International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064

# Automatic Game AI Design

**Pranavkumar Pathak[1], Abu Sarwar Zamani[2], Dipthi Shah[3]**

[1]CCIS, King Saud University, Kingdom of Saudi Arabia

[2]College of Science & Humanity, Kingdom of Saudi Arabia

[3]Dept. Computer Science, Sardar Patel University, India

**Abstract:** ***Video game AI aims at generating an intelligent game opponent which is to compete with player, so game AI design plays an important role in the development of game. Nowadays, most of the game AI is implemented by FSM. But this mechanism has some drawbacks, so we need a mechanism to design game AI automatically instead of FSM. The process of automatic game AI design by UCT is introduced in this paper. In this process, we only take the meta-rules into consideration, while those many complicated detail knowledge is acquired by simulation. Here we propose the approach of UCT-controlled NPC based on CI (computational intelligence). However, this approach will consume lots of computational resources, and the acquired knowledge cannot be stored. To solve this problem, we train Artificial Neural Network (ANN) to make it reusable. The whole design process is validated on the Test-Bed of the game Dead-End. We conclude that from both the simplification of implementation and the reusability, this process outperforms FSM.***

**Keywords**: AI for games, Steer behavior, Path finding in 3d games and simulation

## 1. Introduction

The goal of video game AI is to generate AI that is not only challenging, but also satisfactory. Most of the existing game AI is implemented by FSM. There is no doubt that FSM can easily produce challenging game opponent with good intelligence, however, it does not always provide the best solution. FSM-controlled-NPC game AI generation approach does have some drawbacks compared to the approach of CI-controlled-NPC as shown in Table 1

**Table 1:** Comparison between FSM and CI

| | *FSM-Controlled NPC* | *Ci-Controlled NPC* |
|---|---|---|
| Programming | Low-level coding | Meta-programming |
| Length of code | 286 rows | 98 rows |
| reusability of code | impossible | possible |

To create a challenging game opponent, we propose the approach of UCT-controlled-NPC based on CI, which is regarded as an automatic AI design and development technique. Just opposite to the FSM-controlled NPC game AI generation approach, the CI-controlled NPC approach has many advantages.

CI method relies very little upon the participation of human since it requires little domain knowledge, and it doesn't need manually hard-coding for every single detail. Complex domain knowledge and strategies can be automatically acquired from a large number of computations. So the approach of AI design by CI is also considered an automatic game design and development technique. As a result, the performance of the game AI created from CI is not limited by the domain knowledge of the game developer.

CI-controlled NPC uses the meta-programming, which requires only the definition of a number of meta-rules, without the need of low-level coding of details, therefore the workload of the game developers is greatly reduced. CI-controlled NPC can plan and look forward. Game strategies are formed automatically, instead of being created by developers according to domain knowledge. Thus the intelligence of game could be enhanced automatically. Although the proposed CI-controlled NPC is superior to FSM-controlled NPC as discussed above, the following limitations prevent it from being used for multi-player online games. CI approaches usually consume a large quantity of computational resources, such as CPU and RAM. As a result, it is only applicable for standalone PC games, rather than multi-player online games.

To solve the above problem, we train ANN to store the knowledge acquired by CI-controlled. ANN is stored in the form of a group of weight. When we play the game, once identifying the player using a familiar strategy, we will respond to it through corresponding ANN. Game AI obtains the direction of move by the weighted value of neural network. In this way, we successfully transfer intelligence into solid knowledge.

## 2. Test- Bed of the Game Dead end

Here we take the game Dead-End [He08] as an example, to explain the mechanism of Automatic Game AI Design In the game Dead-End, there are 1 CAT (player-controlled) and 2 DOGs (game AI-controlled) (initial location see Figure 1), moving vertically or horizontally within a 20x20 grid map area. For each step, CAT walks two cells without retreating or remaining static; while the DOG walks one cell. As Cat moves faster, it is necessary for two DOGs fully cooperated in order to beat CAT (player).

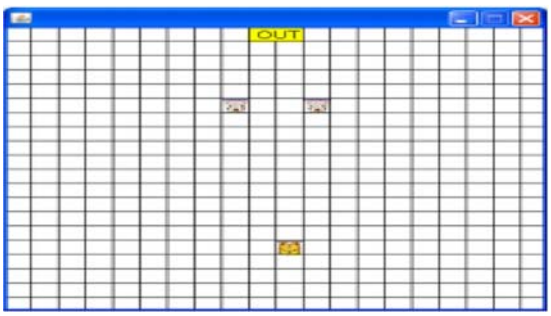
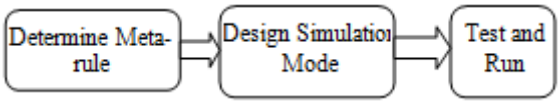



**Figure 1:** Test-Bed of Dead-End

There is just one EXIT in the north of the map called 'OUT'. If any of the two DOGs catches the CAT within 20 steps without letting it escape from the EXIT, the DOG wins (NPC or the opponent wins), while if CAT reaches EXIT within the 20 steps without being caught by DOG, the CAT wins (the player wins).

Generating Challenging Game Opponents by Automatic Game AI Design




## 3. Generating Challenging Game Opponents by Automatic Game AI Design

### 3.1 Determine Meta-rule

Meta-rule is the most basic and vital rule in game. It expresses the fundamental standard to judge the winning and losing of the game. A meta-rule should have the following characteristics.

- Meta-rule does not contain any rule related to the control of game AI, which means that it cares only the final result rather than the way game AI chooses.
- Meta-rule is the fundamental rule set for the game, that is to say, both the player and the game AI should follow it. Only under meta-rule can they choose their own way of moving.
- Next we will give the meta-rules we set for the game Dead-End:
- CAT can walk two DOGs catch the CAT( DOG and CAT overlap) within 20 steps without letting it escape from the EXIT, the DOG wins.
- If CAT reaches OUT within the 20 steps without being caught by DOG, The CAT wins.
- If within 20 steps, neither does DOG catch CAT, nor CAT reaches OUT, it is a tie.

### 3.2 Design Simulation Mode

The so-called computer simulation is that we let the computer simulate those situations that might happen in real games. These situations could be not only complicated, but also numerous if they were considered by humans. However, it is relatively easier work for computers. After computer simulating various sorts of path, we store the paths that can win the game, while wiping out those lose the game. This process makes the game AI gain intelligence. The computer simulation modes we indicate here are mainly CI approaches. CI approaches like MCTS (Monte Carlo for Tree Search) [2] and UCT (Upper Confidence bound for Trees) [4] are used in the computer Go and have received good results. However, few studies are found of their use in video games. For Dead-End, the mode we use is UCT-controlled NPC.

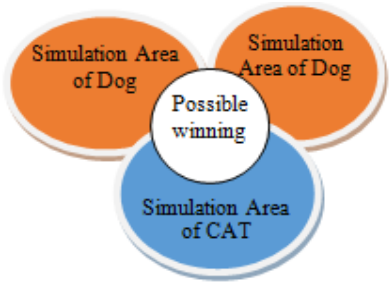



**Figure 2:** Diagram of Simulation Area

#### 3.2.2 The details of UCT-controlled NPC

UCT control of NPC on game Dead-End follows the procedure of MCTS [Chaslot08] and it make the UCT generalized to the specific game genre. Within the limit of simulation time, a large number of simulations might be happen, for example, in one case, the number of simulations reaches almost one thousand times with one simulation time duration of 300 ms.

- Selection
  During each turn, the alternative choices for DOGs are among West, East, North and South; the alternative choices for CAT are among West, East, North and Static (as retreat to south is prohibited in order to have a quick simulation result, Static is chosen instead). In the first step for the first turn, DOG1 randomly selected a branch of West from the trunk in the probability of 1/4 (0.25), which is a pure Monte Carlo method.

- Expansion
  In the first turn, after DOG1 selected the branch of West from the trunk, DOG2 expended the branch of West with 4 Simulation Area Of DOG1 Simulation Area Of DOG2 Simulation Area Of CAT Possible Winning 3849 alternative choices and randomly selected North in the probability of 1/4; CAT expended and make first move of North randomly in the probability of 1/4, then second move East randomly of in the probability ¼.

- Back propagation
  In the next turn, the sub branch is expanding according to the above procedure, until the end of the game is reached (leaf of the tree): either the result of win or lose is got, or turn limit for simulation is reached (in this case the limit is 20). The selected trunk branch is added 1 with a win result, and 0 with a lose result.

## 4. Convert into Knowledge

CI-controlled NPC makes game AI intelligent through computer simulation, but this process is achieved at the expense of consuming a large quantity of computational resources. Therefore, as we mentioned above, this method just suits the standalone PC games, rather than multi-player online games.

To solve the above problem, we search for a method which is able to store the intelligence gained by computer simulation, and through this way, we hope that when encountering the same game environment, we can apply these intelligence directly instead of calculating via

computer simulation again. ANN (Artificial Neural Network) provides us with the solution. We collect the generated data of the game between one-strategy-controlled player and UCT-controlled NPC, which stores the intelligence. Through the training of neural network using the collected data, the intelligence is stored. So next time when playing game against the player with the same strategy, we can call the neural network directly.

The design flow of the neural network for the game Dead-End is given in Fig.4.

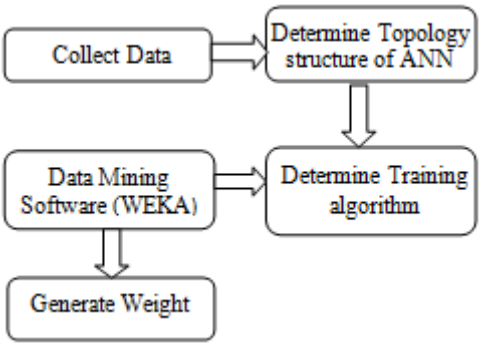



**Figure 4:** Design flow of the neural network

In the game Dead-End, we collect data of the three kinds of strategies for CAT, and train neural network using those data. CAT is controlled by FSM, while DOG is controlled by ANN. We play 200 games with each strategy, and the statistical winning percentages for DOG are shown in the Table 3.

**Table 3:** Winning Percentage of Ann- Controlled- Gog

| | *Winning Percentage* |
|---|---|
| *CAT-twostate* | *98%* |
| *CAT-zig* | *80%* |
| *CAT-S3* | *92%* |

It can be concluded that ANN-controlled DOG well store the intelligence that is gained from UCT-controlled DOG. Till now, we successfully transfer the intelligence into knowledge.

## 5. Conclusion

Automatic Game AI Design plays an important role in the development of video game to some extent. From the aspect of the length of code, the code of Automatic Design is much shorter than that of FSM. Automatic Design doesn't need to account for those minute and complicated details, but let computer simulation takes over the work instead. From the aspect of the complexity of logic, FSM needs the participation of human to design some complicated logic, which consumes more than Automatic Design does both in mental and physical aspects. Moreover, from Table 2, we've proved that automatic game AI design through UCT is feasible. Automatic Design can be used for different games rather than just one particular game. Though the design flow proposed in this paper is only proved by the simple game Dead-End, this game contains basic elements of those complicated game. We ought to believe that Automatic Game AI Design will promote the development of video game.

## References

[1] Barwood, H. & Falstein, N. 2002. .More of the 400: Discovering Design Rules.. Lecture at *Game Developers Conference*, 2002. Available online at: http://www.gdconf.com/archives/2002/hal_barwood.ppt
[2] Church, D. 1999. .Formal Abstract Design Tools.. Game Developer, August 1999. San Francisco, CA: CMP Media. Available online at: http://www.gamasutra.com/features/19990716/design_tools_01.htm
[3] Hunicke, R. 2004. .AI Babysitter Elective. Lecture at Game Developers Conference Game Tuning Workshop, 2004. In LeBlanc et al., 2004a. Available online at: http://algorithmancy.8kindsoffun.com/GDC2004/AITutorial5.ppt LeBlanc, M., ed. 2004a. .Game Design and Tuning.
[4] Workshop Materials, Game Developers Conference 2004.Available online at: http:// algorithmancy. 8kindsoffun.com /GDC2004/
[5] LeBlanc, M. 2004b. .Mechanics, Dynamics, Aesthetics: A Formal Approach to Game Design. Lecture at Northwestern University, April 2004.

## About Authors

**Pranav Kumar Pathak** has done Masters from Sardar Patel University, India. He is currently pursuing PhD in Computer Science (AI) from Sardar Patel University, India. I have also done MBA IT from Manipal University, India. Currently I am working as Lecturer in Computer Science Department at College of Computer and Information Sciences in King Saud University, Kingdome of Saudi Arabia. I have presented several papers at international Journals and conferences.

**Abu Sarwar Zamani** was born on 15th Jan 1982 in Bihar, India. He is working as a Lecturer in Department of Computer Science in the College of Science & Humanity in Shaqra University, Kingdom of Saudi Arabia. He worked in Department of Computer Science in King Saud University, KSA. He has received his Master of Science in Computer Science from Jamia Hamdard (Hamdard University), New Delhi, India in the year of 2007, and later he did Master of Philosophy in Computer Science from Vinayak Mission University, Chennai, India in 2009. He has actively attended and published various research papers in National as well as International.

**Dr. Dipti M. Shah** has done Ph. D in Computer Science from Sardar Patel University, India. Currently I am working as Associate Professor in Computer Science Department at G.H. Patel Pg Department of Computer Science & Technology V. V. Nagar. I have presented several papers at international Journals and conferences.