Game Design with Automata Theory

Ahmed Shuaib

May 24, 2024

Introduction

Aliquam erat volutpat. Nunc eleifend leo vitae magna. In id erat non orci commodo lobortis. Proin neque massa, cursus ut, gravida ut, lobortis eget, lacus. Sed diam. Praesent fermentum tempor tellus. Nullam tempus. Mauris ac felis vel velit tristique imperdiet. Donec at pede. Etiam vel neque nec dui dignissim bibendum. Vivamus id enim. Phasellus neque orci, porta a, aliquet quis, semper a, massa. Phasellus purus. Pellentesque tristique imperdiet tortor. Nam euismod tellus id erat.

Literature Review

There are many ways of incorporating automata in game design that have been considered. Many design decisions take place during the development of a game, these decisions can vary based on the genre of the game.

When developing 2D games that fall in the genre of role-playing games (RPGs), adventure games, and puzzles. A procedural-level generation has been demonstrated using cellular automata. In a thesis from 2022, A genetic algorithm is used to evolve the cellular automata rules applied to generate game levels. It is important to note that a procedural content generator designed using this method is meant to be used during the game development process rather than at runtime. The resulting game levels can have a significant impact on a game's replayability, engagement, and scalability. [1]

Games designed solely using automata are few in number, but they do exist. Most games designed using automata are finite in nature, however, in a paper from 2016 students designed an infinite runner game using mealy machines. The game consisted of several states: running, jumping, flying, and game over. The paper concluded that games designed using automata are less prone to bugs and the development process is simplified by handling many implementation decisions in the design phase. [2]

An example of a game designed solely using automata is Conway's Game of Life. Developed by John Conway the Game of Life is a "zero-player" game utilizing cellular automata. The game is essentially a square grid containing cells that evolve based on three rules: birth, death, and survival. A dead cell will become alive in the next "time step" if three of its eight neighboring cells are alive. A cell can die in the next time step due to overcrowding, which occurs when it has four or more alive neighbors, or from exposure, which happens when it has one or fewer alive neighbors. A cell survives the current "time step" if the cell has either two or three alive neighbors. [3]

Incorporating automata into game design offers a versatile approach. Allowing design decisions tailored to each game's genre. Merging the best of not only the gameplay experience but also the design and development.

Game Design

Input Alphabet

$\sum_{Move} = \{$ U, D, L, R $\}$
$\sum_{Action} = \{$ P, J, O $\}$

$\sum_{Game}$ = { T$_n$, TA$_n$, A$_n$, B$_n$, NB$_n$, D$_n$, ND$_n$}

Description of input alphabet

$\sum_{Move}$: Defines the movement of the player. Up (U), Down (D), Left (L), and Right (R).

$\sum_{Action}$: Defines the actions the player can take. Push button (P), Jump (J), Open door (O).

$\sum_{Game}$: Defines some conditions for the game. Timeout when T$_n$ <= 0. Total attempts, TA$_n$, and state attempts , A$_n$ both of which are some integer. Status of the buttons (B$_n$, NB$_n$), and status (active/disabled) of state (D$_n$, ND$_n$) both of which are either 1 (True) or 0 (False).

States of game

Time State

T$_n^t$: Provides player with time and tracks total attempts.
Effect: T$_n$ = t, TA$_n$++

Platform State

P$_n^t$: Platforms that reduce time.
Effect: T$_n$ -= t

Stair State

S$_n^t$: Stairs that reduce time.
Effect: T$_n$ -= t

Button State

B$_n^t$: Buttons that reduce time and sets status of buttons.
Effect: T$_n$ -= t, B$_n$ = 1, NB$_n$ = 0

Button State (N)

NB$_n^t$: Buttons that reduce time and sets status of buttons.
Effect: T$_n$ -= t, B$_n$ = 0, NB$_n$ = 1

Jump State

J$_n^t$: Jump pads that reduce time.
Effect: T$_n$ -= t

Hole state

H$_n$: Innocent Holes. Effect: No effects.

Ghost Platform State

GP$_n^t$: Platforms that reduce time and depend on status of buttons.
Effect: T$_n$ -= t

Ghost Wall State

GW_n^t: Walls that reduce time and depend on status of buttons.
Effect: T_n -= t

Platform Wall State

$\mathrm{PW}_n^{h^d}$: Walls that reduce time based on damage and when disabled act like platforms.
Effect: T_n -= $h - (d \cdot (++\mathrm{A}_n))$
$\mathrm{D}_n = (h - (d \cdot \mathrm{A}_n)) <= 0$
$\mathrm{ND}_n = (h - (d \cdot \mathrm{A}_n)) > 0$

Cube state

$\mathrm{C}_n^{h^d}$: Cubes that reduce time based on damage and when disabled allow player to move forward.
Effect: T_n -= $h - (d \cdot (++\mathrm{A}_n))$
$\mathrm{D}_n = (h - (d \cdot \mathrm{A}_n)) <= 0$
$\mathrm{ND}_n = (h - (d \cdot \mathrm{A}_n)) > 0$

Game Description

Nullam eu ante vel est convallis dignissim. Fusce suscipit, wisi nec facilisis facilisis, est dui fermentum leo, quis tempor ligula erat quis odio. Nunc porta vulputate tellus. Nunc rutrum turpis sed pede. Sed bibendum. Aliquam posuere. Nunc aliquet, augue nec adipiscing interdum, lacus tellus malesuada massa, quis varius mi purus non odio. Pellentesque condimentum, magna ut suscipit hendrerit, ipsum augue ornare nulla, non luctus diam neque sit amet urna. Curabitur vulputate vestibulum lorem. Fusce sagittis, libero non molestie mollis, magna orci ultrices dolor, at vulputate neque nulla lacinia eros. Sed id ligula quis est convallis tempor. Curabitur lacinia pulvinar nibh. Nam a sapien.

References

[1] Adel Sabanovic and Amir Khodabakhshi, Evolved cellular automata for 2D video game level generation, 2022.

[2] Abid Jamil, Engr. AsadUllah and Mohsin Rehman, An Infinite Runner Game Design using Automata Theory, 2016.

[3] Kuldeep Vayadande, Ritesh Pokarne, Tanmay Patil, Mahalakshmi Phaldesai, Prachi Kumar, and Tanushri Bhuruk, Simulation of Conway's Game of Life using cellular automata, 2022.