Venus

COM 490/L Senior Design Project Document

Gaming Community
Erick Bravo: Group Leader, Documentation/Research
Juan Vazquez: Documentation/Research
Angelina Nantanapramoth: Documentation/Research
Jazlyn Fernandez: Documentation/Research
Daniel Esquivel: Documentation/Research
Kathareeya Atthajaroon: Documentation/Research

Team Email:
erick.ramirez.785@my.csun.edu
juan.vazquez.399@my.csun.edu
angelina.nantanapramoth.486@my.csun.edu
jazlyn.fernandez.210@my.csun.edu
daniel.esquivel.326@my.csun.edu
kathareeya.atthajaroon.496@my.CSUN.edu

Team Website:
CSUN-LLIU/sd.github.io: CSUN Computer Science Senior Design
Dr. Li Liu
Revised: 12/5 Ver 1

EXECUTIVE SUMMARY

Development Standards & Practices Used

Development standards and practices that we used are Agile/Scrum processes. The design pattern that we followed is the MVC design pattern. We also used Unified Modeling Language.

Summary of Requirements

- Allow the user to play the game interactively
- Give hints organically via machine learning
- The game itself will not be affected by any machine learning
- User can choose to use hints or continue playing the game, machine learning will learn in background

Applicable Courses from CSUN Computer Science Curriculum

COMP 380/L (Introduction to Senior design)

COMP 542(Machine Learning)

COMP 482(Algorithm Design)

COMP 469(AI)

New Skills/Knowledge acquired that was not taught in courses

Research into machine learning algorithms, libraries, Patent research into existing ideas
Parsing through existing GitHub documentation

Table of Contents

1 Introduction **4**

1.1 Acknowledgement 4

1.2 Problem and Project Statement 4

1.3 Operational Environment 5

1.4 Requirements 5

1.5 Design Assumptions and Limitations 5

2 Architectural Diagram **6**

3 Modules **7**

3.1 Module -1: User Interface (builds into ONE executable/process) 7

3.1.1 Component 1: Button *7*

3.1.2 Component 2: Hints *7*

3.1.3 Component 3: Privacy Concern **7**

3.2 Module -2: Venus 7

3.2.1 Component 1: Reading Storage *7*

3.2.2 Component 2: Mouse Data *7*

3.2.3 Component 3: AI **7**

3.3 Module -3: Game: The House 7

3.3.1 Component 1: Game files *7*

3.3.2 Component 2: Passing Data *8*

3.3.3 Component 3: Data Separation **8**

4 Data Decomposition **9**

5 Detailed Design **11**

6 Design Rationale **12**

6.1 Design Issues 12

6.1.1 Issue#1: Arranging UML diagram columns 12

6.1.2 Description *12*

6.1.3 Factors affecting Issue *12*

6.1.4 Alternatives and their pros and cons *12*

6.1.5 Resolution of Issue *12*

6.2 Issue#2: Providing Choices to Users *13*

6.2.1 Description *13*

6.2.2 Factors affecting Issue *13*

6.2.3 Alternatives and their pros and cons *13*

6.2.4 Resolution of Issue *13*

6.3 Issue#3: Deciding on Mouse Acceleration *13*

6.3.1 Description *13*

6.3.2 Factors affecting Issue *13*

6.3.3 Alternatives and their pros and cons *13*

6.3.4 Resolution of Issue *14*

7 APPENDIX **15**

1 Introduction

1.1 Acknowledgement

We would like to acknowledge Artur Kot for providing the source code of the browser game, The House, that we are utilizing for this project. We would also like to thank Sony Entertainment for helping us to realize this idea is possible.

1.2 Problem and Project Statement

Games can sometimes be too difficult and can become confusing which reduces the user's experience and immersive capabilities within a game. In many games, the user is started off with a complicated tutorial that may not benefit the user's ability to play the game. Complicated tutorials or instructions can deter users from playing a game because they assume it's too difficult.

Our proposed solution for this problem is to create an AI helper called Venus that can learn the player's play-style and situations where the player is having difficulty. Venus can offer tips to assist the user which will positively increase the user's experience while playing a game. Venus will learn when to offer hints based on the user's clicks and cursor movement. If there is an excessive amount of clicks in one area, Venus will predict the user is having difficulty and provide a hint. If there is excessive mouse acceleration based on the user's confusion, the telemetry provided by this capture would provide Venus the go ahead to offer hints to assist the user. Both of these methods will provide Venus the material needed to determine when to prompt hints.

Venus is an AI helper that can offer assistance by giving hints when the user is having difficulty. Venus will use the methods previously mentioned to determine if the user is struggling or is confused with the tasks within a game. Another feature of Venus is the user will have the option to activate or deactivate hints. This feature is given because perhaps the user doesn't want hints and will instead activate hints only when they truly need it. Venus's purpose is to not play a game for the user, but to provide clues on how to complete difficult tasks.

What is driving this project is we hope to build an AI helper called Venus that can assist users in their gameplay when they face challenges that require assistance to complete it. This is important because this will make playing games more fun and more achievable when it's at a higher difficulty. Also, the program will help players in multiple scenarios and will offer educated tips on how to handle situations.

We hope Venus will enhance the user's experience making it more fun and less stressful to play. Games that are too difficult to do on your own won't be anymore because Venus will

give assistance on how to handle different situations in the gameplay. The output will be less experienced gamers will enhance their skills with the help of Venus and they will enjoy games even more with Venus tagging along with the player.

1.3 Operational Environment

This program will be dependent on a user's hardware where a minimum specific requirement will be required. This will of course go on top of whatever the selected games minimum requirements will also need. The environment that is suitable for our end product would be indoors and should not be exposed to liquids as is standard for any computing product.

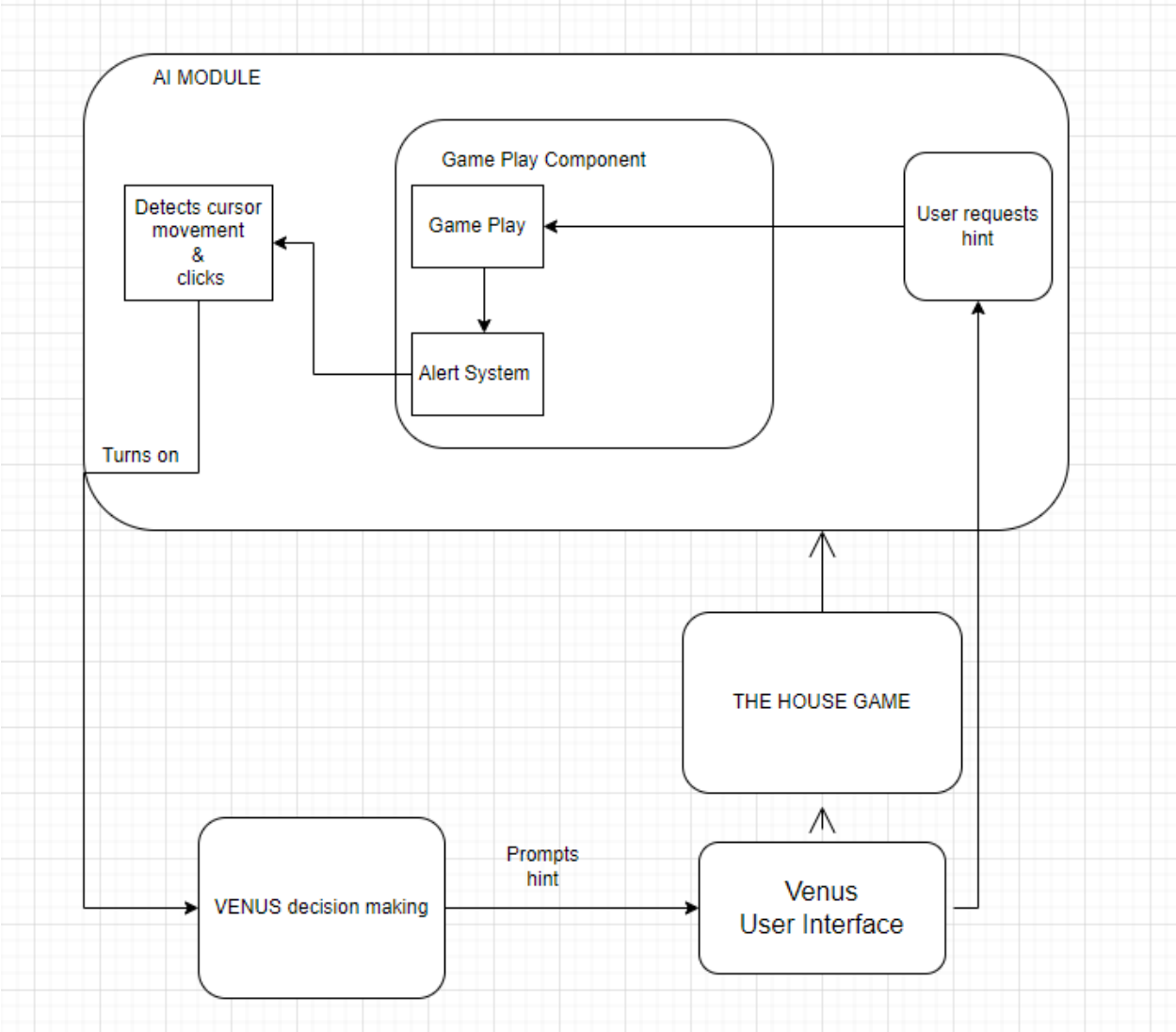
1.4 Requirements

- Functional:
    - Obtain user's mouse data in order for Venus to distinguish whether the user is having difficulties.
    - Users have the option to Disable/Enable Venus.
- Non-functional:
    - Venus should be helpful when prompting hints, not giving vague and confusing ones.
    - Venus should produce a hint quickly after it is prompted to.
    - The AI is not intrusive or distracting in the gameplay.

1.5 Design Assumptions and Limitations

- Assumptions:
    - Our product would be distinct from regular NPCs.
    - Interactions are dynamic rather than planned/programmed.
    - Imperfect information game play so that players can improve and adapt along the way.
    - One-person player game
    - Text-based
- Limitations:
    - Product will need to collect data of user game play before to be truly helpful
    - Find a way to modulate the level guidance to lower the risk of players cheating through the game.
    - The software will work on single player browser based games.
    - Need user’s permission to operate as to prevent invasion of privacy.

2 Architectural Diagram



3 Modules

3.1 Module -1: User Interface (builds into ONE executable/process)

3.1.1 Component 1: Button

The user will be able to interact with the button to enable Venus. If the user does not want hints that Venus provides, they have an option to also disable it.

3.1.2 Component 2: Hints

When the user enables Venus, a text box will appear that gives the based on where the user is struggling.

3.1.3 Component 3: Privacy Concern

Before playing the game, the user will be notified about what data is being collected with the addition of Venus. In order to play the game, the user has to agree, by clicking the, "I agree" button.

3.2 Module -2: Venus

3.2.1 Component 1: Reading Storage

Venus will read The Houses data storage via its jstorage.js file. The data will not be modified, instead it utilizes a pointer to read the data. The pointer data will have its own storage location.

3.2.2 Component 2: Mouse Data

In order for Venus to know when the user is struggling, it will track the mouse's data: clicks and acceleration. The Houses JS click functions will feed into Venus' mouse threshold analyzer. With sufficient user clicks, the mL algorithm component will activate.

3.2.3 Component 3: AI

Venus' mL/AI component will provide the best guesstimate on the user's current state.Using the present state of the game, it will be learning with the user, learning it as it is new to the game.

3.3 Module -3: Game: The House

3.3.1 Component 1: Game files

The game itself is based on Javascript with CSS implementations for the website. The most touched files include: game.js, data.js and jstorage.js.

3.3.2 Component 2: Passing Data

The Houses' user-interface, such as interactive objects, will detect the user's mouse events. The mouse events are derived from utilizing the JS click function, which will pass through to Venus.

3.3.3 Component 3: Data Separation

Venus won't modify the game's data or state. Only the user's gameplay and use of Venus' hints will influence the game data.

4 Data Decomposition

Persistent data:

Game world and save state

Functions from game, Venus, and ml5 API

Venus array during runtime

Collison_nodes Array:

collision_nodes:

//cupboard

'0-3', '0-4', '0-5', '0-6', '0-7', '0-8', '0-9', '0-10', '0-11', '0-12', '0-13', '0-14', '0-15', '0-16',

'1-3', '1-4', '1-5', '1-6', '1-7', '1-8', '1-9', '1-10', '1-11', '1-12', '1-13', '1-14', '1-15', '1-16',

//table + chair

'6-7', '6-8', '6-9', '6-10',

'7-5', '7-6', '7-7', '7-8', '7-9', '7-10',

'8-7', '8-8', '8-9', '8-10',

//fridge

'7-0', '7-1', '7-2',

'8-0', '8-1', '8-2'

The following game function uses the persistent gamedata array:

Game.js corridor function:

```
//reveal the secret door (using phone)
//first check if it has been already done
if ( $.inArray("scene_corridor_phone", played) !== -1 ) {
$('#phone, #phone_mask, #phone_shadow').remove();
//overwrite collision nodes
room.settings.collision_nodes = [];
```

```
$('#8-0, #8-1, #9-0, #9-1').removeClass('collision');

$('#hidden_door').css('height', '246px');

$('<div id="door_hidden_corridor" data-tooltip="Go to the corridor" />')

.appendTo('#corridor')

.tooltip('left');

.VenusGameMLFunction(#corridor);
```

**After .tooltip('left'); this would be augmented to include our hook for Venus, .VenusGameMLFunction(forExample) and allow for the game to work as intended with our modification.*

Entity relationship:

Relational database Game Data - Ml5 Venus Data

Game Data - has array collision_nodes

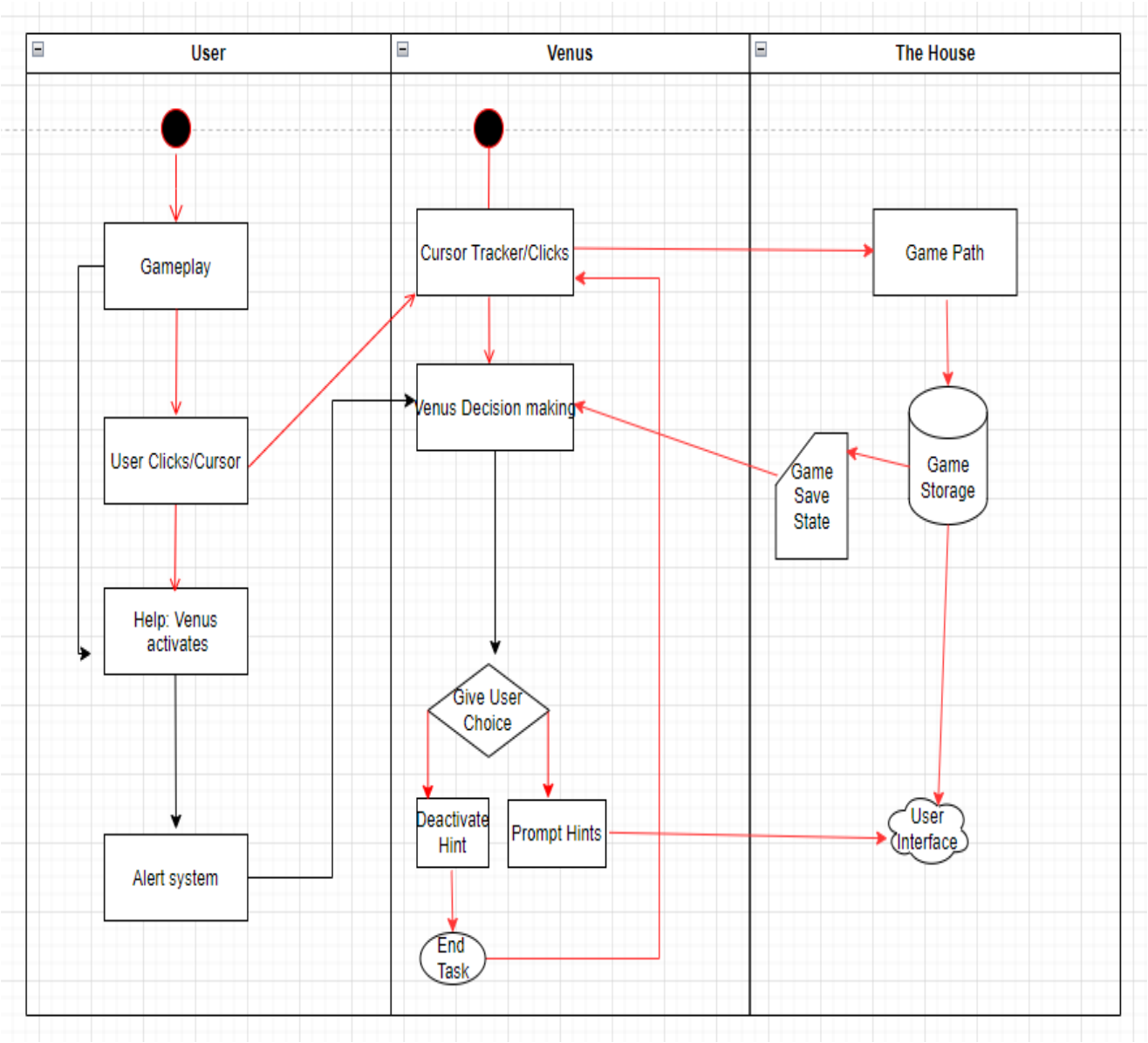
Won't be modified

Venus Data - will use array example_nodes

Ml5 algo has ml5.image & ml5.text categories that will modify Venus data

The index will be managed by Venus to read from game data that was copied over

5 Detailed Design



6 Design Rationale

6.1 Design Issues

6.1.1 Issue#1: Arranging UML diagram columns

6.1.2 Description

We have decided to use an activity diagram for our designs. The issue was raised when arranging columns and attempting to connect the user, Venus, and the game together.

6.1.3 Factors affecting Issue

There were multiple ways to arrange the columns. It could have been arranged like: The House → Venus → User, because the User GUI is how the user will activate Venus.

6.1.4 Alternatives and their pros and cons

Pro of User → Venus → The House: Shows that Venus won't override user decision nor the game play experience. Also shows that the User is the main actor. Venus will activate only when it detects that help is needed.

Con of The House → Venus → User: May bring confusion of Venus playing the game or giving "cheats" to users. May seem confusing by making it look like The House and Venus interact with the User when it's the other way around.

6.1.5 Resolution of Issue

We figured it would be ideal to arrange the columns from User → Venus → The House game. Our main goal is Venus, putting AI adjacent to the user and the game itself enables us to connect the interactions easier.

6.2 Issue#2: Providing Choices to Users

6.2.1 Description

In creating the AI helper, we focused on the user getting help but not having it be forced upon the user.

6.2.2 Factors affecting Issue

When the user chooses "no", there is junk data now. Reusing that data could corrupt the future data for the Venus game state.

6.2.3 Alternatives and their pros and cons

Pros of allowing choice: Users can play the game with Venus watching at the ready in case a "guardian angel" is needed. Data is fed into Venus to be used to improve predictive path guesstimates.

Cons of disallowing choice: Users feel inclined to use Venus and then feel as the game is being played for them. Users may leave the game and not come back or ask for an AI free version.

6.2.4 Resolution of Issue

We allow user's choice to exist and will keep the created data as a reference to improve the machine learning algorithm for the game. Keep the data and use it to improve the algorithm or junk it and have the algorithm be light and trim.

6.3 Issue#3: Deciding on Mouse Acceleration

6.3.1 Description

We were introduced to mouse acceleration functionality of javascript and how it could help advance the machine learning concept in our Venus program.

6.3.2 Factors affecting the Issue

In using mouse acceleration, we have to find how we delineate spikes in graphs that correspond to user game play. These spikes can be a little dependent on the type of mouse or trackpad used as well.

6.3.3 Alternatives and their pros and cons:

Pros to Mouse acceleration:
Having a run through of the game to set a baseline standard could give us the

optimal guide in how to use the acceleration for click-based events in Venus & the House game.

Cons to Mouse acceleration:
Using mouse acceleration could actually be overly sensitive and not trigger in our Venus program. We could have a fallback of “X” amount of clicks in case the acceleration is not sensitive.

6.3.4 Resolution of Issue

We poll some random(or ourselves/family/etc) users in a gameplay test, have their mouse movements captured(method unknown at moment) and average out the data to set a baseline for mouse acceleration. This will give a good way to set requirements for Venus.

7 Appendix

- The House Source Code
  - GitHub - arturkot/the-house-game: Simple adventure game written in HTML, CSS and JS.
- Game
  - The House
- Processing data
  - WO/2020/023321 IN-GAME RESOURCE SURFACING PLATFORM
- Mouse Analysis
  - A fluid human interactive proof in virtual environment | IEEE Conference Publication | IEEE Xplore