GISMO: AN APPLICATION FOR AGENT-BASED COMPOSITION

Yuta Uozumi

Cyber Sound Project, Media Design Program
Graduate School of Media and Governance, Keio University SFC, Japan
isana137@sfc.keio.ac.jp

ABSTRACT

This paper presents an approach to new method for music composition. Traditional Western music is based on priori system. In contrast, complex systems and multi agent systems give a new perspective of linking dynamic composition with bottom-up process. "Gismo" is an application which uses multi agent system. It realizes Dynamic Composition by interactions between agents. In the style of traditional Western music, composers had to design all musical factors. But with Gismo, composers only have to design basic components as agents. Agents generate structure of music in interaction of Gismo framework. The demo movies, including all reference movies in section 3, are available on the Web. (<http://www.dubdb.com/gismo/icmc/>) Please refer to them for agents' behaviors and sounds.

1. INTRODUCTION

Gismo is an application to compose music with multi-agent system. The multi agent system is simulation technique for complex system[1]. A component of the system that behaves autonomously is called "agent". Multi agent system is bottom-up system that decides its behaviors through interactions between agents.

Although each agent behaves according to easy rules, this can generate complex behaviors as entire system. In this system, orders of the behaviors are generated autonomously. This is called "self-organization"[2].

Gismo is designed for generating music with this system. In the style of traditional Western music, composers had to design all musical factors based on its theory. However with Gismo, the composers only have to design musical-components as agents and make them interact with each other. Then, composers operate the application according to results of those interactions.

In this process, composers may find some factors hidden in their original designs, and react to it positively or negatively.

2. SYSTEM COMPONENT

2.1. Development and execution environment

This application is developed with the C and Objective-C programming language. Development environment is

Xcode 1.1 made by Apple Computer Inc[3].
This application requires OSX 10.3 above.

2.2. Model, Framework

In the field of the multi-agent system, various models have already been proposed. This application employs simple a biogeocoenosis model optimized for Gismo.

The users should understand what is happening in interactions between agents immediately, and operate the application to react to them. That is the reason why the adoption model should be simple. The model is the most simple model that connects each agent in relation of food chain. Events "chase" or "escape" occur in accordance with it.

Each agent has its own parameters such as field of vision(View), size(Size), movement speed(Mov). The algorithm of agent is as follows (Figure 1).

1. At first, agents walk around randomly in virtual world.
2. When an agent finds the others, it compares its size with the nearest one. If the other is smaller, it chases and eats the other. If the other is larger, it escapes from the other to survive.
3. When the agent attacks the others, it gets bigger (increases in SIZE parameter). On the contrary, when it is attacked by the others, it gets smaller (decreases in SIZE parameter). If the value of size decreases below the pre-defined value, the agent dies.

This is very simple, however, essential factors exist there, such as conditional judgment, positive-feedback, negative-feedback, and increase and decrease in options, etc. Since the behaviors of agents are visually apparent on this model, the users can understand what is happening in the virtual world.

2.3. Sound

In this application, the users can set arbitrary sounds to be triggered by various events(Table 1). This is done by reading and playing of sound files. Users can use AIFF, WAV and SND format sound files.

When the state of agent changes, the sound set up on each event is played. Consequently, the interactions between agents generate rhythms and/or harmony. The parameters

Figure 2. Gismo Interface

Figure 1. Basic Model

of agents influence repetitions, rhythms or harmonies. For example, in the case where agent repeats "finding and chasing" and "missing", it generates rhythm. If the MOV parameter (movement speed) of the agent is very high, it might generate the effect similar to fast BPM.

EventName	Timing
Calm	Lose sight of other agents
Escape	Escape from other agents
Chase	Chase other agents
DMG	Be Damaged
Death	Die

Table 1. Sound event chart

2.4. GUI

The interface is implemented for the framework of Gismo (Section 2.2), which is designed for users to operate the application quickly and smoothly in response to agents'

actions (Figure 2). The contents are follows.

WorldView: This area displays aspects of virtual space in Gismo. Various behaviors of agents can be seen here.

PaletteInterface: This area is the interface for defining new agent species. When the users click on "+" button, a new agent species is added and all parameters are set to their default values. The users can reconfigure these parameters here to design the new agent species. Also users can assign a sound to each event, such as escape, chase, damaged, death.

AgentPutButton: To put the agent designed within PaletteInterface into the virtual world, the users need to push this button. They can add as many agents as they want.

AgentEditor: The agents put into the world can be controlled with AgentEditor, and it affects the agents immediately.

3. SAMPLES OF IMPLEMENTATION

Let us focus on the actual examples with the application.

3.1. One to One: The simplest case

The interactions of between two agents is the simplest case in Gismo. In this case, two identical agents are put into the world.

When each agent finds the other, it chases and tries to eat the other. In another second, winner and loser are decided. Consequently, the loser escapes, and the winner chases. This relation is static in this case.

Another notable matter is "loop of the world". Gismo processes all sides of the virtual world as loop (Figure 3). Consequently, it generates pseudo metrical structure based on loop structure built by the agents that move endlessly in loop of the world.

In other words, the escaping agent comes at the corner of the world, and then it moves to the counter side by the loop process. At this time, they lose sight of each other. Next, the chasing agent comes at the corner too, and then it moves to the counter side in the same way, and they find each other again. Then Gismo plays sound which was assigned to this event beforehand.

If this process is repeated, it can generate looped sound.

Figure 3. Loop of Virtual World

3.2. Many-body case

Let us discuss many-body case. In this case, a number of agents are put into the world according to the following composition (Table 2). We see from table 2 that the

AgentName	Size	Mov	View	Num
lower	10	50	100	8
medium	30	30	80	4
higher	60	10	60	2
Trialist	10	300	1000	1

Table 2. Example of many-body composition

smaller agents get, the more and the faster they are, and that the larger they get, the fewer and the slower they are. However, the agent "Trialist" is an exception. It is an examinee agent that is especially set. Trialist's "size" parameter is the same as that of "lower" agent, but the other parameters such as movement speed and field of vision are different, and they are much higher than those of the other agents. Thus, Trialist has advantages, and consequently the agent can eat the other agents with its advantages and becomes larger (Figure 4- 1). Before long, the agent can eat "medium" or "higher" agent which were bigger than it at the first(Figure 4- 2, 3).

Moreover, Trialist agent is set for playing white noise when it is chasing after the other agents. Therefore, as the agent grows and its option to eat the others increases, repetition of white noise increases. Also, by setting Trialist to play decaying sine wave when it escapes from the larger, we can see the decrease of its threat as repetition of this sound decreases.

Of course, we can set some other sounds instead of white noise or sine wave. For example, it is possible to generate harmony by assigning sounds that contain note C or G to the agents.

Figure 4. Growing Agent

3.3. Beat generation

AgentName	Size	Mov	View	Num
Ride	20	300	200	12

Table 3. Beat agent

Improvisation music like jazz uses fluctuation beats which, for example, ride cymbal generates. Gismo can generate it with agent's self-organization.

In this case, "Beat agents" are employed (See Table 3). If twelve (or more) Beat agents are put into the world, Gismo can generate fluctuation beats.

The features of this agent are "ultra-narrow field of vision(View parameter is very low)" and "Very fast movement speed(Mov parameter is very high)".

Beat generation process is as follows.

1. At first, Beat agents can not find each other because of their ultra-narrow field of visions, therefore, they walk around randomly.
2. Before long, one agent finds the other and then chases or escapes. According to this event, sound is played.
3. However, they lose sight of each other easily because they have too narrow view of field and move too quickly.
4. If escaping agent enters the field of vision of another agent, this sequence of interactions occurs again.

This simple process brews up a chain of interactions. As a result, it generates complicate fluctuation beats. Furthermore, as a result of that, agents self-organize so that they keep the constant distance among them. (Figure 5).

Figure 5. Self-organization by Beat agents

3.4. Implementation of Swarm

Gismo can make agents swarm (See fig 6). Interaction speed of swarm algorithm is so fast that it can realize granular synthesis[4].

Boids by Craig W. Reynolds is known for model of swarm or flock[5]. However, Gismo adopts simple algorithm like fly's behavior. This method has a defect that swarm breaks up more easily than the method of Boids. Instead, it is compatible with the biogeocenosis model of Gismo and can process more rapidly.

Swarm algorithm is as follows.

1. An agent compares its species with the nearest agent and checks whether it is the same or different.
2. When they are the same species, the distance between them is compared with the fixed value of system.
3. If the distance is greater than the value, they get close to each other, or vice versa. (Then gismo plays sounds in response to these events.) Therefore, those agents keep constant distance based on the fixed value.
4. Consequently, the agents generate swarm behavior like flies.

This algorithm is very simple, however, it generates complicate behavior with the interactions between agents. In this case, if we assign short sound files(30 - 120 ms) to agents, we can get interesting timbres. However, granular synthesis is too heavy for the present version of Gismo to generate sound with it. It is a next task that is necessary to optimize the sound implementation.

Figure 6. Swarming agents

4. CONCLUSION

The application for composition using multi agent system is characterized as follows.

- (A) Shift from the old model that depends on concept and music theory to the new model which make discovering and expanding of concepts and theory dynamic.
- (B) The composition style that operates relationships between agents by manipulating parameters and behaviors of them.
- (C) Shift from the static components such as score, players and instruments to the interactions of sounds by the medium of agents.

Given this factor, Gismo adopts the method which builds up its entire structure in the feedback model, that repeats these three processes; designing of components, interactions between agents and discovering of emergences in interactions by the users. Eventually, the agent-based composition changes the established composition system into the more dynamic and flexible system.

5. REFERENCES

- [1] Takashi Iba, Yoshihisa Fukuhara. *Complex System hornbook: The adventure of intelligence*. Tokyo, NTT publishing, Japan, 1998.
- [2] Stuart Kauffman. *At Home in the Universe: The Search for Laws of Self-Organization and Complexity*. New York: Oxford University Press, 1997.
- [3] Apple Computer, Inc. hXcodeh. Apple Developer Connection. ionlinej, available from <http://developer.apple.com/tools/xcode/r>, iaccessed 2004-12-12j.
- [4] Curtis Roads. *Microsound*. Cambridge, Massachusetts and London: The MIT Press, 2001.
- [5] Craig W. Reynolds, *Flocks, Herds, and Schools: A Distributed Behavioral Model*, in Proc. SIGGRAPH87, July. 1987, pp. 25-34.