer position

Figure 2: (a) An user is using two laser pointers to control our vision-based interactive shepherding motion planning system. (b-e) Extracting the position of the laser pointer(s) from raw images captured by the camera. The (potential) laser pointer position is shown in white.

Multiple Laser Pointers Tracking

The laser pointer tracking system itself employs several stages: (1) background subtraction, (2) color filtering, (3) connected component identification, (4) association of components with laser pointers across successive frames. After the final stage, the location of each component's centroid was transmitted to the shepherding motion planner. Figures 2(b-e) illustrate the images generated after each stage.

Basic Background Subtraction The background image to be used in background subtraction was computed manually from the projected display of the paused simulation, averaged over 15 frames. For our purposes, the background subtraction routine was designed so as to yield a mask to be used on the raw frame, which will be referred to as the primary mask M. Each cell in the primary mask is either 1 or 0. A 1 appears when the Euclidean distance of the difference between RGB channels in the corresponding pixels of the background and raw frame is greater than a particular threshold θ_M (85 worked well in our setup), 0 otherwise. Let B be the background image and R be the raw frame from the camera. B_{ij} and R_{ij} refer to the pixel at the i-th row, j-th column of the background image and raw camera frame, respectively. B_{ij} and R_{ij} are each vectors in $\{0...255\}^3$.

$$M_{ij} = \begin{cases} 1 & \text{if } |B_{ij}R_{ij}| > \theta_M \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Color Filtering Once the background image was computed, the tracker ran a main loop, gathering a new raw frame from the camera at the start of each iteration. In each iteration, a primary mask M was generated, as explained above. From here, color filter calibration was performed, upon manual request by the user. Color filter calibration proceeds by taking the primary mask M and using it to collect all RGB color vectors from pixels R_{ij} in the raw frame wherever $M_{ij} = 1$. This data was clustered using the *EM k-means algorithm*, in our case using the implementation from the open source C Clustering Library (de Hoon 2008). This color filter calibration routine was performed while waving a laser pointer at the projected image of the paused shepherding simulation. Therefore, this gave us a number of clusters based on colors characteristic of the laser pointer. Following this initial calibration routine, in each iteration every pixel from R which passed through the primary mask M was subjected to a test for constructing the secondary mask N. Initially, N=M. Then, for all pixels R_{ij} from the raw frame for which $M_{ij} = 1$, the color clusters obtained earlier were used such that $N_{ij} = 1$ whenever R_{ij} is within a certain Euclidean distance threshold θ_N from the center of any color cluster. In our experience, $\theta_N = 40$ gave relatively good results:

$$N_{ij} = \begin{cases} 1 & \text{if } M_{ij} = 1 \text{ or} \\ & \text{distToNearestCluster}(R_{ij}) < \theta_N \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Connected Component Identification In each iteration of the main loop, the primary filter M (from background subtraction) and secondary filter N (from color filtering) were used along with the raw frame image R to produce a new image Q:

$$Q_{ij} = M_{ij} \times N_{ij} \times R_{ij} \quad (3)$$

Each connected component (with 8-adjacency) of image Q was then labeled according to its contours, with the help of the OpenCV computer vision library. All components containing less than 0.001% of the pixels in Q were omitted (in our case $0.001\% \times (1/100\%) \times 640 \times 480$ pixels $= 3$ pixels) for the sake of reducing noise.

Association of Components with Laser Pointers In order to consistently associate each of the components produced by a laser pointer dot with the same laser pointer across multiple frames, the centroid of each component in the current frame was compared with the centroid of components in the previous frame. For each centroid in the current frame, that centroid was assigned to the same laser pointer as the one representing the centroid from the previous frame whose Euclidean distance was closest. In the event that there were more laser pointer dots in the current frame than in the previous frame, a new laser pointer was identified. In the event that there were fewer, the oldest laser pointer was not immediately dropped. Instead, it was maintained at the same position as in the last frame it existed, up until a sufficient number of frames (we chose 10 experimentally) had passed without the reappearance of a corresponding laser pointer dot. This last scheme was chosen to more robustly handle cases when a laser pointer was accidentally turned off temporarily, when an object between the laser pointer and projected screen briefly suppressed the laser pointer dot, or in similar situations.

The flock

Our flock is modeled using Reynolds' approach (Reynolds 1987) with addition forces that push the flock away from the obstacles and the shepherds. Reynolds' influential flocking simulation established the feasibility of modeling such a system. His work showed that flocking is a dramatic example of *emergent behavior* – global behavior arising from the interaction of simple local rules. Each individual member of the flock has a simple rule set stating that it should move with its neighbors.

Interactive-based Motion Planning

Difficulty of the Shepherding Problem

We view the shepherding problem as a type of motion planning problems. Our goal in this motion planning problem is to find a path for the shepherd so that after the shepherd finishes the trajectory the flock will be in the goal position. It is obvious that the motion planning version of the shepherding problem is highly intractable due to its extremely high dimensional configuration (state) space.

The difficulty of the shepherding problem continues when we look at the problem more carefully. For example, we can further categorize the shepherding problem as a "*highly-underactuated multi-robot cooperative deformable-object manipulation planning problem*."

- Shepherding is a *highly underactuated* motion planning problem since the dimensionality of the configuration space (collectively from all flock agents and from the shepherds) is much larger than the number of control parameters (e.g., the positions of the shepherds).
- Shepherding is a type of *manipulation planning*. More precisely, group control is a problem of manipulating a set of 'active' objects, which can control their own movement without external forces and will be forced to move as the 'end-effector' (i.e., the shepherds) approaches.
- Shepherding is a type of *inherent* multi-robot cooperative task (Parker 2003).
- Shepherding is a motion planning problem dealing with *dynamic and deformable* environment (Rodriguez, Lien, and Amato 2007).

Although there is a strong relationship between the shepherding problem and motion planning, *no existing motion planning methods can fully address the shepherding problem.* For example, research on underactuated robots mostly focused on nonlinear, dynamic, or non-holonomic constraints (Shammas 2006), and most manipulation planning considers only grasping-based manipulations (Siméon et al. 2004). Pushing-based manipulation planning (van der Stappen and Overmars 2007), which is closer to the group control problem, only considers "passive" objects. Even though work exists for planning cooperative manipulating motion by multiple arms (Koga and Latombe 1994; Li and Latombe 1997) and by multiple mobile robots (Yamashita et al. 2003), most multi-robot motion planning considers only a few robots and mostly considers navigation problem without high level interactions. Moreover, planning motion for deformable object usually assumes the object is elastic (Anshelevich et al. 2000; Bayazit, Lien, and Amato 2002d) or that the object can actively deform itself (Rodriguez, Lien, and Amato 2006; Gayle et al. 2005) without the manipulator.

Definitions: Shepherd and Shepherd's Locomotion

Before going into more details about our planner, lets define some terms more carefully. A *shepherd* is an external agent that influences the movement of the flock. A *flock* is a collection of agents that tries to keep away from the shepherd. The shepherd's task is to steer the flock to desired locations. In addition to steering, the shepherd unites separate flock *groups*. In a group, each member can *see* at least one member in that group. Usually, flock separation is caused by repulsive forces exerted from obstacles or shepherds. The *flock contour* is the smallest polygon that encloses all flock members.

A *milestone* is any position toward which the shepherd attempts to *steer* the flock, and a *steering point* is any position toward which the shepherd moves himself in order to influence the movement of the flock; see Figure 3(a). As in (Bayazit, Lien, and Amato 2002e), a milestone is a node of a global *dynamic roadmap* close to the flock and a steering point is a point on the opposite side of the flock from the milestone. A roadmap is an abstract representation of the feasible space in a given environment. A dynamic roadmap is a roadmap storing information that changes dynamically during simulation.

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. The shepherds locomotion remains invariant in different shepherding behaviors and dramatically affects the quality of simulation. We divide the shepherd's locomotion into two sub-problems: *approaching* and *steering*.

Motion Planning for Shepherd

Instead of developing a complete motion planner (that always finds a solution if one exists), which has been shown

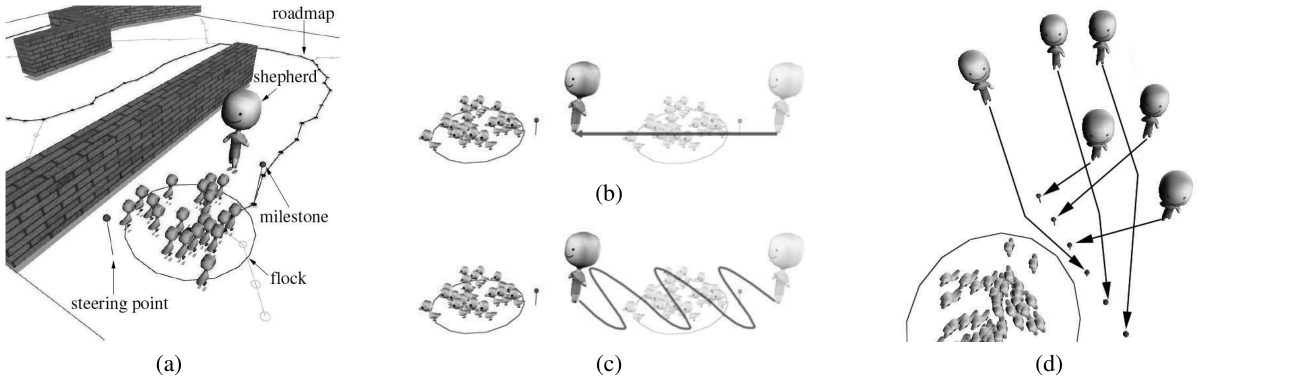


Figure 3: (a) An environment and associated terms. Steering the flock using (b) a straight line (c) a side-to-side motion. (d) Six shepherds approach the flock.

to be impossible in the earlier sections, we sacrifice the completeness to gain efficiency. Our planner will require assistance from the user to provide input (using laser pointers) that can correct the errors made by the planner. Our planner simply re-plans the shepherd's motion after the user provides any modifications. Briefly, our planner will first find a path from the *center* of the flock to the goal position using a roadmap. The shepherd then pushes the flock along the path by performing a sequence of 'local plannings' (i.e., the locomotions) based on the flock's current position. Note that this approach may not lead the flock to the final goal position because, for example, the shepherd may not have enough room to navigate the flock. Therefore, the planner will depend on the user to recognize the problem and provide input to correct this. From the interactions between the user and the planner, our new method gains both efficiency and accuracy.

More specifically, we have developed the motion planner for shepherd using adaptive roadmap-based techniques (Bayazit, Lien, and Amato 2002c; 2002b; 2005; Lien et al. 2005b). We used the global information provided by our adaptive roadmaps to improve the behavior of autonomous characters, and in particular, to enable more sophisticated group behaviors that are impossible using traditional (local) flocking methods. A roadmap of a given environment is a motion planning technique that attempts to represent the connectivity of feasible areas in an environment; see Figure 3(a). An adaptive roadmap is a roadmap whose node and edge values can be updated according to information gathered by agents in a group. We also extended ideas from cognitive modeling and embedded "behavior rules" in individual flock members and in the nodes and edges of the roadmap. Key features of our approach include:

- The roadmap provides a convenient abstraction of global information in complex environments.
- Adaptive roadmaps (e.g., modifying node and edge weights) enable communication between agents.
- Associating rules with roadmap nodes and edges enables local customization of behaviors.

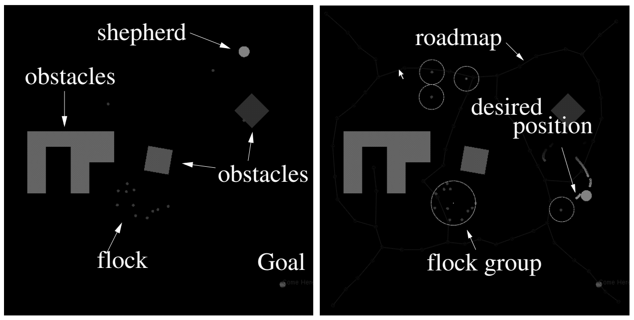
Single Shepherd. A shepherd can use roadmaps to steer the flock and to re-group separated flock members. When the shepherd re-groups separated flock members, the shepherd does not consider how its own movements will affect the flock. For example, when the shepherd approaches the flock, the shepherd does not attempt to avoid disturbing or separating the flock. We focus on improving the shepherd's movements to gain better control of the flock's motion and use this improved control to demonstrate a wider variety of shepherding behaviors. An improvement of the shepherd's locomotion can be seen in Figures 3(b) and 3(c). We have shown that a shepherd can control a flock more efficiently using more intelligent locomotion techniques. We observe that flock separation and traveling time were reduced when the shepherd uses these new locomotions in both open and cluttered environments. We have also shown that our locomotions enhance the shepherd's ability to handle larger flocks and to handle different types of flocks.

Table 1: Experiment with one laser pointer

	using laser pointer		
	without hints	with hints	simulated
time (sec)	80.6	57.1	105.0

Multiple Shepherds. We further extend this work to multiple shepherds and we study how a group of shepherds can work cooperatively **without communication** to efficiently control the flock. We explored how multiple shepherds can be used to gain better control of more difficult or larger flocks. In real working situations, it is typical to see several dogs herding a flock. In such circumstances, the shepherds need to work cooperatively to guide the flock; see Figure 3(d).

We observe that a relatively large flock (≈ 40 members) or a flock composed of members that are more difficult to control, such as cattle, which is less afraid of the shepherds thus harder to 'push', can be herded more efficiently using multiple shepherds than with a single shepherd.



(a) without visual hints (b) with visual hints

Figure 4: (a) Navigation without visual hints. (b) Our motion planner provides visual hints to the user by suggesting the desired trajectory of the shepherd.

Experimental Results

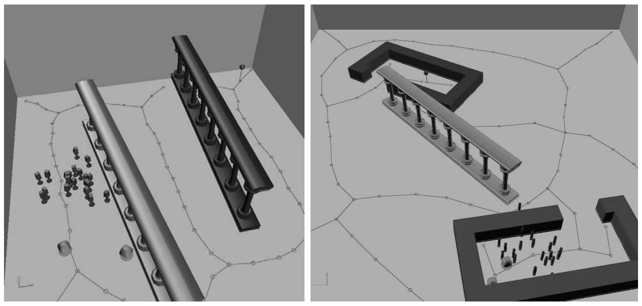
In this section, we evaluate the proposed system. All videos captured during our experiments can be downloaded from `http://cs.gmu.edu/~jmlien/shepherding/`. These videos demonstrate how a single or multiple laser pointers can be used to solve the shepherding problems.

In our first set of experiments, we compare the total navigation time for a single shepherd to push a flock with 15 members to the goal. We asked four people to guide the shepherd individually first using a laser point *without* the visual hints and then we asked the users to perform the same task again with dynamic visual hints shown on the screen. These users are allowed to have practice runs before the experiments. We also assessed autonomous navigation of the shepherds without user inputs based on the method described in the previous section. Figure 4 shows the differences between the navigations with and without visual hints. The visual hints provided by the motion planner include:

- the desired positions of the shepherds (shown in green)
- the path that connect each (sub-)group to its goal position
- groups of flock that are visible from each other (shown as yellow circles)
- global roadmap for navigating the flock groups

Table 1 shows the averaged navigation times from these three approaches from four users.

It is clear from Table 1 that the computer-only planner is the slowest among the three approaches. Without visual hints, navigation using laser pointer is about 25 seconds faster than the computer-only planner. With the help from the planner (with visual hints), users save about 23 more seconds. This shows our strategy is working. We observe that the plans and the visual hints generated by our planner is more useful for some situations (e.g., when flock forms a single group) but when there are several separated flock groups, the plan generated by the planner is difficult to follow for either the computer-only or the user-controlled shepherd. However, the users soon learn that they do not need to follow the visual hints when there are many separated groups and therefore outperforms both "without hints" and "computer-only" methods.



(a) Environment 2. (b) Environment 3

Figure 5: Two additional environments in our experiments

In the second set of experiments, we study the navigation efficiency by varying the flock size and the shepherd size. All visual hints from the motion planner are rendered in this set of experiments. Table 2 shows experimental results using the environment in Figure 5(a). We compare the navigation times using one and two laser pointers. Note that navigation tim is measured in simulation time steps and each time step is $\frac{1}{25}$ seconds. The navigation times are not significantly different when the flock size is less than 20. However, the navigation times using two laser pointers are about 400 and 6000 time steps faster than using just one laser pointers for navigating the flock with 25 and 30 members. When we compare the autonomous navigation using one or two simulated shepherds, the difference is even more pronounced. Even though the autonomous navigation outperforms the interactive navigation with small flock, it simply cannot navigate more than 20 members in this environment. In many cases (i.e., those marked with > 30000), the shepherds are not even able to push the flock forward.

Table 3 shows another set of experimental results using the environment in Figure 5(b). In this experiment, using two laser pointers also result in more efficient navigation. However, in this example, we found that the autonomous navigation with two shepherds outperforms the interactive navigation. The main reason for this is because of the limitations of our interactive interface that restricts the minimum distance between the laser pointers. In this environment, shepherds need to stay closely in order to push the flock out of the first enclosed region. Our interactive shepherds failed to accomplish this task and have to break the flock into subgroups, which make the navigation more difficult. We expect to improve the efficiency of the interactive navigation when more sophisticated tracking methods are used.

Conclusion and Discussion

In this work, we investigated the possibility of addressing the shepherding problem using motion planning strategies. From an algorithmic point of view, the shepherding problem is extremely challenging due to its large state space and spanning a broad range of research areas. However, no

Table 2: Navigation time in Environment 2 (Figure 5(a))

		size of the flock					
		5	10	15	20	25	30
shepherd control type	1 laser pointer	4400	5585	6624	8949	10395	19453
	2 laser pointers	3931	5863	6862	*8585*	*9964*	*13437*
	1 autonomous	*1309*	*3571*	*4976*	>30000	>30000	>30000
	2 autonomous	1499	4534	7253	16247	>30000	>30000

Table 3: Navigation time in Environment 3 (Figure 5(b))

		size of the flock
		20
shepherd control type	1 laser pointer	12376
	2 laser pointers	11726
	1 autonomous	>30000
	2 autonomous	8026

existing motion planning methods can provide an efficient way to solve the problem. Therefore, we propose an approach that incorporates computer-human interaction techniques with algorithmic robotics: an interactive motion planning method. In our experimental results, we have shown that this combination indeed provides performance improvement over the human-only and the computer-only approach.

Although our system shows promising results from the first prototype system, there are still a long way to go before we can have a better understanding of the shepherding problem. There are many possible ways we can take to improve our methods.

Learning from Users

One way to improve our system is to design our planner so that it learns from the user inputs. As we observed in our experimental results, the planner does not perform well when the flock separates into small groups. The planner should identify such cases easily and should start to observe the order that users (re-)group the flock. We understand that the users may also make mistakes, thus an evaluation step is necessary before learning. To do so, the planner will first observe if the laser pointer follows the provided hints or not. If so, then the user and the planner agreed on the same strategy. If not, the planner will evaluate if the user control makes the flock closer to the goal state or reduces the number of the separated groups. If the flock control is indeed improving, the planner will remember in what order the flock groups are handled.

References

Anshelevich, E.; Owens, S.; Lamiraux, F.; and Kavraki, L. 2000. Deformable volumes in path planning applications. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2290–2295.

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

Aubé, F., and Shield, R. 2004. Modeling the effect of leadership on crowd flow dynamics. In Sloot, P. M. A.; Chopard, B.; and Hoekstra, A. G., eds., *ACRI*, volume 3305 of *Lecture Notes in Computer Science*, 601–621. Springer.

Balch, T., and Arkin, R. 1998. Behavior-based formation control for multirobot teams. *IEEE Trans. Robot. Automat.* 14(6):926–939.

Bayazit, O. B.; Lien, J.-M.; and Amato, N. M. 2002a. Better flocking behaviors using rule-based roadmaps. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 95–111.

Bayazit, O. B.; Lien, J.-M.; and Amato, N. M. 2002b. Better group behaviors in complex environments using global roadmaps. In *Artif. Life*, 362–370.

Bayazit, O. B.; Lien, J.-M.; and Amato, N. M. 2002c. Roadmap-based flocking for complex environments. In *Proc. Pacific Graphics*, 104–113.

Bayazit, O. B.; Lien, J.-M.; and Amato, N. M. 2002d. Probabilistic roadmap motion planning for deformable objects. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2126–2133.

Bayazit, O. B.; Lien, J.-M.; and Amato, N. 2002e. Better group behaviors using rule-based roadmaps. In *Proceedings of the Workshop on Algorithmic Foundations of Robotics (WAFR'02)*.

Bayazit, O. B.; Lien, J.-M.; and Amato, N. M. 2005. Swarming behavior using probabilistic roadmap techniques. *Lecture Notes in Computer Science* 2005(3342):112–125.

Block, B. 2008. Vortex agility and dog training. http://www.vortexagility.com/trek.htm.

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

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

de Hoon, M. 2008. Open source clustering software: C clustering library version 1.43:. http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/software.htm.

2002. Endangered Wildlife Trust. EWT airport safety project. http://www.ewt.org.za/.

Fingas, M. F. 2001. *The basics of oil spill cleanup*. Lewis Publishers, 2nd edition.

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

Gayle, R.; Segars, P.; Lin, M. C.; and Manocha, D. 2005. Path planning for deformable robots in complex environments. In *Proceedings of Robotics: Science and Systems*.

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

Kenny, J. M.; McPhail, C.; Farrer, D. N.; Odenthal, D.; Heal, S.; Taylor, J.; Ijames, S.; and Waddington, P. 2001. Crowd behavior, crowd control, and the use of non-lethal weapons. Technical Report A274644, Penn State Applied Research Laboratory.

Kirkland, J., and Maciejewski, A. 2003. A simulation of attempts to influence crowd dynamics. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 5, 4328–4333.

Koga, Y., and Latombe, J.-C. 1994. On multi-arm manipulation planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 945 – 952.

Li, T.-Y., and Latombe, J.-C. 1997. On-line manipulation planning for two robot arms in a dynamic environment. *Int. J. Robot. Res.* 16(2):144–167.

Lien, J.-M.; Bayazit, O. B.; Sowell, R.-T.; Rodriguez, S.; and Amato, N. M. 2004. Shepherding behaviors. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 4159–4164.

Lien, J.-M.; Rodriguez, S.; Malric, J.-P.; and Amato, N. M. 2005a. Shepherding behaviors with multiple shepherds. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 3413–3418.

Lien, J.-M.; Rodriguez, S.; Tang, X.; Maffei, J.; Corlette, D.; Masciotra, A.; and Amato, N. M. 2005b. Composable group behaviors. Technical Report TR05-006, Parasol Lab, Dept. of Computer Science, Texas A&M University.

Miller, G., and Cliff, D. 1994. Co-evolution of pursuit and evasion i: Biological and game-theoretic foundations. Technical Report CSRP311, School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.

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

Potter, M. A.; Meeden, L.; and Schultz, A. C. 2001. Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *IJCAI*, 1337–1343.

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

Reynolds, C. W. 1994. Competition, coevolution and the game of tag.

Rodriguez, S.; Lien, J.-M.; and Amato, N. M. 2006. Planning motion in completely deformable environments. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2466–2471.

Rodriguez, S.; Lien, J.-M.; and Amato, N. M. 2007. A framework for planning motion in uncertain environments. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*.

Schultz, A. C., and Adams, W. 1998. Continuous localization using evidence grids. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 4, 2833–2839.

Shammas, E. A. 2006. *Generalized Motion Planning for Underactuated Mechanical Systems*. Ph.D. Dissertation, Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA.

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

Siméon, T.; Laumond, J.-P.; Cortés, J.; and Sahbani, A. 2004. Manipulation planning with probabilistic roadmaps. *Int. J. Robot. Res.* 23(7-8):729–746.

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

van der Stappen, D. N. A., and Overmars, M. 2007. Pushing a disk using compliance. *IEEE Trans. Robot. Automat.* 23(3):431–442.

Vaughan, R. T.; Sumpter, N.; Henderson, J.; Frost, A.; and Cameron, S. 2000. Experiments in automatic flock control. *J. Robot. and Autonom. Sys.* 31:109–117.

Yamashita, A.; Arai, T.; Ota, J.; and Asama, H. 2003. Motion planning of multiple mobile robots for cooperative manipulation and transportation. *IEEE Trans. Robot. Automat.* 19(2):223–237.