Simulating Crowd Motion with Shape Preference and Fuzzy Rules

Jen-Yao Chang
Computer Science Department
National Chengchi University
64, Sec. 2, Zhi-Na Rd., Taipei, Taiwan 116

Tsai-Yen Li
Computer Science Department
National Chengchi University
64, Sec. 2, Zhi-Na Rd., Taipei, Taiwan 116

Abstract

Crowd simulation is becoming indispensable in computer games and animations. Several approaches to this problem have succeeded in generating flocking behaviors with virtual forces but it remains difficult to manipulate the collective behaviors of a crowd to respect certain desired shape. In this paper, we propose a crowd simulation system that can generate a feasible path taking the crowd to the goal while maintaining a user-specified shape as much as possible. The system consists of two key components: a path planner that can search for a feasible path for a region of flexible shape and a local motion controller based on three classes of fuzzy rules for generating desired agent behaviors. We will demonstrate the implemented system with several examples to show the generated path and the corresponding crowd motions adhering to a desired shape.

Keyword: Motion planning, computer animation, multi-agent system, crowd simulation

1 Introduction

Group formation has many applications in the entertainment industry. For example, formation of a group in a battle game like the one shown Figure 1 could be a key factor in defeating an enemy. Different shapes formed by a group could provide different functions for distinct physical and visual effects. We often see animations showing a crowd of congregated animals through a shape of elephant, geometry, or a death's-head in traditional cartoons to exaggerate the visual effects and to convey special contextual meanings to the audience.

In traditional animation production, animators manually arrange the position of every animated object in each key-frame in order to show the final flocking behavior. This process is very tedious and time-consuming even with the helps of animation tools available today. In the literature of computer animation, some motion-planning techniques have been proposed to generate a path for a rectangle shape and then constrain motions of the agents in a group in this shape. However, the rigid shape greatly limits the applicability of this technique in a real animation production. In this paper, we propose a new crowd animation tool, in which a new motion planner is designed to take the requirement of flexible shape into account. Given the geometric description of environment, the planner is capable of generating a collision-free path for the deformable object while accounting for the preference of a user on the shape of the crowd. The final crowd motion is generated with the help of fuzzy rules in a agent-based simulation system to show the desired visual effect.

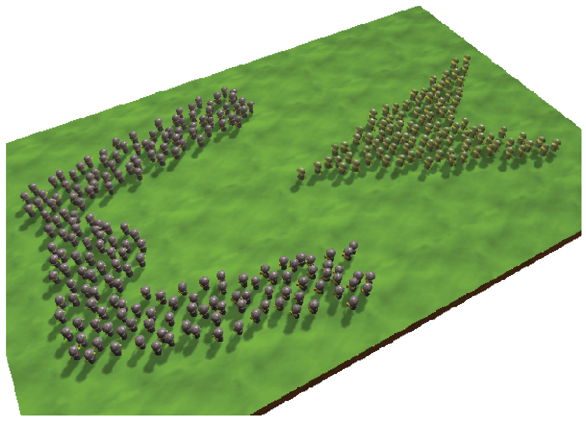
Figure 1. Snapshot of group formation in a battle game

Next, we will first describe some research pertaining to our work and then show how to formulate this new planning problem and the algorithm for generating such a path automatically. Several experimental results showing how the shape of the crowd changes during a path will also be given at the end.

2 Related Work

Simulating emergent behaviors for virtual agents is a common topic in computer animation. Reynolds proposed a virtual force model (separation, cohesion, and alignment forces) for the simulation of flocking behaviors [1][2]. The main advantage of this approach is ease of use. However, it is also difficult to tune the weights of these forces directly in order to achieve a specific effect. Anderson used a simple force model in [3] to create a flocking behavior for a crowd with an expected appearance by continuously adjusting the position and velocity of the agents according to the estimated global shape. This method can generate a crowd motion respecting the given desired appearance but the time spent in adjusting the positions and velocities of the agents is also too large for on-line applications.

In the literature of Robotics, it has been a classical problem to generate a feasible path for a robot moving from its initial configuration to a goal configuration. One can find many approaches to this problem in Latombe's book [4]. However, most the traditional motion planner

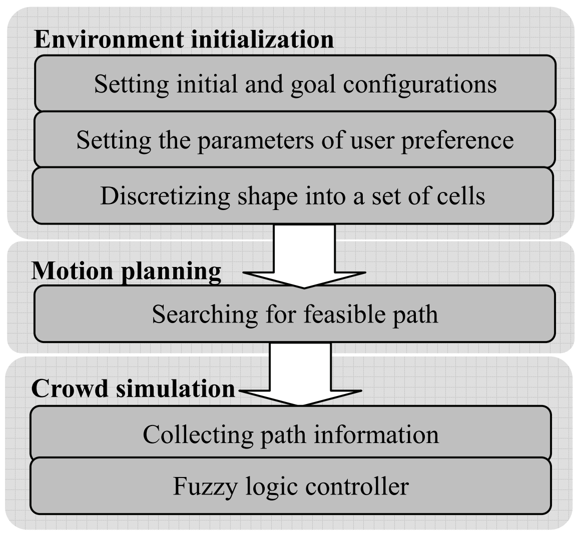
Environment initialization
Setting initial and goal configurations
Setting the parameters of user preference
Discretizing shape into a set of cells
Motion planning
Searching for feasible path
Crowd simulation
Collecting path information
Fuzzy logic controller

Figure 2. System architecture

considers the robot as a rigid body or chains of rigid bodies whose shape cannot be changed during the trajectory. In [5], Bayazit added some global information in the roadmap of the environment to facilitate more sophisticated flocking behaviors, and proposed three group behaviors exploiting global knowledge of the environment.

Kamphuis modeled a group of crowd with a deformable shape, and planned the global motion of the shape by Probabilistic Roadmap Method (PRM) [6]. He used a group potential field to control the local motion of entities in the shape. This method can solve the problem of controlling the appearance of a moving crowd, which cannot be solved by applying behavior rules only. However, the bottleneck of this method is that the crowd can only move along the path consisting of simple shapes and constrained by narrow passages.

Adaptive fuzzy logic controller (FLC) provides a good solution for autonomous robot to move in an uncertain environment. The main advantage of FLC is that it is highly adaptive and nimble in tackling the uncertainty of control system through linguistic presentation. Tunstel used the idea of FLC to design an autonomous robot having three types of basic behaviors as described in [7]: *goal-seek*, *route-follow*, and *localize*. Corresponding primitive actions were also designed for each behavior.

3 Problem Description

The architecture of the system proposed in this work, as shown in Figure 2, consists of two key components: a path planner to search for a feasible path for a flexible region and a local motion controller based on fuzzy logics. We assume that we are given a geometric description of the environment as well as the initial configuration and a goal position of the virtual crowd. We hope that that proposed motion planner can generate a feasible path for the region of flexible shape automatically. While

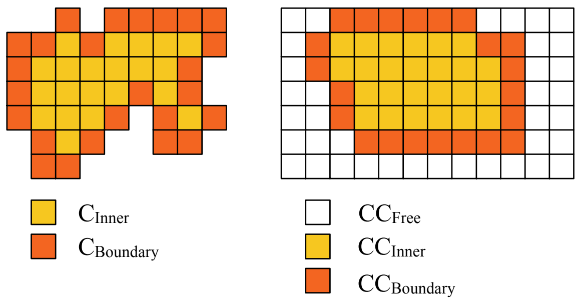

Figure 3. Example of shape configuration and configuration coverage map

accounting for the length of the path, the planner should generate a path that can allow the crowd to maintain some preferred shape as much as possible during the movement of the crowd along the path.

Our system first searches for a feasible path for the flexible region of a fixed volume. We use a set of cells called *shape configuration* to describe the region of the flexible shape. The system generates new shape configurations in the neighborhood of the current shape configuration in the process of motion planning until the goal configuration is reached. The path returned by the planner consists of a sequence of shape configurations in which the crowd will be put into simulation using fuzzy logic controllers to control the motions of the agents for desired flocking effects.

4 Path Planning for a Flexible Shape

4.1 Configuration and coverage map

We represent a region of flexible shape with a conserved volume by a fixed number of connected discrete cells in the workspace. The obstacles in the environment are also represented as forbidden cells in the workspace. , A shape configuration, denoted by S, is collision-free if and only if all of the cells in S do not overlap with the obstacle cells.

In order to facilitate the measure of incremental movement of S, we distinguish two types of cells in S: *inner cells* and *boundary cells*. A boundary cell for a shape configuration, denoted by $C_{Boundary}$, is defined as the cell having at least a free cell in its neighbors. An inner cell for a shape configuration, denoted by C_{Inner}, is the cell inside the region of shape configuration but is not a boundary cell. Examples of these two types of cells are illustrated on the left of Figure 3. Note that these two types of cells are for a shape configuration. Since each cell consists of two degrees of freedom, the complexity of the overall configuration space, growing exponential in the overall degrees of freedom is rather large. However, the spatial and temporal relations of these cells are not without constraints. We assume that the region needs to move as a connected component and the speed of the movement for each cell is limited. In the planner, we ensure that the region moves continuously by allowing

```
Procedure: Generate_New_Shape_Configuration
Input. A configuration $S$.
Output. A new shape configuration $S'$.
1.  Clone $S'$ from $S$
2.  if all cells in $C_{Boundary}(S')$ have no collision-free
      neighbors then return nil
3.  Randomly choose $C_f$ in $C_{Boundary}$ such that a
      neighbor of $C_f$ must be collision-free
4.  let $v$ be the vector from $C_f$ to the free neighbor
5.  Add all $C_{Boundary}$ cells of $S'$ into a list $Q$ sorted
      according to the distance to $C_f$
6.  while $Q$ is not Empty
7.  begin
8.      poll the front element $C_i$ from $Q$
9.      if $C_i + v$ is free in workspace then
10.         poll the rear element $C_j$ from $Q$
11.         remove $C_j$ from $S'$
12.         add $C_i + v$ into $S'$
12. end
13. return $S'$
```

Figure 4. Procedure for the generation of a new shape configuration

only one cell of difference in two consecutive configurations. Hence, we only need to consider the variation of $C_{Boundary}$ in generating a new shape configuration.

We use a data structure called *Configuration Coverage Map* (*CC-Map*) to record the trace of regions covered by the generated shape configurations. This coverage map contains three types of cells: CC_{Inner}, $CC_{Boundary}$, and CC_{Free} as shown on the right of Figure 3. CC_{Free} are the unvisited cells in the freespace. Similar to the definition of the region in shape configuration, CC_{Inner} are the set of inner cells in the covered region so far while $CC_{Boundary}$ is the set at the boundary. In order to ensure that the motion planner makes progress in each step of the search process, a new shape configuration is considered valid only if it can expand the coverage region by at least one cell. In other words, the difference of the region for a new shape configuration, S_{new}, and the existing coverage map must be at least one cell.

4.2 Generating a new shape configuration

The procedure used for generating a new valid shape configuration is shown in Figure 4. The procedure starts by duplicating the given shape configuration S to S'. Then the system checks if there exists at least one cell in $C_{Boundary}$ of S' that is collision-free. The system randomly chooses a seed cell C_f and a moving direction from the intersection of $CC_{Boundary}$ and $C_{Boundary}$ of S'. If the new neighboring cell along the chosen direction is collision-free, then we move other cells in $C_{Boundary}$ along the same direction by one cell if the new cells are collision-free. The way we move the whole region is by removing the cells at the other side of the boundary and put them in the new cells along the moving direction. Since the number of cells is not changed, the volume of the region is also conserved.

4.3 Motion planning for flexible shape

In the planner, we use a Best-First Search (BFS) algorithm, commonly used in the motion planners for low dimensional search space, to search for a feasible path. In the algorithm, the planner keeps track of the explored valid configurations that can be further expanded, called OPEN, and selects the best configuration in it to explore further according to some criteria such as an objective function. In this work, we define the objective function as the weighted sum of the following two scores. One is the distance score computed according to the average distance between the cells of the current configuration and the goal position. The other is the shape score determined by the difference of the ideal shape and the current shape.

When a shape configuration is selected for exploration, all new configurations that can extend the region of CC-Map further are inserted into the OPEN list for future exploration. The system will continue to generate new shape configurations, update the CC-Map until the generated shape configuration covered the goal position (success) or the OPEN list becomes empty (failure). If the goal configuration (any configuration that can contain the goal position) is found, we can then backtrack the search tree to obtain the path for the flexible shape.

5 Fuzzy Rules for Crowd Simulation

According to our observation of the interaction between people in our real life, we model the behaviors of the agents in a crowd with a three-layer fuzzy logic: *type of model*, *behavior model*, and *primitive action*. Each type of models contains several behavior models, and each behavior model contains primitive actions. Three types of behaviors are defined in our system: *Intra-agent*, *reactive*, and *inter-agent*. The fuzzy behaviors use rules and linguistic variables to describe the relation between sensation and actions. Each primitive action is represented by a distinct control policy governed by fuzzy inference. We use a fuzzy knowledge base (FKB) to store the fuzzy rules and linguistic variables and use a fuzzy logic controller (FLC) to control the motions of the agents. In each control loop, the system fuzzifies the position and velocity of the agents and some environmental information for the processing of FKB. Then it defuzzifies the results to obtain the control parameters driving the motions of the agents.

6 Experimental result

We use the scene in Figure 5 to demonstrate how the selection of different weights on distance and shape affect the paths generated by the planner. Key snapshots,

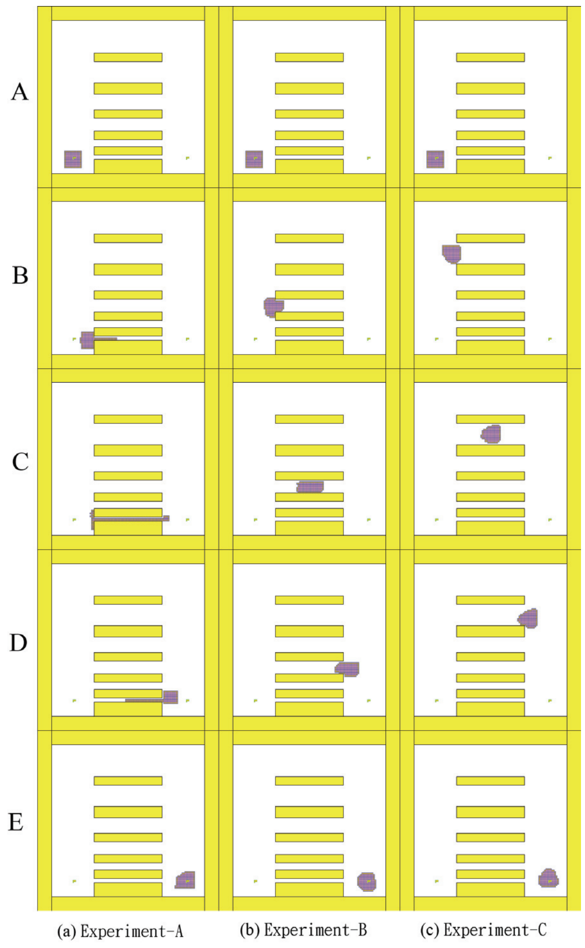

Figure 5. Key snapshots along the paths of a flexible object in various experiments

arranged from the top to the bottom, of these experiments are shown in Figure 5. The weights on the distance score and shape score are the only parameters that are different among these experiments. The weights (shape/distance) for experiments A, B, and C are (0%/100%), (25%/75%), and (75%,25%), respectively. Since distance is the only criteria used in searching for the path in experiment A, a shortest path passing through the narrowest passage was generated.

7 Conclusion

It is a time-consuming and challenging task to create the animation of a crowd motion that conforms to the environment and respects a specific shape. In this paper, we have designed a motion planner to address the problem of how to generate the path for a region with flexible shape and fixed volume. We use the concept of coverage map in workspace to ensure the completeness of the best-first search algorithm. The system allows a user to specify his/her preference on path length or ideal shape with weights on the corresponding evaluation scores. In addition to the application of crowd simulation, the configurations and paths generated by the motion planner can also be applied to liquid simulation as show in Figure 6. We have also designed a fuzzy-behavior model to facilitate the motion control of the agents inside the moving shape. More examples created with these fuzzy control rules will be reported in the future.

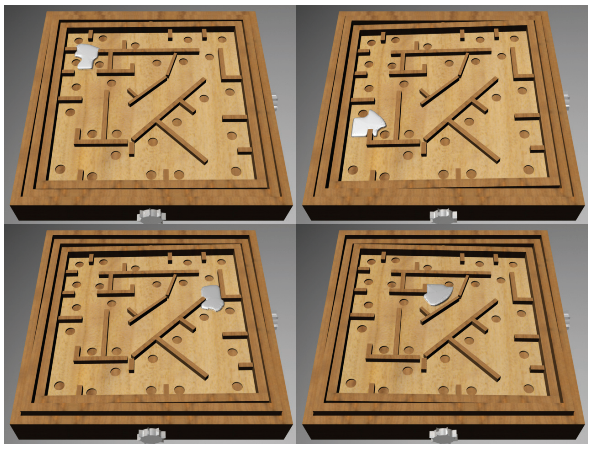

Figure 6. Applying the result generated by the planner to the simulation of liquid motion

Acknowledgement

This research was founded in part by the National Science Council (NSC) of Taiwan under contract no. NSC 93-2213-E-004-001

References

[1] Reynolds CW (1987), Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics: ACM SIGGRAPH*, pp.25-34

[2] Reynolds CW (1999), Steering behaviors for autonomous characters. In *Porc. of 1999 Game Developers Conference*, pp.763-782

[3] Anderson M, McDaniel E, Chenney S (2003), Constrained Animation of Flocks. In *Proc. of ACM Eurographics/SIGGRAPH Symp. on Computer Animation 2003*

[4] Latombe JC (1991), *Robot Motion Planning*, Kluwer, Boston, MA

[5] Bayazit OB, Lien JM, Amato NM (2002), Better Flocking Behaviors in Complex Environments Using Global Roadmaps. In *Proc. of the 2002 Artificial Life (ALIFE)*, pp.528-534

[6] Kamphuis A and Overmars MH (2004), Finding Path for Coherent Groups using Clearance. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp.193-202

[7] Tunstel E, Danny H, Lippincott T, Jamshidi M (1997), Autonomous Navigation using an Adaptive Hierarchy of Multiple Fuzzy-Behaviors. In *Proc. of Intl Symp on Computational Intelligence in Robotics and Automation*, pp. 276-281