論文
2-10-8

유전자 알고리즘을 사용한 지능적인 실시간 게임 캐릭터 (Intelligent Real-time Game Characters using Genetic Algorithms)

안 태 홍* 강 성 관** 이 상 규*** 김 우 정**** 김 홍 기*****
(Tae-Hong Ahn) (Sung-Kwan Kang) (Sang-Kyu Lee) (U-Jung Kim) (Hong-Ki Kim)

요 약

애니메이션과 컴퓨터 게임에서 오늘날 주요 관심사는 캐릭터의 행동을 미리 정의된 게임 로직이나 미리 설정된 동작에 의해 제어된다. 게임 개발자들은 더 풍부하고 더 상호보완적인인 게임을 제작하고자 하기 때문에 자주 이러한 접근은 한계에 부딪힌다.

본 논문에서는 더욱 지능적이고 강제적인 컴퓨터 게임을 만들기 위해서 유전자 알고리즘을 사용한 게임 모델을 제안한다. 학습능력에 기반한 유전자 알고리즘의 사용은 변화와 능동적인 게임 환경을 지원하도록 연속적인 진화를 하는 캐릭터를 활용할 것이다. 임의의 실시간 게임이 제안된 시스템의 수행과 제한성을 평가하기 위해 수행되었다.

ABSTRACT

In the majority of todays animation and computer games, the behaviours of characters are controlled by pre-defined game logic or pre-generated motion. As game developers strive for richer and more interactive games, they often encounter limitations with this approach.

This paper attempts to construct a game model using Genetic Algorithms (GAs) in order to produce more intelligent and compelling computer games. Based on learning ability, the use of GAs will enable the characters to continually evolve, providing a changing and dynamic game environment. A real-time game was implemented to investigate the performance and limitations of the system.

1. Introduction

The main challenge facing game developers today is to create increasingly rich, more compelling experiences for the game player. In most of todays animation and computer games, the movement of characters is pre-defined by rules [2] [12] [14] or pre-generated using a key-frame based animation system or motion capture.

However, as game developers strive for more interactive and more intelligent games, these approaches encounter some severe limitations:

* 정회원 : 전남과학대학 컴퓨터 게임제작과 학과장
** 정회원 : 전남과학대학 겸임교수
*** 정회원 : 전남과학대학 컴퓨터게임제작과 겸임교수
***** 정회원 : 동신대학교 컴퓨터학과 부교수

논문접수 : 2001. 10. 9.
심사완료 : 2001. 10. 20.

**** 정회원 : 정보통신부지원 IT 분야 겸임교수

- *Limited interactivity:* since all character actions are pre-defined or pre-generated, interactions between game characters, game environment and game players are limited. As a result, this can detract from the overall realism of the game.
- *Unmanageable complexity:* the development of more elaborate games brings an increased likelihood that the complexity of the game logic will become unmanageable [1] This also requires a great deal of CPU process time and memory, reducing the performance of game playing, especially in a real-time game.
- *Labour and cost intensive:* generating all game algorithms and movement sequences for a game character, particularly in a dynamic environment, is time consuming and costly.

Increasingly, Artificial Intelligence (AI) technologies are being developed and applied to commercial games. However these methods are mainly based on pre-defined rules or behaviour patterns and the above limitations are still substantial.

This paper will review the current approaches to AI technologies in computer games. An intelligent game model using GAs will be constructed and described in detail. Through the implementation of the GAs concept within a real-time test game, the performance and limitations of the model will be observed and analysed.

2. Intelligence in a Computer Game

In general, game players enjoy games which offer the experience of realistic, stimulating and exciting games [7]. In other words, they want to play with intelligent opponents in a realistic environment. However, pre-defined game algorithms often limit the interactions between game characters and the game player or between the characters and the game environment. The more familiar players become with the game logic the more chance that they will loose interest. Artificial intelligence in computer games implies that the computer-controlled characters are able to exhibit some kind of cognitive functions when taking actions or reacting to the player's actions. Currently two major approaches to Game AI are Rules-based AI and Behaviour Pattern.

2.1 Rules-based Game AI

Rules-based AI remains the foundation of opponent intelligence in most games. A rule usually consists of a condition (stimulus) and an action (response), such as if chased by a predator, then run away. This approach generally includes the finite state machine (FSM) and the fuzzy state machine (FuSM) [17].

2.1.1 Finite State Machines (FSM)

An FSM [9] is currently the most common AI technology in use. It is a logical hierarchy of rules in a fixed number of possible states. Each character is represented as an object, which has an internal state. Depending on events (stimuli) in the outside world, the object will change to a different state, and reflect this in how it behaves (response).

2.1.2 Fuzzy State Machines (FuSM)

The FuSM is similar to the FSM except that a given set of stimuli maps, not only to a specific response, but also to a set of possible responses [17]. Incorporating ideas from fuzzy logic, this method brings great flexibility into the games domain, especially where the condition is distinct such as true or false. This method can therefore generate a variety of responses to a given set of stimuli. For example, if the opponent is too strong, run away. Here the words too strong are simple to judge in terms of true or false and there could

be many possible responses to the situation. A variety of methods have been developed to select an appropriate response, such as probability weights, or threshold to trigger an action.

2.2 Pre-defined Behaviour Patterns

Pre-defined behaviour patterns are often used to provide intelligence into computer games. A pattern is a sequence of steps to perform in order to accomplish a task, and can be triggered when certain conditions arise. For example, when a person drives to his/her house, whether aware of it or not, he/she follows a pattern: get into the car, start the car, drive to the destination, stop the car, turn it off, get out, and so on. This is a pattern of behaviours and is a good way to implement seemingly complex thought processes in game AI. In fact, many games today still use such patterns for much of the game logic [11].

Both rules-based AI and behaviour patterns, however, have significant limitations in terms of flexibility and variety. The preparation of all the possible rules and patterns are greatly limited in practice. Even when many behaviour patterns or rules are prepared, it is difficult to control the large number of possible combinations with the use of such algorithms.

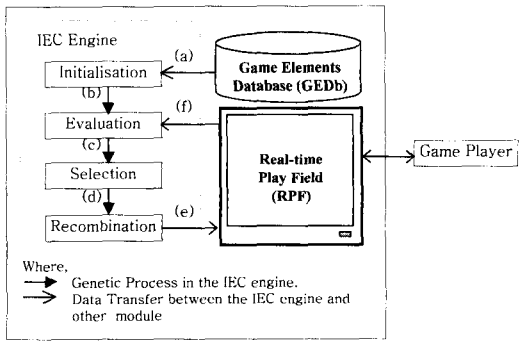
3. GAs in Computer Games

A number of game developers have started investigating the use of genetic algorithms (GAs) in their games. The main applications of GAs in computer games today are concentrated on strategic games such as chess [13], cards [6] [15], politics [16], and some economic strategy games [4], rather than real-time games. The two most successful applications of GAs in recent games are Creatures and Cloak, Dagger, and DNA (CDDNA).

Creature could be categorised as a simulation, like Sim, rather than a genuine computer game because it does not involve a clear goal and lacks the competition aspect, both of which are essential characteristics of a game. It is a type of strategy game rather than a real-time game that is the focus of this paper. Oidian System's Cloak, Dagger, and DNA (CDDNA) is another strategy game making use of GAs to create smarter opponents. The goal is to conquer the game board by capturing more factories in order to support the players armies. Players do not actually move their game pieces, but submit a list of orders for the pieces to move, build or disband. The computer players play the game by running a special program, a gene, which generates the same kind of orders.

4. Intelligent Evolutionary Game

Based on the characteristics of GAs in computer games, an Intelligent Real-time Game (IRG) Model was constructed. Using the learning ability of GAs [5] [8], this model provides for continuous evolution of the characters which are then able to learn from their environment, adapting to it and becoming better equipped to survive.



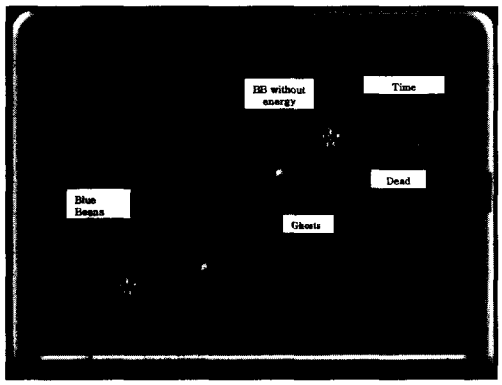
[Fig. 1] The IRG process model.

The IRG model was constructed based on GAs and the object-oriented data structure. The overall system consists of three major modules: Game Elements Database (GEDb), IRG Engine, and the Real-time Play Field (RPF), [Fig.1] The GEDb contains all the information about the game and supplies it to the IRG engine and RPF. The IRG Engine module receives information about the game environment and characters, along with evaluation criteria. Using this information the evolution process can proceed. Subsequently, through the evolution process, a population of new characters is generated to replace the previous generation. This engine is the only problem-independent module in the model. Therefore it can be applied for different types of games. This module uses the research outcome of both the evolutionary process and genetic representation of individual behaviours from the EDGE system [10]. The RPF is the actual working area of the game play. This is where the game players play the game. The game environment is defined by the initial game logic. However, as the evolution proceeds, the game story changes and expands.

5. Implementation

The IEC model was implemented for a simple 2-D real-time game in order to simulate the performance of the GA-based game engine. The IRG engine is derived from the EDGE system, originally implemented in C, and trans-coded to Macromedia Director script language for easy user interface design. This game (see [Figure 2]) involves two kinds of characters. There are Ghosts and Blue Beans: it consists of two Ghosts, or predators, hunting ten Blue Beans (BB), which are weak but, in game terms, they are the good characters . A deterministic algorithm to pursue BBs controls the Ghosts. The BBs use GA-based AI to run away from the pursuing Ghosts. Hence the goal of the game is for the BBs to survive as long as possible. The fitness of each BB is assessed by the amount of time they survive.

The current version of the game runs automatically without any interaction from the player. When human players play this game, the IEC engine can be applied to the computers characters, ostensibly the two Ghosts.



[Fig. 2] An Example game based on the IRG model. The two Ghosts chase ten BBs. A BB becomes pale when it runs out of energy.

5.1 Evolving the Blue Beans (BBs)

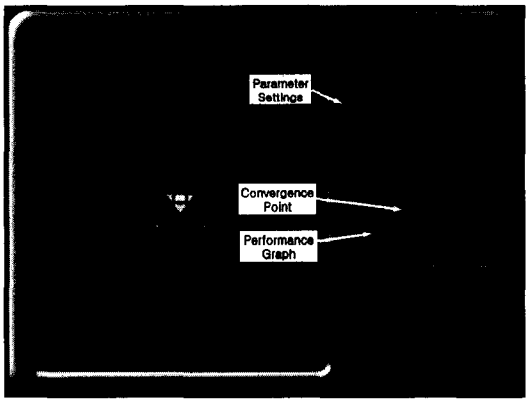
Each BB object has three self-controllable behaviours: to decide when to start running (considering the distance from a Ghost); to turn in any of eight directions and to speed (the distance of a step). They do not have any other knowledge except the three behaviours. Each BB has a limited (not controllable) energy supply and every movement consumes a certain amount of energy, according to the speed and running distance. Therefore, every BB needs to control its movement economically. For example, the BBs should avoid unnecessary movement to save energy when not being chased. The BBs do not know any strategies in the early stage. As the game proceeds, they start learning from experience. The evolutionary process keeps changing genotypes of individual characters and makes the genotypes better structured and more intelligent.

5.2 The Ghost Characters (Ghosts)

The Ghost characters behave, or move, based on a FSM. In the RPF, a Ghost looks for a BB, follows it and kills it, then it looks for another BB. The basic structure of stimuli, internal states and responses in a Ghosts object is as follows:

- if ready, look for a BB
- if a BB is found, chasethe BB
- if a BB is caught, kill the BB
- if the BB is dead, go to state ready

5.3 Learning Performance



[Fig. 3] GAs settings and the Performance Graph. The graph shows the improvement of average survival time, or intelligence, of BBs.

The test game started with the settings of GAs parameters as in [Fig. 3].

The parameter settings in Figure 3 show that the process starts with ten BBs to run over fifty generations or stages. In every Recombination process, each character will have about 90% chance of mating with another one. And for each gene the chance of mutation taking place is approximately 3%.

The following is a summation of what happened during testing: When the evolution started, the average fitness, or intelligence, of characters was very low (38 seconds survival time). As the evolution process proceeded, the intelligence increased dramatically (340 seconds survival time by the 15th generation), Figure 3. The best fitness value throughout the 50 generations was 461 seconds. Genotypes of superior classification are saved in the GEDb and can later be used as well trained or intelligent characters. The improvement stopped when the process reached a convergence point. This convergence problem is quite a common but critical problem of general GAs applications. The performance fluctuates a little over the whole process. However, if necessary this obstacle can be overcome by preserving the best population.

6. Conclusion

This paper introduced GAs into a real-time game engine in order to incorporate more intelligent and efficient game AI. Existing game AI approaches have been reviewed and their limitations discussed. The process and representation of general GAs were modified and applied in the IRG engine. The model was implemented using a test game and its performance was analysed.

The results of the research and implementation show very promising outcomes and beneficial aspects of the new model as well, as the use of GAs in game AI.

- On the whole, a pre-defined game logic tends to result in computer games being inflexible and inefficient when adapting to different situations [3]. The IRG model however does not necessarily require any pre-defined rules to control the character behaviours. Instead, the gradual process of evolution enables characters to adapt themselves to the changing game environment and to improve their intelligence. This presents a more economical and flexible way to achieve intelligent game AI.
- In general, the use of GAs can not guarantee an optimal solution. However, the simple but robust algorithms work very well for large and complex problems and give near optimal solutions. This feature provides an opportunity for this game engine to be applied for use in a variety of complicated situations.
- Since the new model does not require pre-defined rules or behaviour patterns, this model has the potential to significantly reduce the workload of game designers and programmers in preparing the rules and patterns. The whole process of game development will become faster and more economical.
- GAs involve randomness, which is one of the most essential behaviours in a computer game. A random behaviour often makes it harder for the player to anticipate the computer characters next moves, and is a good method of helping the characters make a selection when there isn't enough information to use a deterministic algorithm.
- Using GAs can control the level of difficulty. At the start of a game, the level of difficulty is easy for the player because the game characters are virtually a new species and are in almost total ignorance of their environment. However, as the game progresses, the difficulty level increases.

Despite the many promising aspects of GAs and the IRG model, the model still needs to overcome some fundamental issues of GAs such as premature convergence, control of evolution speed, population-based operations, generic knowledge representation mechanism. Some of these issues may be resolved by combining the IRG model with other existing technologies. By doing so the system could preserve computation time which would otherwise be used in a trivial, perhaps less important evolutionary process. Also the use of Neural Networks may be an advantage by enhancing the power of the IRG model and thus allowing individual self-learning.

※ REFERENCES

[1] Abramsky, S. and Mellies, P.: Concurrent games and full completeness, LICS99, Trento (1999)

[2] Badler, N.I., Barsky, B.A. and Zeltzer, D. (ed.): Making them move: mechanics, control, and animation of articulated figures. Morgan Kaufmann, San Mateo (1991)

[3] Blass, A.: A game semantics for linear logic, Annals of Pure and Applied Logic, 56 (1992) 183-220

[4] Condon, A.: Computational Models of Games, MIT Press, UK (1988)

[5] DeJong, K. and Spears, W.M.: An analysis of the interacting roles of population size and crossover in genetic algorithms, in H.P. Schwefel and R. Manner (eds.), Parallel Problem Solving from Nature, (1990) 38-47

[6] Dietterich, T.G.: Applying General Induction Methods to the Card Game Eleusis. AAAI 1980: (1980) 218-220

[7] Duntemann, J.: Breathing life into your arcade game sprites, PC Techniques, 5(2) (1994) 89

[8] Grefenstette, J.J.: Optimisation of control parameters for genetic algorithms, IEEE Trans SMC, 16 (1986) 122-128

[9] Hopcroft J. E., Ullman, J. D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, Reading, MA (1979)

[10] Jo, J.H. and Gero, J.S.: Space layout planning using an evolutionary approach, Journal of Artificial Intelligence in Engineering, 12(3) (1998) 149-162

[11] LaMothe, A.: Building Brains into Your Games, Game Developer Magazine, August (1995)

[12] Maes, P. (ed.): Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. MIT Press, Boston (1990)

[13] Reif, J.H.: The complexity of two-player games of incomplete information, Journal on Computers and System Science, 29(2) (1984) 274-301

[14] Tu, X.: Artificial Animals for Computer Animation: Biomechanics, Locomotion, Perception, and Behaviour. ACM Distinguished Ph.D Dissertation Series, Springer-Verlag (1999)

[15] Tzeng, C.H. and Purdom, P.W.: A Theory of Game Trees. AAAI 1983: (1983) 416-419

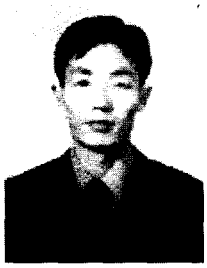
[16] Vincke, S.: The ORC problem, Game Developer (1998)

[17] Woodcock, S.: Game AI: The State of the Industry, Gamasutra, 2(46) (1998)

안 태 홍

2월 조선대학교 전자공학과 (공학사)
1992년 8월 조선대학교 대학원(공학석사)
1998년 2월 조선대학교 대학원(공학박사)
1993년 ~ 전남과학대학 컴퓨터게임제작과 학과장
2001년 ~ 정보통신연구진흥원 평가위원
관심분야 : 이미지프로세싱, 인공지능, 패턴인식, 게임엔진, 가상현실, 애니메이션 등

강 성 관



1997년 2월 조선대학교 전자공학과(공학사)
1999년 2월 조선대학교 대학원(공학석사)
1999년 3월 ~ 조선대학교 대학원(박사과정)
1999년 3월 ~ 2000년 2월 조선이공대학 겸임교수
2001년 3월 ~ 전남과학대학 겸임교수
관심분야 : 이미지프로세싱, 인공지능, 패턴인식, 게임엔진, 가상현실, 애니메이션 등

이 상 규

1993년 ~ 2001년 8월 전북대학교 (공학사)
1998년 ~ (주)제스턴 근무
2001년 3월 ~ 전남과학대학 컴퓨터게임제작과 겸임교수
관심분야 : 게임엔진, 게임기획, 가상현실, 인공지능, 게임제작

김 우 정

1996년 2월 광주대학교 산업디자인과(이학사)
1997년 ~ (주)마이크로코리아 근무
2001년 3월 ~ 정보통신부지원 IT분야 겸임교수
관심분야 : 그래픽, 웹프로그래밍, 애니메이션, 3D 모델링

김 홍 기

1984년 전남대학교 계산통계학과(이학사)
1986년 전남대학교 대학원 (이학석사)
1996년 전남대학교 대학원 (이학박사)
1991년 ~ 동신대학교 컴퓨터학과 부교수
관심분야 : 공간데이터베이스, 컴퓨터그래픽스, 멀티미디어시스템