Katherine Fitzpatrick, Isaac Men, Caitlin Hemming, and Karen Pham aka Team 10 KICKs

Olivieri and Wang

CS 190

18 May 2017

Milestone 3: Ruff Prototype

Your team will have a ruff (puns) prototype illustrating the main mechanics of the game based on Milestone 2. Your team should also have sample sounds illustrating how your sound design choices will integrate in the game.

One document (.doc, .docx, .rtf, or .txt) explaining the current state of the game, the current status of the milestone goal, what each group member worked on, the number of hours worked on the game as a group, and what went wrong / right for the week. Students will also submit their executable file (for Mac or Windows).

Game Idea:  
 There is a child alone in their house trying to find their beloved pet dog. However, there is also a monster chasing after them. The child is in a long “T”-shaped hallway and must use audio/visual cues to find their dog before being caught by the monster. The game is 2D with an overhead view that uses certain keys from the keyboard.

Current State of Game:

The monster was coded to spawn randomly and move in a straight line. If the player is hidden, then the monster will pass by without affecting the player. If the monster touches the player, both the player and the monster freezes, ending the game. In order to continue, the player must restart the game using the “r” key. The monster does not move towards the player when they are in view at this state in the game. The player can control the child using the “wasd” keys. Also, the player can hide under tables in order to avoid being captured by the monster. The dog that the player finds is stationary in a preset location. It also makes barking noises when there is a fork in the road to tell the player their location. If the player went in the direction opposite of the dog, then they get caught right away. The bark is heard from either the left or right side of the speakers/headphones. When the player touches the dog, there will be a text display congratulating the player for winning. The map is a simple, straightforward “T”, and dimly lit so the player should rely mainly on audio cues to find the dog. The monster, child, and maps have rough designs that have not been finalized yet. Some of the sounds are chosen, mixed, and added to the prototype.

Status of Milestone Goal:

Completed Milestone #3 with a rough prototype.

Group Logistics:  
What each member worked on:

Katherine: “Other” Programming Stuff :D

* Monster Code
  + Catching child
  + Movements
* Hiding Mechanic for child
  + How to win/lose
* Debugging Code

Isaac: “Lighting” and Programming Stuff :D

* Flashlight Mechanic
* Child Code
* Insert sounds
  + Dog in a forked path
  + Monster
* Debugging Code

Caitlin: Sound stuff :D

* Choosing Sounds
* Editing/Mixing
* Different sounds of the monster
  + Mechanical, inorganic
* Flashlight sounds for on/off
* Other

Karen the Pirate: Arrrrrt stuff :D

* Monster Designs
* Child Character Design
* Hallway design
* Writing the document

Number of hours worked on the game as a group:

Around 7 hours.

What went wrong:

Some of the sounds we initially wanted/searched for were not what we expected. For example, the bacon sound we found was too wet, thus we decided not to use that sound. We were unsure of how to do actual lighting for the game, but eventually settled on using pseudo-lighting with custom sprites. Another problem occured when we pushed a file in GitHub; it changed some of the files already there, so we had troubleshoot. Eventually, we solved most of the issue. Also, we wanted to darken the environment to include a flashlight mechanic that would allow the player to see only with the light on. However, we encountered problems with the scaling and implementation of it so we turned the feature off for now. At one point the monster was able to spawn but it wasn’t spawning fast enough nor was it spawning frequently enough. We discovered this when playtesters were able to complete the game before monsters spawned. In order to resolve this, we made the monster spawn instantly and decreased the speed of the child. Debugging the code took the most time, however we were able to make the game run. Also, there were many ideas in the design of the game and what to include, but we had to pick and choose which ones we needed. For sound, there were problems in balancing and shift pitches, but we figured it out using Wwise by getting more familiar with the program. Initially we wanted to have sound coming from front to back, but we figured that directional sound can best be distinguished from the left or the right.

What went right:

Everything else. All four of us were able to meet up together for most of our meetings and communicated well. The coding, art, and sound process went okay and we spent a lot of time working on the project. We were able to code the character and the monster. Finding sounds through the sound library was easy since there were many choices available; mixing them together went well too. Figuring out how to code the components was not too difficult. We were able to pull together a pretty ruff prototype. The player is able to move and hide successfully, the monster spawns and can freeze the player, the player is able to win and lose the game, there is sound for the monster and dog, and the sounds were able to be played separately through the left or right sides. A playable game was made with audible sounds.

Possible Ideas to Implement in Final Game:

We still need to edit the sounds of the game and decide on which monster sounds to incorporate in our game. Other sounds such as wind, windows, footsteps, door creaks, house creaks, and more may be added to the final product. The child may make noises to call their dog and cry/whimper when they are scared. The dog sounds could be expanded for variety such as growls and barks. There can be variation in the way the monster looks when they get the player and a menu screen/instructions can be added.

For game mechanics, we need to give the player the ability to open/close doors and enter different hallways. When a player chooses the wrong door, they could be trapped and the monster gets them. They will have a flashlight that points straight ahead no matter the direction the player walks/faces. However, the player is able to move the flashlight clockwise using the “e” key and counterclockwise using the “q” key. The flashlight would reveal a small portion of the map. If the flashlight hits the monster, it will go away.