Mechanic Maker 2.0

Mentor: Megan Sumner
Students: Corey Polo, Max Zang, Caly Zheng

Fall 2023

1 Research Context and Problem Statement

The majority of game development requires programming knowledge. However, many people have expressed interest despite having zero to little programming experience. To make game development more accessible to users without previous programming knowledge, Mechanic Maker was developed. Mechanic Maker 1.0 allows an individual to collaboratively create game rules with an AI agent co-creator. The AI agent provides suggestions for object placement when designing a level. It currently uses a rule based algorithm, which is adapted from the AI agents used in Morai Maker [1].

In 2019, a user study was conducted by Guzdial et al. [1] which tested the implementation of three different AI agents utilizing different AI models: Markov Chain, Bayes Net, and Long Short Term Memory (LSTM). In Morai Maker, a turn based interaction between the user and the AI is used. The user study found that, on average, participants deleted 60% of the agent's changes, which indicated that many participants remained unsatisfied with all agents. Additionally, they found that the usage of Morai Maker was greatly varied such that none of the AI agents were able to handle the variety of human participants [1].

In September of 2023, Megan Sumner conducted a user study based on the work of Vardan and Guzdial to determine Mechanic Maker's relative ability to facilitate game creation for non-programmers and programmers alike [2]. This study has been submitted but not yet published. However, our research primarily builds off this user study. In Mechanic Maker, users can place sprites and make inputs on a per-frame basis, iterating frame by frame. This is similar to the Morai Maker software which Mechanic Maker is based on. Unlike Morai Maker, however, the AI agent in Mechanic Maker provides suggestions instead of executing them.These suggestions are indicated by sprites with distinct colouring and opacity from those of which the user has placed.

From this study, the results showed that the majority of users enjoyed using this tool, and feedback from this study was mostly positive that Mechanic Maker had potential for game development. However, the rules learnt from the user's playthrough of Sokoban and Flappy Bird were compared against the rules in the example, and showed to have a low accuracy of 44.6

While effective, the co-creator agent in previous studies is limited as it only offers the user one potential path of creation. Additionally, Sumner identified improvements that could be made to the UI to make it more user friendly. We aim to improve Mechanic Maker through the backend AI agent and the UI to allow for more ease of use and creativity from the user.

2 Proposed Solution

There are multiple different approaches that have been suggested to improve the effectiveness of Mechanic Maker, which can be broken down into improving the AI agent and the user interface. Some suggestions to improve the AI agent are to implement different ML models such as variational autoencoders and hierarchical mixture models. We will be primarily focused on ameliorating the current Rule-Based Model. Our main objective is to incorporate previously learnt rule-sets from the 2023 user study into the rule based model. In doing so, the model will be able to learn faster and from a wider variety of rules, not just from those it learns from the user in the current session. It is our goal that this will provide more desirable paths of creation for the user, thus making game development more accessible for those without prior programming experience.

2.1 Improved Rule-Learning

In order to improve the rule-based model, we will make changes to the model so that it can accommodate previously learnt rules. In order to do so, we will need to create a database of previously learnt rules which the backend can read and then learn from. The data gathered from the user study will be taken, the rules from each session will then be generated and added to the database. This collection of rules will not only be used in improving the rule-learning model, but will be used to train other machine learning models, such as variational autoencoders.

Given the two-dimensional format of the tool, and given commonly shared game-control fundamentals of two-dimensional games (e.g. arrow keys for movement, spacebar for jump, etc.), we aim to accelerate initial rule learning by incorporating the rules database created during the pre-processing step. Instead of completely generating rules from scratch solely based on user actions at the outset of a new session, the database will be queried in search of previously learned rules which most closely match currently generated rules and recommend them to the user. This will ideally lead to quicker mechanic implementation, as well as providing more accurate suggestions.

To improve the usability of the AI agent, we will adjust the current rule-based model to display two potential moves, instead of the current singular one. This will give the user more options for a next move, and also increase the number of times the AI agent is being used.

By integrating learnt rulesets from previously collected data and providing multiple options to the user, we hope that the AI learning model will be more effective in providing more desirable paths of creation to the user, and thus aid in facilitating creativity. In turn, this will make game creation more accessible for users without prior programming experience.

2.2 User Experience

In order to best leverage the new back-end architecture toward our goal of creating a tool which aids in game development, we will additionally overhaul certain aspects of the user experience. Based on feedback from the user study, there are two main areas in which features need to be changed or added: information design and user interface. We will adjust the way information is presented to the user in two main areas: the state-space of the current session, and new paths provided by the AI co-creator.

We will reduce the amount of available visible frames from three to one, thus requiring the user to manually switch between frames. While the same amount of state-space information will be available to the user, the amount available at any given time will be reduced. This will reduce complexity both computationally for the user interface as well as experientially for the user. We will also adjust the visuals such that the previous frame information appears alongside the AI co-creator's suggestions.

Given the back-end changes, our hope is to create a learning infrastructure which is able to provide the user more than one suggestion at any given time. Towards that end, we will more clearly define the co-creator's suggestions and implement the ability to visually present those suggestions.

3 Implementation Plan

Our improved rule-based model and user experience will be implemented by working in three main areas of the project: the front-end user interface, the rules-learning back-end, and the code base which connects the two.

3.1 Improved Rule-Learning

In order to create a database of previously learned rules, we will reproduce all of the rule output files created during the initial Mechanic-Maker user-study sessions. This will be done by running the CSV files representing frame data generated during the user study through the currently implemented engine learning model. Rulesets for each session will then be outputted to a folder, which can then be queried to improve and accelerate rule learning during the initial stages of a new session.

Currently, the learning model goes through three distinct steps when creating engines: adding rules, changing rules, and deleting rules. It then searches the space of neighbouring engines to find the neighbouring state with the lowest amount of error. Since the preprocessed ruleset database contains only those engines which have already met the threshold for error, and since the rulesets comprising the database are the result of user's interations of the Mechanic-Maker tool, the initial stages of game creation will be greatly accelerated for the user. To implement this, we will write a subroutine which queries the database after each user-created frame in search of the ruleset most closely resembling the user's. Afterwards, the top engine is returned, and is then displayed to the user through the front-end in Unity. In order to display more than one option to the user, we will adjust the rule-learning engine to return the top two engines, instead of the current singular engine.

3.2 User Experience

The front-end of Mechanic Maker is implemented in Unity. As such, all user interface changes will be implemented via the Unity development platform. Most of the changes will be done programmatically through the C-Sharp scripts linked to the various UI elements including locking the camera and reducing the amount of visible frames. The information design tasks, such as more clearly depicting the AI co-creator agent's suggestions, will be handled within the Unity interface itself.

The front-end in Unity will also be adjusted to display two options to the user instead of one. This will be implemented by reapplying the existing solution of displaying the AI agent's suggestion to the second engine chosen.

Another major task is to implement a feature allowing users to delete a frame in the middle of those generated during the current session. A deletion will then propagate through those frames ahead of the point of deletion and update according to the updated ruleset. This will be implemented by processing the updated state-space through the engine learning algorithm and then updating the visual representation of the state-space in Unity.

Our main goal is to finish implementation of these changes before the second user study, which will be conducted in January of 2023. This will allow us to verify the effectiveness of the implementation and the improvement in the user experience.

4 Evaluation

In order to assess the success of the changes to the Mechanic Maker tool, we will conduct a second user study. The data collected in this iteration of the user study will be compared against the data collected during the first user study. In the first study we collected two types of information. First, from the survey we collected self-reported quantitative, qualitative, and demographic information. Second, we logged all major Mechanic Maker events, all final games, and all of the frames produced by users.

In the one portion of the original study, participants were shown reference videos for versions of the games Sokoban and Flappy Bird created in Mechanic Maker, and then tasked with recreating the games. Using the rules created in the reference version of the games as a baseline, we were able to assess the accuracy of the rules created by participants. This measure from the original study will serve as a baseline of comparison for the second iteration of the study. Additionally, the survey conducted during the first study will serve as a comparative baseline to assess the user experience of using the tool. It is our aim that implementing these changes will result in a marked benefit to users of the tool.

In previous user studies, participants were asked questions about the ease of use of Morai Maker and Mechanic Maker. The same will be conducted in the second user study on Mechanic Maker.

References

[1] Matthew Guzdial, Nicholas Liao, Jonathan Chen, Shao-Yu Chen, Shukan Shah, Vishwa Shah, Joshua Reno, Gillian Smith, and Mark O. Riedl. Friend, collaborator, student, manager: How design of an ai-driven game level editor affects creators. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, New York, NY, USA, 2019. Association for Computing Machinery.

[2] Vardan Saini and Matthew Guzdial. A demonstration of mechanic maker: An ai for mechanics co-creation. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 16(1):325–327, Oct. 2020.