Design Document for:

Astronaut Geometry

*A hodgepodge game about learning Maths*

“If it’s for kids, why is your debug mode a feature?”

All work Copyright me 2022

Created by Patrick Vance

V0.5.0.0

Tuesday, June 21, 2022

# Design Document History

This section will explain the history of this document

**Version 0.5**

Initial construction of document in preparation for peer feedback

# Development History

This section will outline the design process and changes to the overall game due to iterative development and feedback

**Version 0.1.0**

Initial design concept for the game

1. Created low fidelity prototype for level design
2. Created low fidelity prototype for UI infrastructure
3. Began to design level concepts for the three levels

**Version 0.1.1**

Scaled back on initial design concept after receiving peer/teacher feedback

1. UI concepts simplified to reduce clutter
2. Simplified and streamlined gameplay elements including pickup quests and level puzzles
3. Scaled back overall scope for pre-alpha preparation from four levels to one

**Version 0.2.0**

Initial game construction

1. Designed basic level architecture
2. Added player controller and playercontroller scripts
3. Scripted UI elements and pickup objects

**Version 0.3.0**

Adding full elements to first level

1. Added NPCs with dynamic text
2. Added scripts for resolving clues
3. Added debug mode for testing
4. Imported objects used for interacting with level

**Version 0.3.1**

Fiddling with colliders, physics materials, and linear drag

1. Considered scrapping entire project over physics issues
   1. How can 2 colliders possibly act so poorly together
   2. Created a custom physics material and modified standard player physics to bypass physics issues

**Version 0.5.0**

Finalization of core scripts and completed rough design of first level

1. Having all scripts in place will exponentially speed up further development
2. This version marks ‘Rough Draft’ content release

# Overview

Astronaut Geometry (otherwise known as AG) is an educational game made for students between grades six and ten with a primary focus on teaching geometric concepts. The overall model for the game is scoped as an action/adventure platformer where the player controls a moveable astronaut and navigates individual levels to find clues, culminating in a level end that tells a geometric formula or illustrates the solution to a geometric equation.

Visually, AG is designed such that individual level objects and platforms, when possible, maintain a consistent theme representing a specific geometric shape. The first level, for example, is comprised entirely of square tiles that are square (blocky) in nature, with square pickups and a square puzzle to interact with. The player will retain their individual astronaut character throughout levels, and in some cases objects or NPCs will break theme to maintain scope and feasibility.

The primary purpose of the game is education to satisfy the following learning objectives:

* Introducing young students to geometric concepts
* Providing geometric facts to enhance knowledge
* Showing geometric formulas through understandable visual constructs
* Drawing parallels between world design and underlying geometry

# Problem Statement

Current educational games that facilitate learning geometry through play only cover basic overviews and identification of shapes and do not delve into middle-school educational requirements for geometry, such as geometric formulae and calculations.

# Project Purpose

This project aims to create an intermediate-level experience to challenge middle-school students to continue their geometry education through an active adventure platformer-style game. The game will be comparatively more challenging and engaging than current geometry games designed for younger children.

# Gameplay

The overall game is subdivided into individual levels; in a given level, a player must complete two to four tasks. After completing a task, the player is given a clue (mathematical fact) that will help them learn about a given shape or geometric formula. Tasks include but are not limited to the following:

* Interacting with NPCs
* Gathering collectible objects
* Interacting with objects multiple times
* ‘Push-Pull’ puzzles
* Navigating an area within a time limit

When a player has collected all clues for a given level, they will navigate to the level-end NPC and interact with them. The NPC will then provide dialogue answering the challenge for the level.

Tasks and puzzles will take on a variety of the above-listed forms, but generally be thematically related to the level’s geometric shape architecture.

# Mechanics and Interface

The player is placed within a bounded world which is defined and constrained by a level-based tileset. Default unity physics apply for gravity; the player is able to jump or double-jump to reach higher elevation to navigate and complete levels. To compensate for oddities with the Unity physics engine, tiles are supplied with a custom 2d physics object that provides 0 friction, giving a ‘sliding’ locomotion feel, and slowed by linear drag on the player character.

A main staple of player interaction is the interact key (E), which serves composite functions of talking to NPCs, interacting with objects, and picking up objects for push-pull interactions.

As they do not fit the scope of this game, there is no inherent economy, and combat does not exist in any form.

The main camera is centered on the player and does not dynamically move. The UI is also statically anchored to the main camera and drawn in front of all other objects. The UI displays a current timer for the level, what clues have been collected, and progress on pickup collection for that level.

Various interactive objects are placed in the levels. Currently, the following objects exist:

* Four distinct NPCs to talk to
* A television that can be freely pushed using rigidbody physics architecture
* Two doors that allow teleportation to and from an auxiliary level add-on
* A square used for the first level’s puzzle that fills in with smaller squares to demonstrate area

The game is instanced and will restart upon loading the game executable. Saving will not be an option or feature in the current design scope. Currently, there are no transitionary level screens or options menus.

Future iterations will include the following:

* An options menu that pauses the game timer including the listed options
  + Volume sliders for SFX and music
  + Quit to desktop
* A ‘help’ submenu
* Keyboard display as input reminders for movement and actions
* A pause button (P)
* A start menu that displays before the first level
* Transitionary scenes between levels that serve as a pause function and level recap

A debug mode is included in the current game – the player can press Z to toggle an infinite jumps mode. Currently, this debug mode is listed by the first NPC

# Story, Narrative and Backstory

In the initial design of the game, story and individual character development was forsaken over functionality and gameplay. Future iterations will include a small, unobtrusive overarching storyline tying the levels together, with a few repeat characters including the initial ‘help’ NPC and the level-end NPC. The characters will have a light backstory of generically trying to help the player achieve their goals for altruistic purposes and/or a general love of Mathematics.

# Levels

The current game iteration is comprised of a singular ‘Level 1’ which is also considered the training level as it introduces the control scheme and general constructs of the game world/objectives. The goal of level 1 is to ultimately find the formula to calculate the area of a square. Along the way, the player picks up the following clues:

* A square has four sides
* All side lengths of a square are the same
* You only need one side measurement to find the area.

Movement with WASD keys is assumed as prior knowledge by the player, but will be iterated into a static sign at the start of level 1. The player interacts with a NPC near their spawn point to learn about collecting clues, learn about the Z toggle for jumps, and learn about the amount of clues for level 1.

Clues are earned in the following ways:

* Speaking to an NPC for a ‘free’ clue
* Collecting 9 neon squares placed around the game map
* Teleporting to an offshoot puzzle room and completing a simple puzzle by pressing E 9 times

The level itself is fairly open-ended within the tile constraints and there is not a singular defined path that the player needs to follow. As long as all three tasks are completed, the player may end the level at any time. Tasks may be completed in any order.

There is one NPC in the level that serves simply as dialogue and a false flag to discourage the player from entering the puzzle room; the player needs to ignore this false flag and proceed to collect a clue.

# Development and Major Programming Concepts

This project was developed in Unity using their 2d engine for ease of object and UI creation.

Custom coding in C# was used to create (currently) ten different scripts to control game function including but not limited to player movement, dynamic text options for individual NPCs, transition modes between individual levels and sublevels, and individual puzzle programming. Scripts were designed with the specific focus of being malleable and able to be fluidly applied between levels, increasing initial scripting time but reducing the amount of required recoding going forward.

Prefabs were used in some instances where repeated object iteration would be expected, such as collection puzzles and story puzzles.

This project demonstrates enumeration, encapsulation, access of object parameters through other objects, object instantiation, and polymorphism.

# Project Constraints and Other issues

As this is a one-person project, there are some deficiencies in overall look and feel of the artistic direction for the game.

# Continued Development

This project is not currently in a complete form ready for final release. In addition to peer and instructor feedback, development aims to accomplish the following additional goals:

* Construction of a second and third level
* Introduction of a basic, unobtrusive storyline
* Improving physics on movement and dropping
* Addition of transition screens between levels
* Custom free music for level backgrounds
* Increase in congruence (when possible) of art direction for entire game
* Introduction of full options menu controlling sound, controls, and help text