Universidad de Costa Rica
CI-2807, Teoria de Juegos, I ciclo 2019

Esc. de CC. de la Computación e Informática
Dr. Markus Eger

Project Requirements

Overall philosophy and goals

The general idea for the project for CI-2807, Teoria de Juegos is that groups of four students collaborate to design and develop a video game in Unity. The students are free to choose any genre of game or other interactive experiences for this project, with the only restrictions being that the game supports single player play, consists of three or more levels/stages/zones, contains some AI controlled character or system and uses Markov Chains (or a more advanced content generation method) to generate content or control the AI behavior. The project consists of three main parts:

- A design document describing the mechanics of the proposed game (8% of the grade)
- A prototype, which constitutes a first playable version (20%)
- The final product (25%)

Additionally, each group is expected to create a simple website (e.g. by using github pages) to document their progress and report their progress on the project in the form of a short presentation every Tuesday in class. These presentations are graded with 0.5% of the grade each, for a total of 7%. During the first of these presentations, on March 19, each group is given 10 minutes to introduce its members and provide three to five short (one to three sentences) pitches for possible games. Subsequent presentations will be limited to a maximum of 5 minutes per group, and not all group members are required to speak. Additionally, on June 11, each group will be given extra time to talk about their prototype, and will play another group's prototype and give constructive feedback.

Note that, while the game design document is the first deliverable, it would not be unexpected for game play to work out differently than originally planned. Any such changes should be recorded in the design document, and discussed with the instructor when they become necessary.

Finally, the main focus of this class are the programming aspects of game development, and therefore the students are not expected to produce their own art assets. There will be no point deduction if a group uses e.g. a simple box as their protagonist instead of the mighty Elven sorceress their design document describes. However, students are encouraged to peruse the Unity asset store for free art assets that fit the game they are developing, or use art assets from other repositories, such as opengameart.org. It is, however, *required* that students name the sources of their art assets and do not use copyrighted material without permission. All assets that are used must therefore be listed in a file `asset-sources.txt` in the project.

Markov Chains

Each game is required to use Markov Chains in a way to generate content or control AI behavior. Markov Chains consist of states and transition probabilities between these states. For example, a Markov chain could consist of three states, A, B, and C, with transition probabilities according to the following table:

	A	B	C
A	0.5	0.25	0.25
B	0.1	0.7	0.2
C	0.3	0.3	0.4

This can be read as: If the current state is A, the probability that the next state is A is 0.5, the probability the next state is B is 0.25, and the probability that the next state is C is also 0.25. The Markov Chain can be "run" by starting in one state (either a predefined one, or chosen at random), and picking the next state according to the given probabilities, and repeating this process as long as desired.

Markov Chains have many applications in video games, such as content generation. Consider, for example, a Markov Chain in which every state represents a letter of the alphabet. By running this chain for 5-8 steps, a word can be generated. The nature of the resulting word depends on the transition probabilities of the Markov Chain. By using different transition probabilities, differently sounding words can be generated. To determine transition probabilities, one can use existing text, and use the relative number of subsequent appearances of letters as transition probabilities. For example, if the input data is "eat", "ate", and "eta", the letter e is followed by the letter a once, and by the letter t once, so the transition probabilities from the state corresponding to e are 0.5 to the state corresponding to a and 0.5 to the state corresponding to t. By using a larger corpus of input words, and by only using words that have particular properties, the Markov Chain can generate new words that resemble the input words. For example, by using Costa Rican town names as the input, the new words that the Markov Chain generates will resemble Costa Rican town names, without necessarily being real places. This online name generator can be used to explore how Markov Chains generate words: https://www.samcodes.co.uk/project/markov-namegen/

However, the states in Markov Chains can represent any concept, such as:

- Enemies in a game, and running the chain determines which enemies attack the player
- Tiles in a level, and running the chain generates a new level
- Rooms in a castle, and running the chain generates a new castle
- Words from a book, and running the chain generates new (nonsense) text

Deliverable 1: Design Document   Deadline: March 31, 2019

Objective: Design the game play experience for a game, and describe the necessary parts in a structured way.

Goals:

1. Design a game that fulfils the requirements set forth in the course.
2. Describe game play in a structured way.

Evaluation: Since the class provides great freedom to the groups as to which kind of game (or interactive experience) they want to develop, the proposed project is not evaluated in terms of how it appeals to the instructor, but rather of whether it fulfills the technical requirements of the class. The main focus of this assignment, however, is the communication of game play concepts. This portion will be graded on whether all content that is required is present in the design document and how clearly the ideas are presented.

1. Gameplay (2%)

- Whether the game provides a single player experience
- Whether there are three or more levels/stages/zones
- Is there a part/character that is controlled by the computer in an intelligent way
- Is the proposed implementation language/environment reasonable

2. Description and Structure (6%)

- Does the Design Document contain all required contents:
    - A team name
    - A game name
    - A short, one paragraph “pitch” for the game
    - A detailed description of the game components (objects, attributes, relationships)
    - The game mechanics, including how player input is translated into changes of the game state, and the win/loss conditions (if applicable)
    - A short description of each team member: name, skills, and responsibilities
    - A rough division of labor between these team members, as well as any technology to be used. If you don’t use Unity, please note why, and which of your team members bring the required skills.
- Are gameplay concepts described in a concise and consistent manner
- Does the design document convey how the game works, and which decisions the player has to make in order to achieve the goals presented by the game

Deliverable 2: Prototype — Deadline: June 9, 2019

Objective: Implement a playable version of the game presented in the design document. This version is not necessarily free of bugs and fully balanced, but it should be possible to experience all aspects of the game.

Goals:

1. Get familiar with Unity and set up the project for the game
2. Design the code structure
3. Implement all game play mechanics

Evaluation: The main goal of this deliverable is to produce a playable version of the final product, which will then be iterated upon to turn it into a final product. Because a maintainable project structure will reduce the likelihood of game breaking bugs in the final product, part of the evaluation of this deliverable will be based on how maintainable the codebase is. The main focus of the evaluation, is the implementation of the game mechanics.

1. **Project structure (5%)**

- Is gameplay divided properly into scenes (if applicable)
- Are assets in the project grouped properly
- Is the code structured in a maintainable way, with classes representing single concept, and proper reuse of functionality

2. **Gameplay Implementation (15%)**

- Are all gameplay elements described in the design document present in some form
- Is the game playable in a way that conveys the mechanics to the player

Deliverable 3: Finished Game — Deadline: July 3, 2019

Objective: Submit a fully polished version of the game, in which the user/player is properly introduced to the game mechanics, and the game is playable and behaves as intended.

Goals:

1. Provide a proper introduction and conclusion to the game.
2. Balance the game
3. Fix bugs
4. Polish the game

Evaluation: The deliverable for this assignment should consist of a complete game, in the sense that a player can start the game, is gently placed into the game world, can play the game in a way where their decisions/reactions matter, and is communicated the outcome in an appropriate way. For example, this guidance could consist of a menu system, and scoring screen. It is also expected that this version is free of bugs. Additionally, the game play experience is evaluated again to give students the opportunity to improve upon any shortcomings in the prototype.

1. **Player Guidance (7%)**

- How is the game presented to the player
- Is the player introduced to the game gracefully
- Is the outcome of the game/level/episode presented to the player appropriately
- Is the user experience appropriate for the game

2. **Software quality (8%)**

- Are there any bugs (game breaking, visual, control problems, unintentional AI behavior)
- Do player decisions contribute to the outcome of the game in a meaningful and discernible way
- Is the game balanced as intended

3. **Gameplay experience (10%)**

- Are all gameplay elements described in the design document present in the game
- Do the mechanics behave as expected
- Are there any missing or superfluous mechanics
- Is it conceivable that the game is enjoyable for its intended audience
- Are all levels/stages/zones present