Researching AI Technologies Created in Japan in the 1980s and 1990s

Youichiro Miyake
DiGRA JAPAN, y.m.4160@gmail.com

Abstract

Reconstructing the history of video games will have a positive influence on game development in the industry. This paper will reconstruct ‘80s and ‘90s AI technologies used in the Japanese game industry, based on interviews with developers and surviving design documents. Most ‘80s and ‘90s game development documents have not been preserved, with a few notable exceptions. In general, the history of ‘80s and ‘90s game development in Japan is not well documented, and this is also the case for the history of in-game AI. Over the past three years, I recorded oral histories from game creators who created AI for games in the ‘80s and ‘90s, in order to reconstruct the history of in-game AI development in Japan. Some interviews were for DiGRA JAPAN Oral History Task Force, and others for a video game website. The interviews are regarding enemy AI in “PAC-MAN” (NAMCO, 1979), the “Job-Controller” system made for NAMCO’s game engine around 1980, and AI for the Mahjong game “Gambler Jiko Chushinha” (GAME ARTS, 1987). Specific examples of AI application are explained, and their results are connected in order to make one historical flow of in-game AI techniques.

1. Background

There are two methods that can be used to make an NPC (Non-Player Character) move. One method is to control the character using the game system. The game system controls not only NPCs, but all the objects shown on-screen. The other method is to make the character autonomous. An autonomous character has its own sensors that gather information from the game environment, a decision-making module, and a body. An autonomous character can sense the environment around it, make decisions, and move its body by itself.

Most game design in ‘80s video games uses NPCs as moving game objects. The NPC operates only in a set space, and has a simple role in relation to the player, such as attacking, talking, protecting, or escaping. The game system controls NPCs using script language, so controlling NPCs in this way is called scripted AI. Scripted AI is generally used to create the character control systems used within game systems.

The change from scripted AI to autonomous AI was a crucial development in the history of character AI. In modern game design, it is possible to dynamically create exciting situations by spawning autonomous objects on the fly in game fields. The origins of the way AI is used in modern game design can be seen in “PAC-MAN” (NAMCO, 1980).

2. The origin of video game AI: PAC-MAN

The full design documentation for “PAC-MAN” has been preserved perfectly, and all the documents were published in 2019 in the Journal of the Japanese Society for AI (Iwatani, Miyake, and Takahashi, 2019). In these documents, Iwatani explains how enemies move around the map so as to surround the player while leaving one escape route open. The behavior of enemies plays an important role in game design. In “PAC-MAN”, the behavior of enemy characters creates the dynamics of the game, by giving the player opportunities to escape from dangerous situations. Each enemy has specific features and characteristics according to its color.

The game design of “PAC-MAN”, and its use of character AI, is the origin of modern game design. Innumerable “PAC-MAN”-like game designs were published since, and the innovativeness of the game’s design is difficult to understand from a modern perspective.

Another important feature of “PAC-MAN” is its “attack waves”. The game system of “PAC-MAN” controls the number of enemies that appear on the map. Furthermore, it controls when they attack the player, and

when they run away from the player. When attacking, the enemies surround the player, but when running away, the four enemies each move separately towards the corners of the map. This gives the gameplay an artificial rhythm (Figure 1).

An AI that controls the dynamics of an entire game is called a Meta AI, or AI Director, and "PAC-MAN" was the first game to use a Meta AI. A Meta AI was again used in "LEFT 4 DEAD" (Valve Software, 2008) (Booth, 2009) in 2008 and "Warframe" (Digital Extremes, 2013) (Daniel, 2013) in 2013, and has become an important technique for in-game AI.

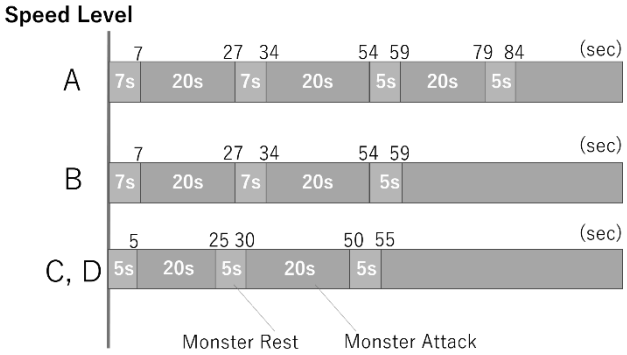


Figure 1. Timing table for enemy characters' tactics (Iwatani, Miyake and Takahashi, 2019)

Toru Iwatani, the game designer of "PAC-MAN", says in his interview with Youichiro Miyake (Iwatani, Miyake and Takahashi, 2019):

> "The player will not want to have to constantly keep escaping, so I used "attack waves", creating times when the enemies do not follow the player. I set intervals when the enemies attack the player, and intervals when they run away from the player. When the attacking interval finishes, the four enemies each run away towards their starting positions at the four corners of the map. From the player's perspective, I imagined it would feel like a lucky escape from being surrounded. I simulated all the characters' moves in my brain."

3. Meta AI in Xevious

"Xevious" (NAMCO, 1982) is also one of the first examples of Meta AI. "Xevious" is a 2D shooting game, and it uses a table to set the spawning order of flying enemies, going from weak to strong. When a player is defeated by an enemy, the table rolls back to weaker enemies. The system is therefore able to adjust the difficulty according to each player.

Masanobu Endo, the game designer of "Xevious", says (Endo 1987):

> "In Xevious, a simple but interesting AI was implemented. It judges the player's skill level, and makes enemies stronger according to it. A player with a high skill level encounters stronger enemies, and a player with a low skill level encounters weaker enemies. If the difficulty of a game is fixed, a common problem is that the game is too difficult for a beginner, but too easy for a skilled player. This AI solves that problem. A low-skilled player can play the game for longer, and a high-skilled player can enjoy exciting gameplay."

4. Utility system: Gambler Jiko Chushinha

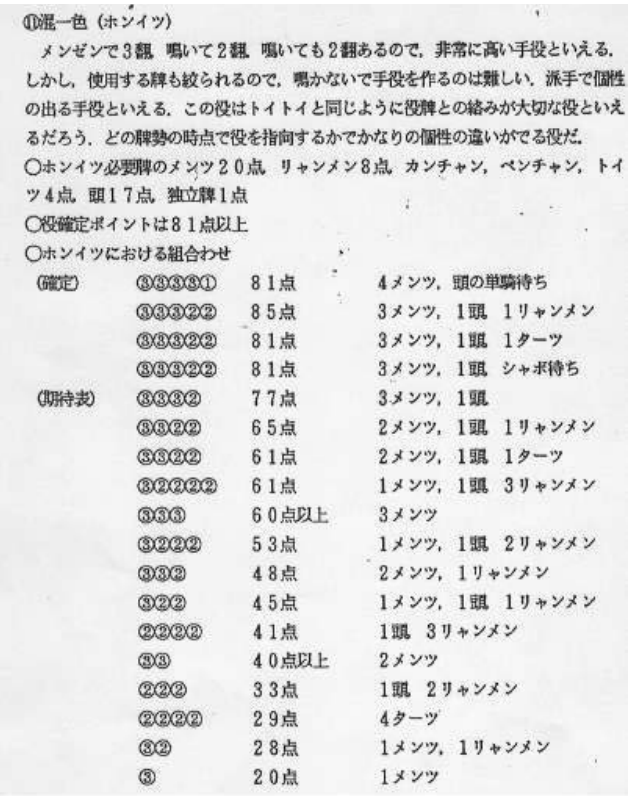
The use of utility-based decision-making systems is a popular method in not only '80s video games, but ones from all eras. It is a simple method in which one choice is selected from multiple choices by evaluating each of them. For example, an NPC decides one enemy to attack out of multiple enemies.

"Gambler Jiko Chushinha" is a mahjong game developed in 1987 by Yoichi Miyaji, the president of GAME ARTS at the time. The AI in the game decides which move to make, without cheating, using only information that is also available to human players.

In a mahjong game, there are many combinations of tiles. The utility system evaluates each combination as a numerical value, and the evaluation changes depending on the current game situation. In order to create the design document for "Gambler Jiko Chushinha" (Figure 2), Yoichi Miyaji played numerous games over a one-and-a-half-year period, continually adjusting the evaluations to ensure correct values for all possible combinations.

Yoichi Miyaji says in his interview with Youichiro Miyake (Miyaji et al., 2018):

> "I spent one and a half years play-testing the game. I repeated test plays thousands of times—tens of thousands of times—to get feedback for the evaluations."



①混一色（ホンイツ）

メンゼンで３翻，鳴いて２翻，鳴いても２翻あるので，非常に高い手役といえる．しかし，使用する牌も絞られるので，鳴かないで手役を作るのは難しい．派手で個性の出る手役といえる．この役はトイトイと同じように役牌との絡みが大切な役といえるだろう．どの牌勢の時点で役を指向するかでかなりの個性の違いがでる役だ．

○ホンイツ必要牌のメンツ２０点，リャンメン８点，カンチャン，ペンチャン，トイツ４点，頭１７点，独立牌１点

○役確定ポイントは８１点以上

○ホンイツにおける組合わせ

（確定）	③③③③①	８１点	４メンツ，頭の単騎待ち
	③③③②②	８５点	３メンツ，１頭，１リャンメン
	③③③②②	８１点	３メンツ，１頭，１ターツ
	③③③②②	８１点	３メンツ，１頭，シャボ待ち
（期待表）	③③③②	７７点	３メンツ，１頭
	③③②②	６５点	２メンツ，１頭，１リャンメン
	③③②②	６１点	２メンツ，１頭，１ターツ
	③②②②②	６１点	１メンツ，１頭，３リャンメン
	③③③	６０点以上	３メンツ
	③②②②	５３点	１メンツ，１頭，２リャンメン
	③③②	４８点	２メンツ，１リャンメン
	③②②	４５点	１メンツ，１頭，１リャンメン
	②②②②	４１点	１頭，３リャンメン
	③③	４０点以上	２メンツ
	②②②	３３点	１頭，２リャンメン
	②②②②	２９点	４ターツ
	③②	２８点	１メンツ，１リャンメン
	③	２０点	１メンツ

Figure 2. Utility table for Gambler Jiko Chushinha (Miyaji et al., 2018)

5. The first game engine in the world: NAMCO's Job-CON

In the first half of the 2010s, many game developers developed original game engines. They competed to make high-quality game engines in order to create AAA games. But the world's first game engine was created by NAMCO.

"Job-Controller" is the name of NAMCO's game engine, designed and developed by Shoichi Fukatani (1954-1985), who led technical development at NAMCO. It was a high-performance, object-oriented game engine, implemented in assembly language, and was used for many of NAMCO's games in the '80s. Using this game engine made it possible to create games just by implementing object behaviors, without having to rewrite the main game routine. All assets in a game are different types of objects, and a developer modifies objects by adding behaviors and attributes to them. NAMCO's "Job-Controller" is the oldest game engine, and is also the origin of the task systems now used in game engines around the world.

The basic principle of "Job-Controller" is that stack pointers are rewritten dynamically, creating a sequence in which object operations are executed (Figure 3). From a modern programming perspective, it is a little akin to hacking in its methodology, but at the time, it was an elegant way to create an object-oriented game engine.

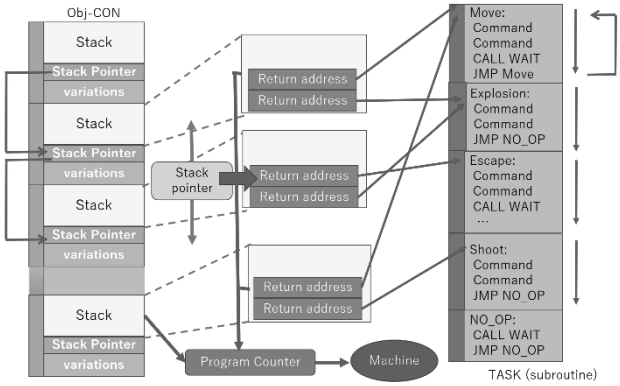


Figure 3. Job-CON in NAMCO (Kurosu and Miyake, 2009)

Kazuo Kurosu was a programmer at NAMCO. He joined NAMCO in 1979. He was part of the first generation of NAMCO's engineers. He made many famous games, such as "RALLY X", "BOSCONIAN", and "Libble Rabble" (NAMCO, 1980, 1981, 1983).

Kazuo Kurosu says in his interview with Youichiro Miyake (Kurosu and Miyake, 2009):

> "Shoichi Futakani, who is no longer with us, joined NAMCO before I did. He made a unified system that simplified the development process. This was at a time when the number of new employees had greatly increased, so he wanted to make things simpler for them and ensure coherency. The fundamental system was called "Job-CON", and it had another system called "Obj-CON" running on top of it—but they ran independently of each other, just like a psuedo multi-task environment! Shoichi Fukatani would get all the development staff together and give lectures to explain how it worked."

Many NAMCO games were built and executed using this same system. Game developers could concentrate on game development, without having to consider the fundamental system being used.

6. The history of video game AI technologies in the 1980s

These cases of video game AI in the '80s are not large-scale compared to the present era from 2015 onwards, but they demonstrate high-quality and complete forms of in-

game AI. They represent the birth of video game AI. Notably, the main concepts seen in AI systems in modern games, such as Character AI, Meta AI, Navigation AI, and Task system, also appeared in primitive forms in the in-game AI systems created in Japan in the '80s. (Table 1).

Now we are able to understand the details of these works, but at the time, the knowledge likely remained within the companies. In Japan, high-level game AI technologies were created in isolation, within a local community or company.

Table 1. '80s in-game AI technologies

Year	Game title	AI Technology
1980	PAC-MAN	Character AI / Meta AI
1980	Job-CON	Game Engine/ Task System
1982	Xevious	Meta-AI
1987	Gambler Jiko Chushinha	Utility system

A common comment from these developers is that did not intend to develop AI, and that they simply wanted to make good games. AI technologies were invented simply as a result of considering the user's emotions when playing.

Toru Iwatani says in his interview with Youichiro Miyake (Iwatani, Miyake and Takahashi, 2019):

> "I think any game AI technology was invented as a result of caring about the user's emotions, considering how to make a game that is exciting and fun, and trying to avoid the user disliking the game."

7. Yukito Morikawa's AI works in the 1990s

AI in the video game industry in the '80s was independent from the second AI boom, but in the '90s it was related to it. The second AI boom involved three main technologies: neural networks, genetic algorithms, and expert systems. There are several important instances in which these AI techniques were applied to game design in Japan.

Yukito Morikawa used neural networks and genetic algorithms in his games. The main character in "GANBARE MORIKAWA KUN 2 GO" (muumuu, 1997) has a neural network brain. It learns the correct behavioral responses to different environments through user instruction. Monsters in "Astronoka" (muumuu, 1999) evolve according to genetic algorithms. Enemy monsters evolve according to the environment, with parents selected via trap-based battles between players and monsters. Monsters that are successful in battle will become parents and pass on their genes. "KUMAUTA" (muumuu, SIE, 2002) is an unusual game in which a bear writes lyrics and makes a melody by itself. A user can only select three words to be included in the lyrics.

Yukito Morikawa says in an interview with Youichiro Miyake (Morikawa et al. 2019):

> "So, first of all, I read all the AI books I could get at Shinjuku's Kinokuniya book store. At the time there were only three shelves of them at the book store. … All the ideas for my games, including "Astronoka" and "KUMAUTA", came to me during the half-year I spent studying AI. So, the ideas all came at around the same time, and the games just happened to get made in the order that they did."

Yukito Morikawa was not a game designer, and he had a background in CG creation for the TV industry. His AI work was a new phenomenon; it was independent from and unrelated to the AI work seen in the Japanese game industry in the '80s.

8. Seaman's development in the 1990s

"Seaman" (SEGA, VIVARIUM, 1999) is a game in which the user raises a creature known as a Seaman, which learns to talk using natural language as it evolves. The user can enjoy natural conversation with the Seaman via voice input.

Some academic approaches to AI conversation exist: replaying sentences from stored data based on keywords spoken by the user, creating sentences generated from the predetermined patterns, or using neural networks. The approach used in Seaman, which was developed by Yutaka Saito, is developed from the predetermined pattern method.

All lines were recorded as sound clips using Saito's voice. The Seaman remember words that the user has said previously, and selects the next line according to certain rules. Seaman's recorded voice has different variations for the same lines; Saito recorded multiple versions for each line, which have different meanings according to the different pronunciations.

Yutaka Saito says in his interview with Youichiro Miyake (Saito et al., 2017):

> "If there's one thing that I want to say to everyone, it's that we really need to stop using approaches based on written language. We need to realize that the written grammar that we learn is useless when looking at natural conversation. The only way to approach natural conversation is to reconstruct unique grammar for spoken language. When recording all the voices for Seaman by myself, I realized that there are rules that exist within the melody of spoken language."

9. The connection between the 1980s, 1990s, and 2000s

In the '90s, new Japanese game developers studied AI techniques, and produced games based on new concepts. A new game genre, "AI games", was created, but it did not become mainstream. The objective at the time was simply to make good games (Table 2).

Table 2. '90s in-game AI technologies

Year	Title	AI Technology
1997	GANBARE MORIKAWA KUN 2 GO	Neural Network
1999	Astronoka	Genetic Algorithm
1999	Seaman	Natural Language Conversation
2002	Kumauta	Auto-generation

Many developers were busy pursuing 3D Graphics techniques, and successors of these '90s AI works did not appear. Before 2015, they were rarely looked back upon. However, with the third AI boom, a lot of attention was paid to machine learning and natural conversation, and these works were looked back on as part of the history of in-game AI.

Created in the United States, DOOM (id Software, 1993) had a big influence on the video game industry internationally. CG and AI are the two important factors in 3D game development, and both of them can be found in DOOM. Thus, the source code for DOOM and Quake (id Software, 1996) being published by the developer as open source had a huge impact. Furthermore, the Black Book written by Michael Abrash, the principle engineer of Quake, was also published (Abrash, 1997). It explains the techniques used by Quake for CG and AI in a 3D environment. The AI techniques used in DOOM and Quake are: navigation AI to find a path through the 3D environment, state machines for decision making, and perception of 3D spaces, such as the BSP (Binary Space Partitioning) method (Table 3). This was the starting point of western 3D game development seen in '90s and '00s, and in the '00s, the Japanese game industry was unable to lead 3D game development.

Academic researchers in 3D CG researched real-time rendering, and from 2000, academic AI researchers began to research AI in video games. As a result, the Japanese game industry fell behind the United States and Europe.

Table 3. '90s in-game AI technologies in the United States

Year	Title	AI Technology
1996	Quake	Perception AI /BSP
1996	Quake	Navigation AI
1996	Quake	State machines

It is not possible that modern in-game AI systems created by developers in the United States and Europe in the '00s and '10s are connected directly to Japanese in-game AI systems from the '80s, because there were no reference materials available in English for these Japanese examples. These Japanese in-game AI systems had little influence on video game AI around the world. But even now, these works are of historical importance, because they provide good examples of combinations of AI technology and game design.

In many cases, Western game development works on a technology-first basis. After a fundamental technical base has been built, game development begins. Fundamental in-game AI techniques were built as a base for game development in the United States and Europe over 20 years, from 1995 to 2015. In contrast, Japanese game development is design-first, both historically and today. Unique AI systems were created for specific game designs, but once development ended, these techniques were abandoned, because many Japanese developers thought that they were only of use for specific titles rather than being general techniques. The Japanese game industry produced some AI techniques, but was unable to build a fundamental AI technology base by combining these creations.

This paper is the first document in English to introduce the overall history of in-game AI in Japan. By considering past examples, the Western and Eastern history of in-game

AI can be unified. But even now, in both the West and the East, lots of old AI work in game development remains unknown. More oral histories and investigations into usage of AI in video games should be made, in order to reconstruct a chronological history of in-game AI by connecting together these different examples.

References

Abrash, Michael. 1997. *Michael Abrash's Graphics Programming Black Book*, Coriolis Group. Accessed September 15, 2019. https://github.com/jagregory/abrash-black-book.

Booth, Michael. 2009. *Replayable Cooperative Game Design:Left 4 Dead*. GDC. https://www.valvesoftware.com/ja/publications.

Brewer, Daniel. 2013. *AI Postmortems: Assassin's Creed III, XCOM: Enemy Unknown, and Warframe*. GDC. http://www.gdcvault.com/play/1018223/AI-Postmortems-Assassin-s-Creed.

Endo, Masanobu. 1987. *Xevious Seminar*. Itoi Shigesato's TV Show for Digital Playing. Fuji Television.

Iwatani, Toru. Miyake, Youichiro. Takahashi Mirei. 2019. "Toru Iwatani Interview: How was PAC-MAN born?." *Journal of the Japanese Society for Artificial Intelligence 34, no.* 1: 86-99. http://id.nii.ac.jp/1004/00009586/.

Kurosu, Kazuo. Miyake, Youichiro. 2008. "Kurosu's Programming Background: Interview with Kazuo Kurosu about Task Systems." *Report of Advanced Technology for Digital Contents, Digital Contents Association Japan 2009*: 301-330. http://www.dcaj.or.jp/project/report/pdf/2008/dc_08_03.pdf.

Miyaji, Yoichi, Youichiro Miyake, Manabe, Daisuke Koyama. 2018. "The first mahjong AI development", *denfaminicogamer*. https://news.denfaminicogamer.jp/interview/181024.

Morikawa, Yukito. Rie Narusawa, Yoshitaro Honjyo, Youichiro Miyake. 2019. "The development story of AI games", *denfaminicogamer*. https://news.denfaminicogamer.jp/interview/190424b.

Saito, Yutaka. Hirotaka Oosawa, Mirei Takahashi, Yoichiro Miyake. 2017. "Interview with Saito Yutaka: Will Seaman reveal the future of conversation agents?", *32*, no. 2: 172-179. http://id.nii.ac.jp/1004/00008564/.