CPSC 490 Project Proposal
Colin Dunn
Spring 2015
Adviser: Julie Dorsey

MOTIVATION
While the computer science department at Yale offers many challenging and intriguing opportunities for its students, there does seem to be an absence in its push for interdisciplinary projects. For this project, I hope to remedy this situation by reaching across disciplines to partner with a fellow student from the art department. Our common ground falls in the form of video game creation, which combines the perfect balance of technical and aesthetic design. Having played video games almost my entire life, I have an unpursued interest in processes involved with their creation. Driven by an interest in both computer science and the arts, I will be partnering with a colleague, who specializes in graphic design, to create a two dimensional video game.

OVERVIEW
For this project I am partnering with David Shatan-Pardo, MC ’15, who will be using his work on this game as a credit for an independent art study. The game we intend to create will be a two dimensional, single player game. The game will be offline, and will take the form of a puzzle-platformer. The player will have to progress through horizontally and vertically scrolling levels by jumping, climbing and using a variety objects in the environment. At times, the player’s progress will be halted until they are able to solve an environment based puzzle that will enable them to progress through the level.

As there are many puzzle-platformer games in existence, our game will need a gimmick to ensure its originality. At any given point throughout the gameplay, the world in which the player is currently in will be accompanied by two alternate, parallel worlds. With the push of a button, the user will be able to switch between these worlds at will. The worlds will be largely similar, yet will have very key differences, such as differences in platforms or special objects. To progress through certain parts of a level, or to solve specific puzzles, the player will need to travel between these worlds, using the differences between them to their advantage. Each of these three worlds will have a consistent and unique aesthetic style, giving the user a tool for visual exploration. Inspiration for this game stems from works such as Portal, Fez, Braid and Super Brothers Sword and Sworcery EP.
DETAILS

From a technical standing, the project will be programmed in C++, and will be written and maintained using Windows Visual Studio 2013. On top of C++, the game will use SDL, Simple DirectMedia Layer, a cross-platform software library developed for the creation of high-performance video games. The game will initially be designed to be playable only on the Windows operating system, but with the use of SDL, we hope to port the game to Mac OS X and Linux with relative ease. Despite being a library for gaming, SDL is not a game engine. A game engine is a framework that generally contains a rendering engine, a physics engine, collision detection, animation and sound support, and many other aspects that ease gameplay development. I will not be using an existing game engine; instead I will be creating my own using the SDL library. As I am not using a preexisting game engine, there will be a large variety of game attributes that I will have to handle manually. Some of the major attributes are as follows:

- **Rendering**
  
  Using SDL, I must design an efficient way of creating a game window, and rendering textures to that window. Rendering is the basis of gameplay, as without it, there will be nothing for the user to experience. Rendering must be simple and efficient, as it will be extremely extensive throughout the game. In addition to the rendering of a character and a basic level, rendering must be done for background layers, objects and items, and non-player characters, as well as menus and the basic user interface. In addition to basic rendering, I must create a mechanism for clipping and rendering sprite sheets. Sprite sheets are a long sequence of images, each representing a different frame of movement. They are the basic tool for creating animations in 2D games, and will be used in all aspects of the game ranging from player motion to loading bars.

- **Level Organization**
  
  Level organization is extremely important, and it provides the environment with which the player interacts. Without the environment, there cannot be a platformer. Levels can be broken down into varying parts, mainly the static parts of an environment, and the dynamic parts of the environment. For many 2D games, the static environment is generally represented by tiles of a fixed size on a grid system. This is the approach that will be used in our game. The dynamic portion of the environment is more difficult to organize, as it may contain a variety of objects such as moving platforms, special items, and non-player characters. An effectively organized level is extremely important, as it can significantly reduce loading time.
Collision Detection and Physics

Collision detection is extremely important, as it is the bridge between a character and a level. If a character finds at any moment that their movement overlaps with a non-accessible portion of a level, the character’s movement must be restricted. This will be one of the most challenging implementations, as collision detection occurs every frame, and thus must be quick to prevent the game from slowing down. Using a grid system, collision detection is simple for solid blocks, but becomes much more challenging for slopes. Physics is definitely an important addition as well, as a character’s acceleration, maximum velocity, jump force, and gravity have a large effect on gameplay mechanics.

The end goal of the project is not to create a game with hours of gameplay, but instead to create an engine that allows for simple creation of levels and efficient play. By the end of this project we hope to have three complex and playable levels that demonstrate all of the unique aspects of our game. If the tools designed through this project are successful, this should allow for the transition into a full-fledged game to progress with ease.

DELIVERABLES

February 1

• Collision Detection for Solid Blocks
• Collision Detection for Slopes
• Animations and Sprite Clipping

March 1

• Doors, Buttons and Keys,
• Level Organization and Loading
• Level Designer

April 1

• Level Design – 3 Playable Levels
• Sound and Story Design
• Story Design

May 1

• Finishing Touches
• Testing and Demonstrations
• Final Paper