The Bigger Picture

Design Documentation

Flinn Fraher

Game Overview

Description

2.5/3D puzzler with 2D platforming, interaction between the 2D and 3D space, and 2D physics-platforming puzzles.

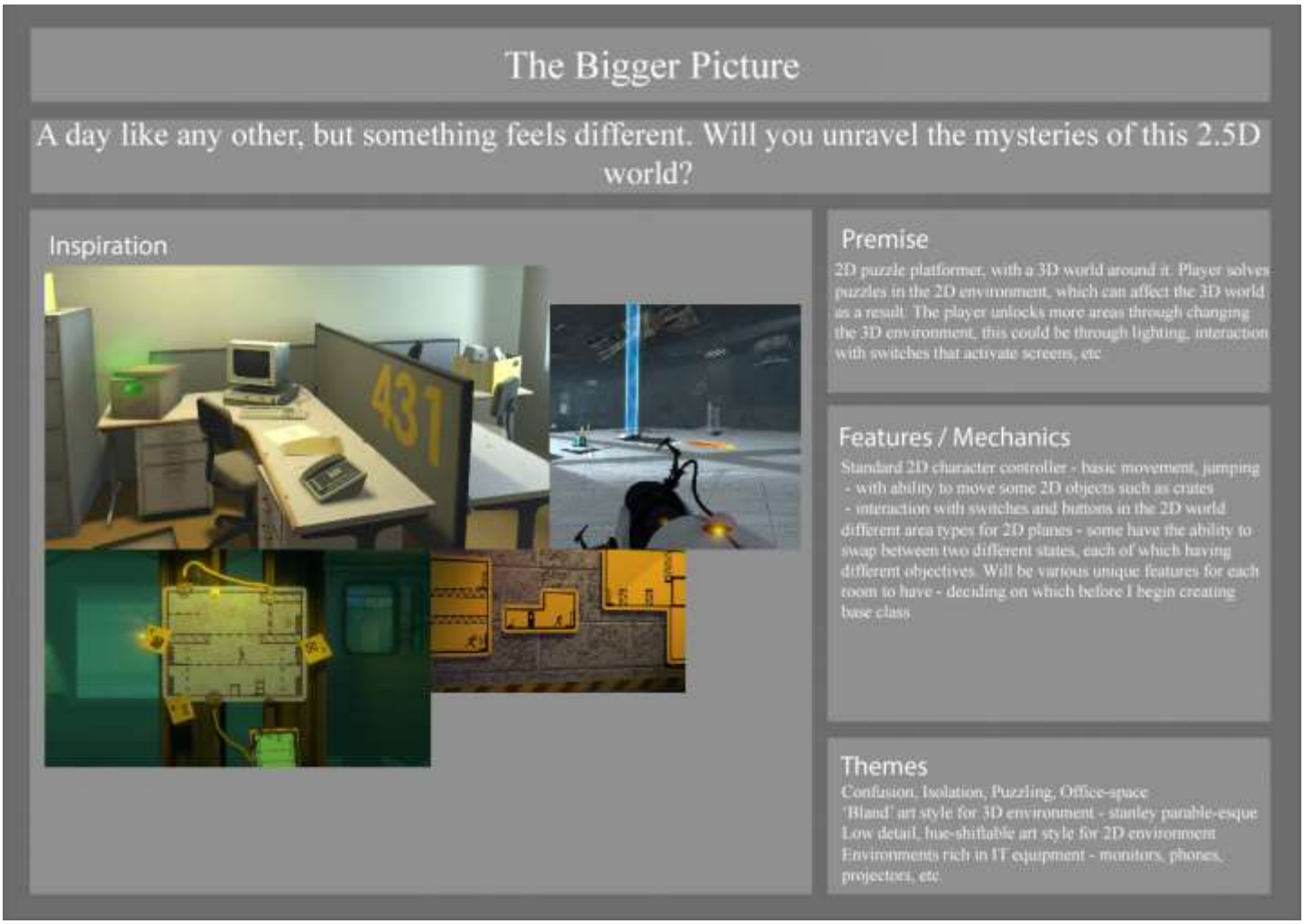


FIGURE 1 - ONE-SHEET DOCUMENT

Potential Titles

The Bigger Picture,

Theme

Main themes of the game.

Aesthetic

Overall, the game takes a lot of inspiration from The Stanley Parable in its aesthetics – the gameplay takes place in a small office with 90's style technology littered around the room, along with some elements of modern technology. The accompanying soundtrack is a jazz/soft rock mix (not too dissimilar to what could be heard in some coffee shops) to compliment the slower pacing of the gameplay.

3D Assets

most of the 3D models in the game will take on a low-poly art style. This is both to alleviate time spent on asset creation during the development period for myself, and to compliment the simplistic sprites and 2D assets I will be using.

2D Assets

2D assets will take on a simplistic art style – this is to simplify the asset creation process for myself, and also to accompany ideas I have regarding the adjustment of hue in different gameplay areas to keep playable regions feeling unique visually.

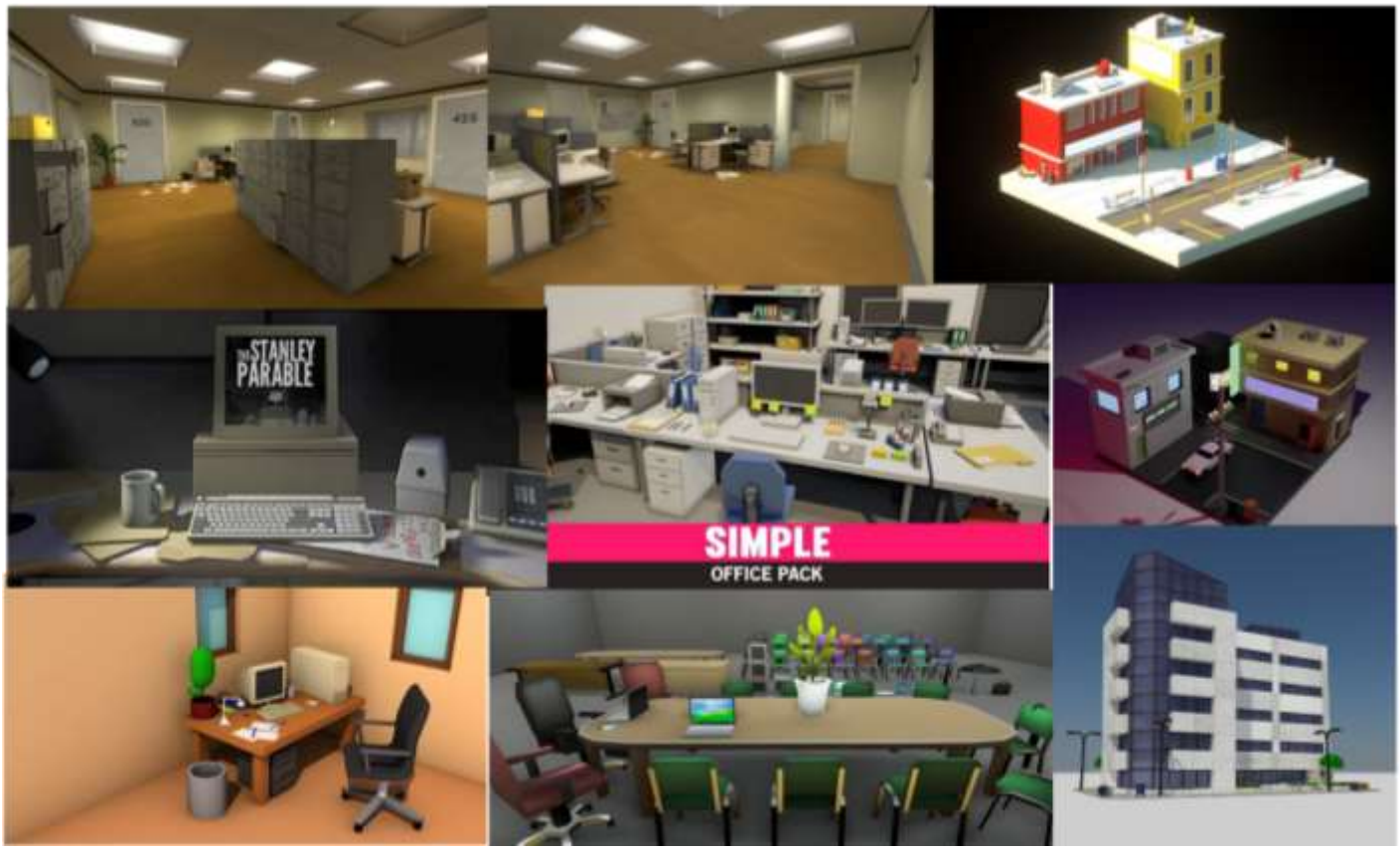


FIGURE 2 - MOODBOARD FOR ASSETS

Core Gameplay

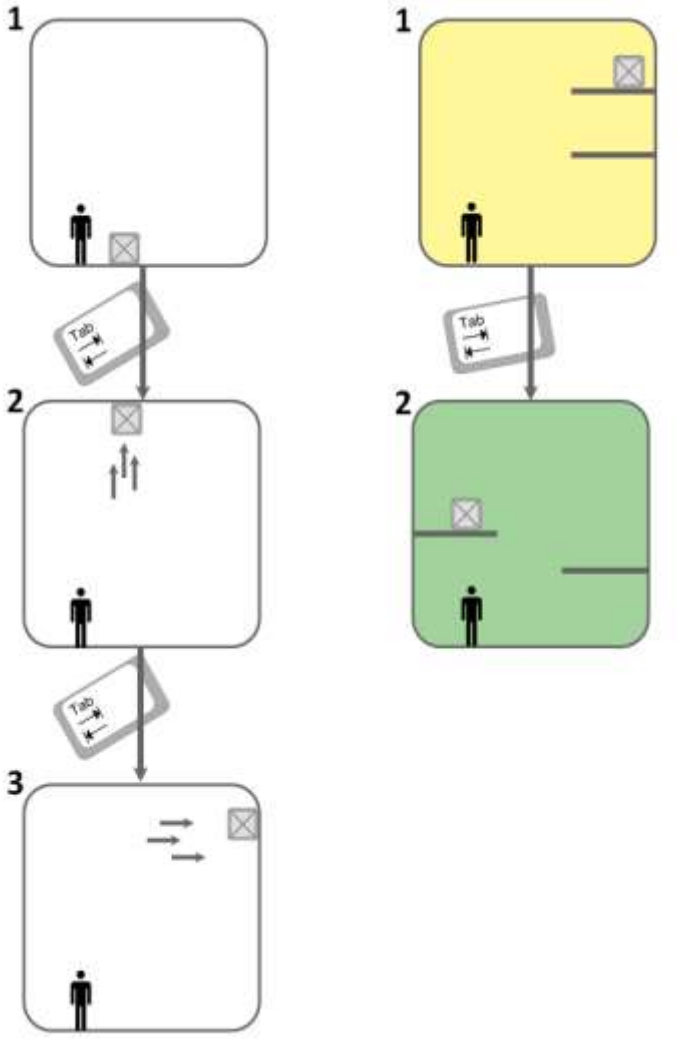


FIGURE 3 - THE DIFFERENT TYPES OF INTERACTIVE ROOMS

Brief Overview

The core game loop of The Bigger Picture consists of the player solving puzzles in order to progress to more playable regions of the office area – tasked with using a discrete set of core mechanics, the player will have to combine all the mechanics introduced to them in order to solve increasingly challenging puzzles around the office space – from moving crates, to manipulating gravity.

The primary task of the player throughout gameplay is to find ways to unlock subsequent areas for the 2D player to move to – these can be accessed in a variety of ways, from moving a crate in the 2D space to a pressure plate, activating a button or lever in the 3D space, to utilising a room with multiple states to activate the next area.

Area Types

Specific areas can be manipulated further by the player. In fig. 3, you can see the two area variants – to the left is the gravity-controlled areas – these playable regions can have their gravity manipulated by the player pressing the tab key, allowing them to move objects to otherwise unreachable areas.

To the right of fig. 3, you can see the second variant of the playable area. When the tab key is pressed inside these areas, the geometry and contents of the room are changed to a different state and can be toggled between each time the tab key is pressed.

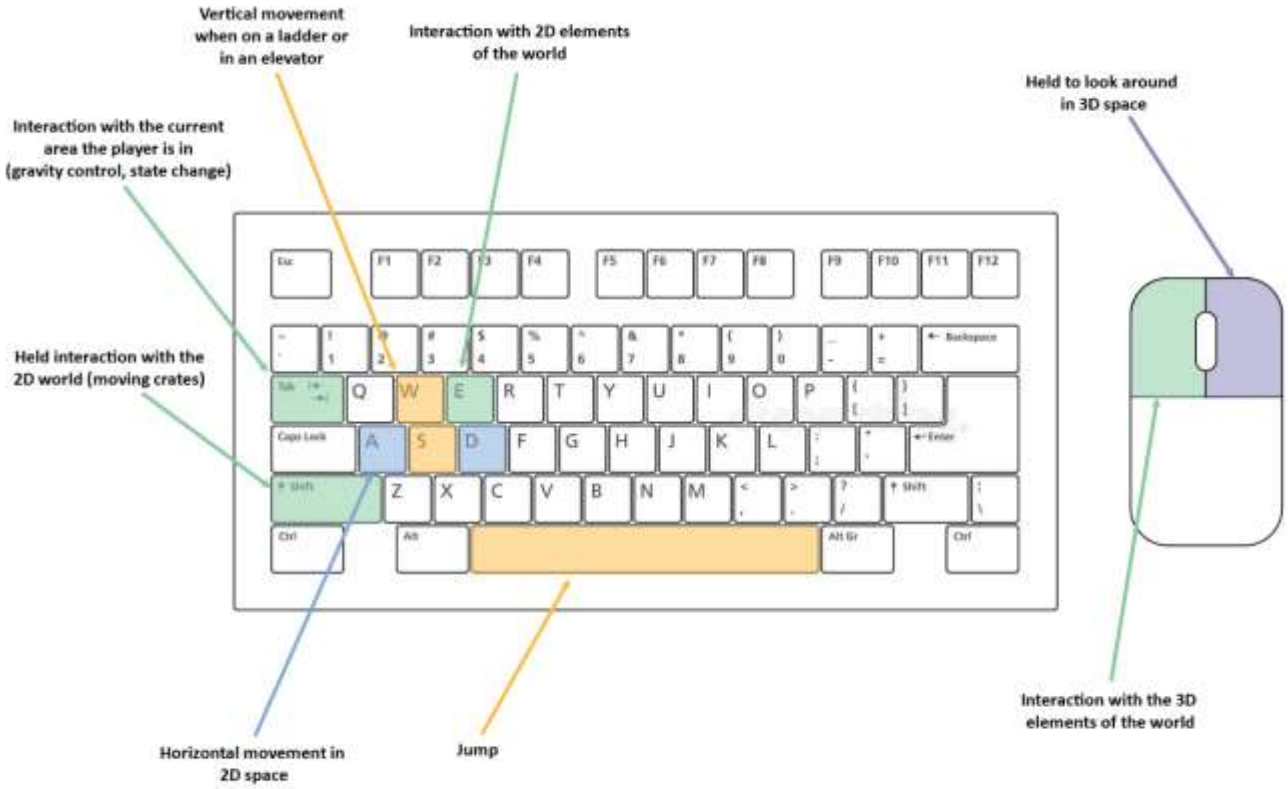


FIGURE 4 - KEYBINDING LAYOUT

Visual Loops

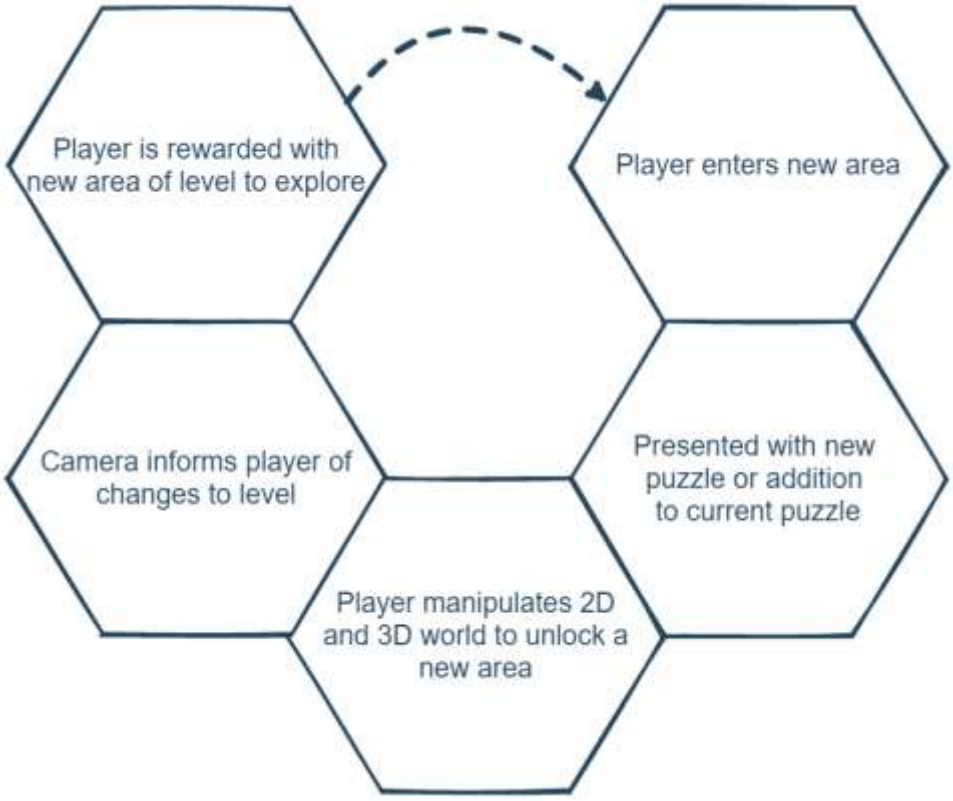


FIGURE 5 - THE CORE GAME LOOP

Mechanics overview

2D Environment Interaction

In the 2D space, the player can interact with various gameplay objects to solve the different puzzles in each playable area. By using the E key (see fig. 4) the player can interact with an item that is within a set radius to them. Other objects which require continuous interaction can be manipulated using left shift – this is predominantly used to move crates and similar gameplay items around the level.

The player is also capable of manipulating some specific variants of areas in the level (see fig. 3) creating greater mechanical depth and greater challenge in later puzzles which involve different area types.

3D Environment Interaction

While traversing the 2D space, the player can move their 3D camera around the level and interact with various 3D objects to altar gameplay. In some puzzles, the player must activate buttons and other interactable in the 3D space, to progress to the next 2D gameplay area.

Camera Management

The main camera in the game is moved between different key points in the level with the use of a camera management system – when certain gameplay events occur, the camera can be moved to a specified location for a duration of time, smoothly moving from one point to another. The camera also moves along specified curves created with Sebastian Lague's Path Creator tool[1], allowing the camera to move between playable areas that are a longer distance from one another in a predictable and smooth way.

Non-Core Gameplay

The player can look around the office space and interact with various office props placed around the level and move them around by holding the left mouse button – the intention of this is to keep the surrounding 3D space feeling alive as the majority of gameplay takes place within the 2D space. Being able to tinker with objects around the level also encourages the player to look around in the 3D space while playing, which is key to some puzzles that are in the critical path of the level.

Player Motivation

Alongside engaging the player with various puzzles, the primary hook of the gameplay loop is the ability to move around the 3D space of the level by moving the two-dimensional player around – the resulting interaction on the 3D space creates a satisfying resulting gameplay loop, which encourages the player to traverse more of the level and unlock more spaces to explore.

Inspiration

Theme / Style

Inspiration for the theme / style of the game.

Mechanics

Inspiration for the mechanics / features of the game.

[1] https://assetstore.unity.com/packages/tools/utilities/b-zier-path-creator-136082

Features

MoSCoW Breakdown

Must

- Camera management – movement around the 3D scene based on 2D position of player.
- Player interaction with gameplay objects
- Interaction with current playable region (gravity-controlled areas, multi-state areas)
- Area activation/deactivation
- 2D area connection system – movement between different areas
- Base ambient events – hint system for tutorials

Should

- 3D stylised environment
- Sound manager, with ambient sounds and a backing soundtrack
- Dynamic, varied camera movement – along splines for longer distances
- Area reset system.
- Damaging lasers which destroy some gameplay objects and cause the player to respawn.

Could

- Lighting-based puzzles involving the playable regions – 2D areas which can be revealed by activating a light in the 3D space.
- Additional surface types in the 2D space – ice, liquids which prevent the player moving key gameplay items in different ways.
- Different hazards in puzzles – damaging liquids using fluid simulation, standard patrol enemies and moving lasers.

Would

- Varied levels taking place in different environments.
- Additional 3D environment interaction – physics puzzles which take place in the 3D space (moving cables into sockets to activate an area, for instance)

Narrative

Setting

The game takes place in a retro-modern office environment – fitted with a variety of technology from various eras to obscure any relations the player might make in time frames that could harm overall immersion into the environment. The use of both modern technology and old accompanies the overall mood of the narrative, which invites the player to explore the game world with the unique mechanics in place.

Mood

The overall presentation of the game will take a satirical form – the company that owns the office building the game is set in are known only as 'The Office Corp Ltd.' And the background soundtracks are moderately slow jazz melodies to accompany the overall tone of gameplay.

Delivery

The primary narrative of gameplay is delivered through the use of in-game prompts, environment assets that shed some light into the office outside of the empty state the player finds it in, and the introductory text inside the main menu.

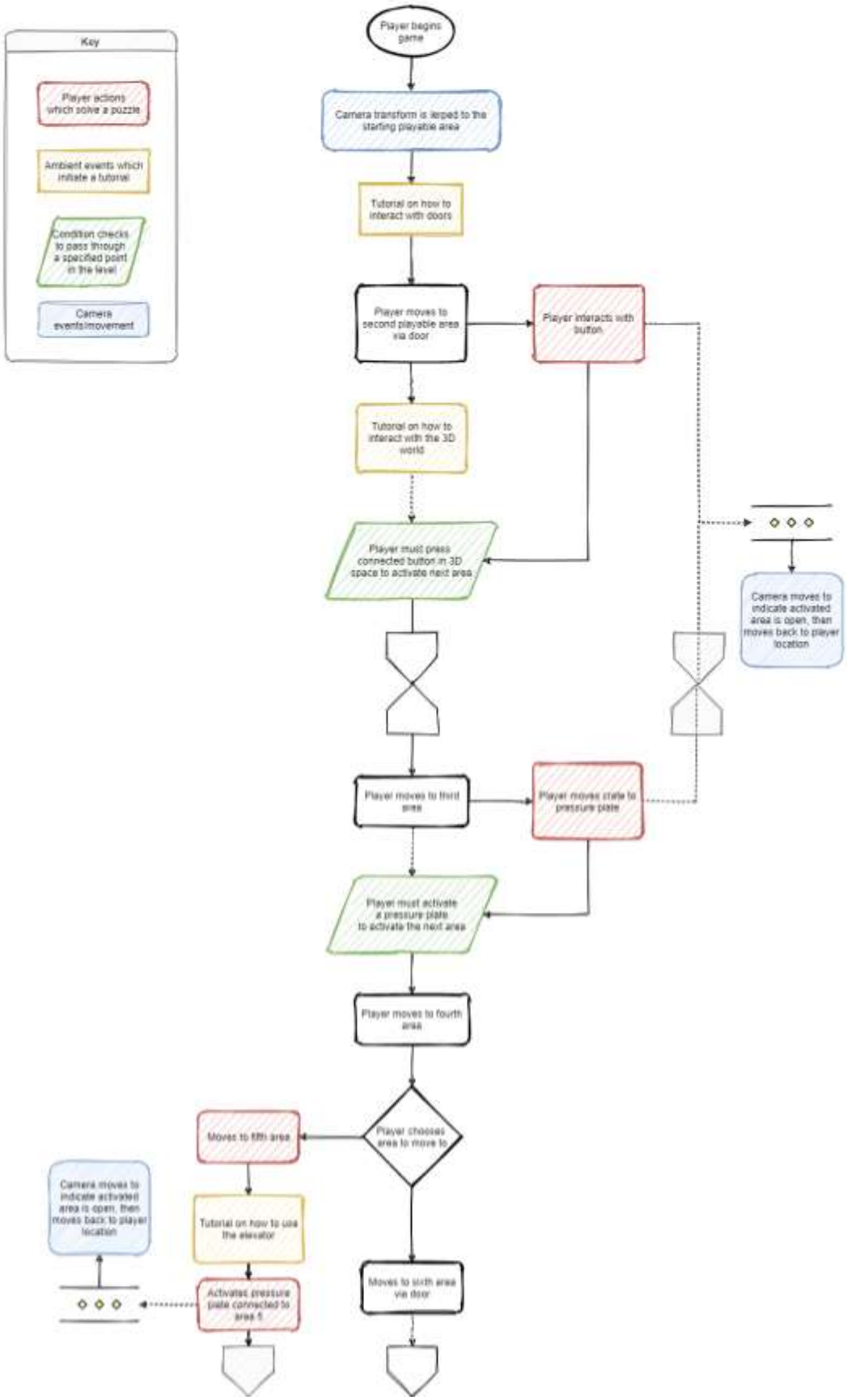
Plot

Game plot points.

Levels / Areas

Each playable area in the level will consist of a single 2D orthographic camera, with a tileset applied.

Primary Gameplay Flowchart



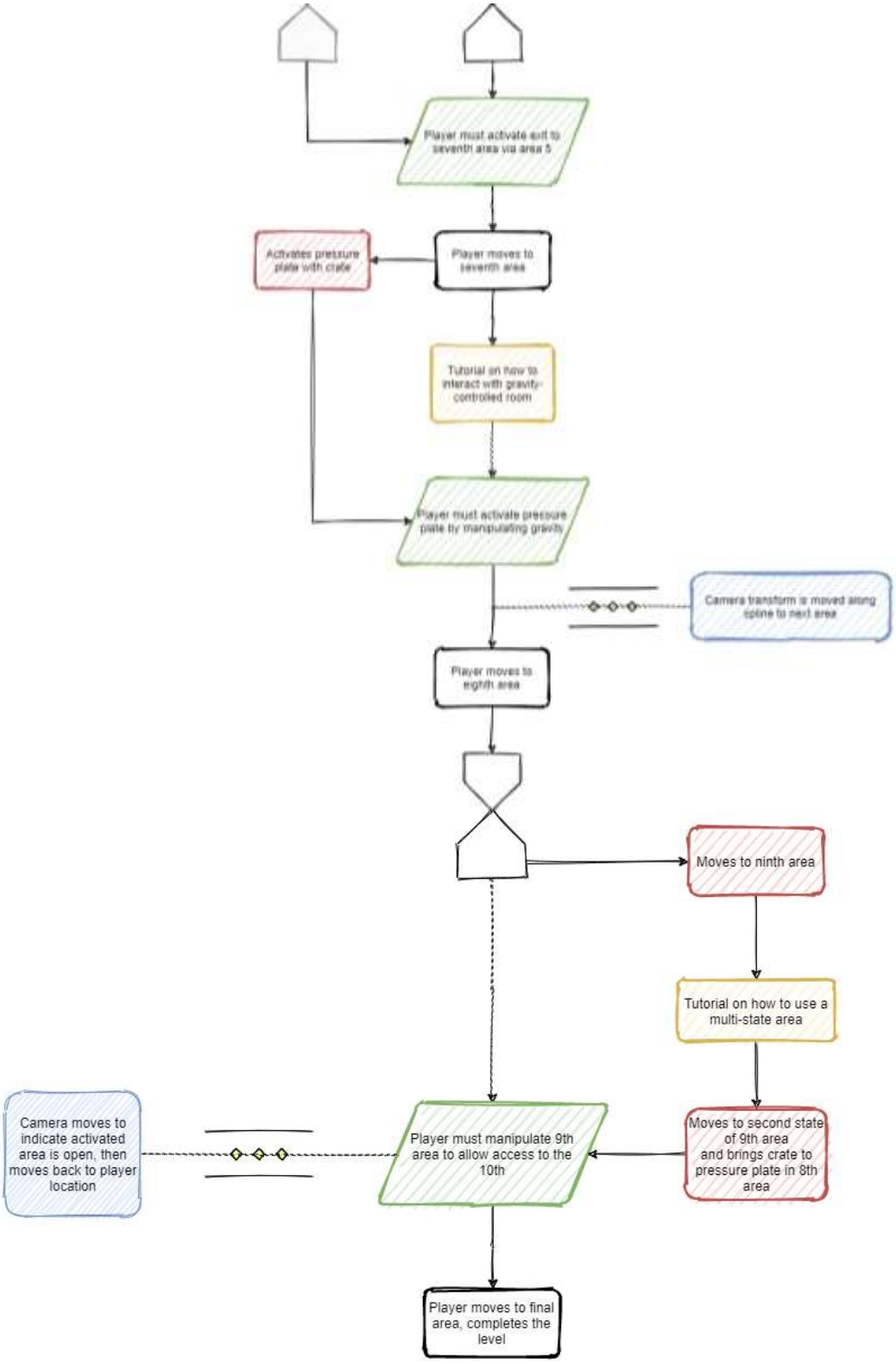


FIGURE 6 – FLOW DIAGRAM OF THE FIRST LEVEL

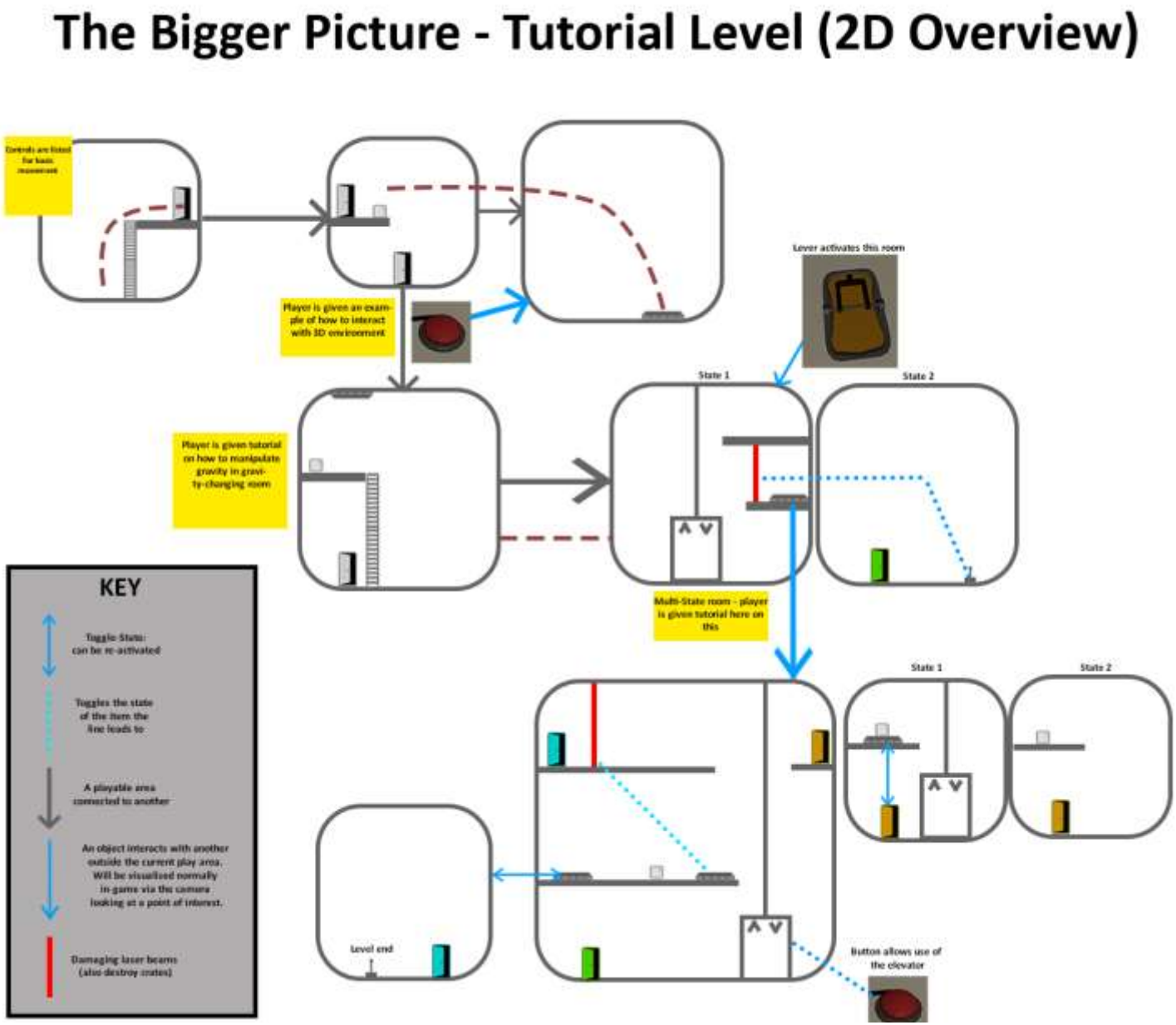


FIGURE 7 - MAP OF THE FIRST LEVEL

Framework

I have included a brief technical overview of the framework created for the playable areas in the project below. For a more detailed look at the technical production of this project, see my breakdown video accompanying this submission.

Overview

To manage the playable regions in the game, a singleton pattern is used in the form of an areas manager. This manager handles the current state of each playable region – if a region should be activated at any point during gameplay, the areas manager can call subsequent methods inside the camera manager and player controller to drive gameplay and make new areas accessible to the player. Each component of the framework is independent from one another, with the use of interfacing allowing a predictable initialisation chain to take place between gameplay objects, and interaction between each manager being handled in a way that adheres to separation of concerns.

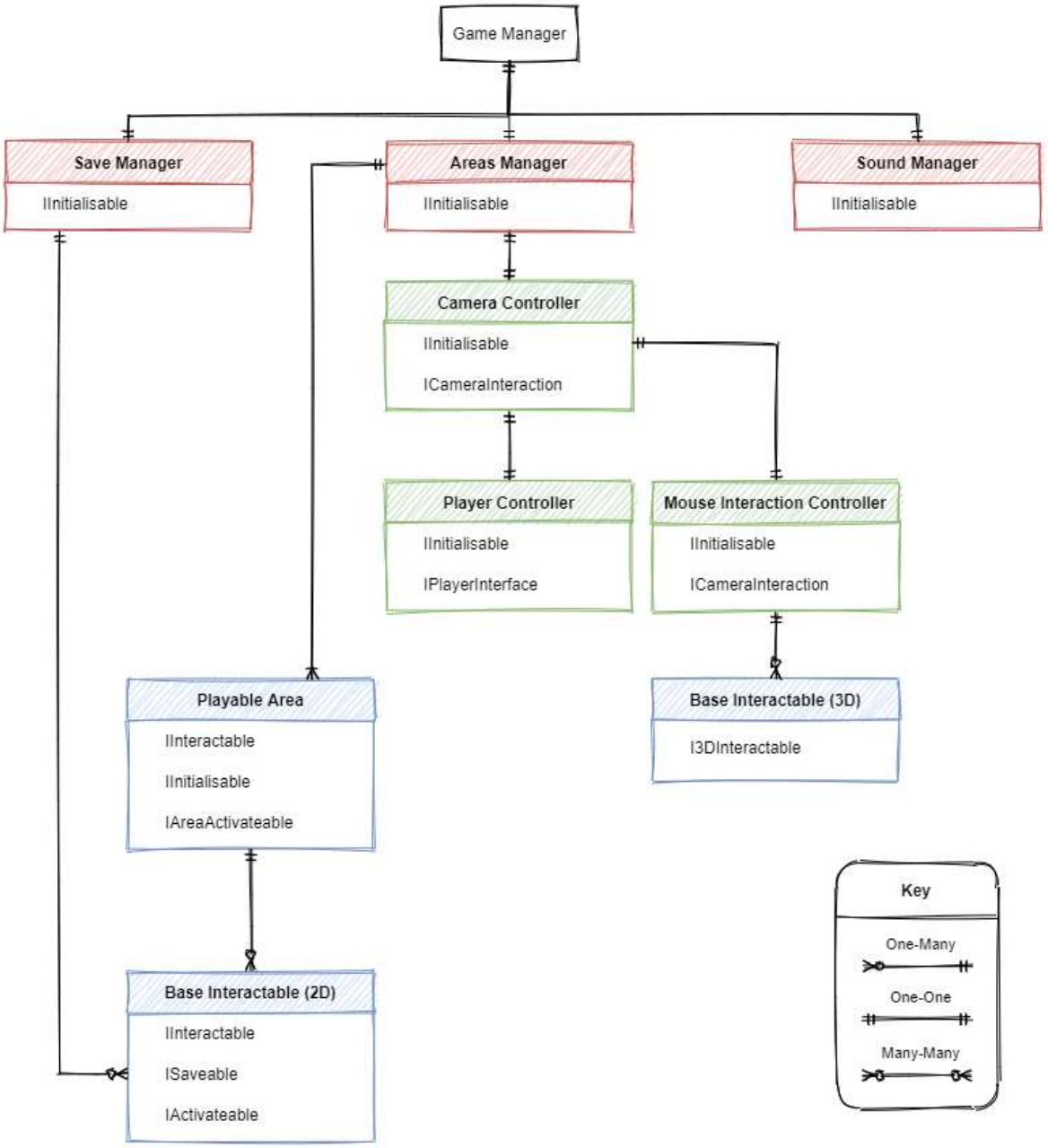


FIGURE 8 - CLASS DIAGRAM OF THE BASE FRAMEWORK

Asset List

Below is the finalised asset list for my project. For a more detailed breakdown, please refer to my sprint tables via the Digital Academy Forum. I will also be appending a change log here, which will detail the contingencies that have taken place if I have been unable to add a specific feature in the production timeframe.

Assets

Below is the asset list of the current game, based on the current build of the game and the accompanying sprint sheets created for the sprint forum posts.

Asset Name	Asset Type	Production Time	Brief Description	Source
2D Playable Areas	Script/Component	10-12 hours	The 2D playable areas in the game – and all sub area-types	Self-made
Camera Controller	Script/Component	10 hours	The 3D Camera controller in the game – drives the controller in various ways based on the 2D player	Self-made
2D Character Controller	Script/Component	7-8 hours	2D player movement and interaction with other gameplay objects	Self-made
2D Interactable Objects	Script/Component	6-7 hours	All 2D interactable objects and their respective scripted components.	Self-made
Base Trigger Colliders for ambient events	Script/Component	2-3 hours	Base triggers in the 2D space responsible for driving UI, and some gameplay events	Self-made
2D Continuous Interactibles	Script/Component	1-2 hours	2D objects which are driven by continuous interaction by the player character	Self-made
3D Interactable Objects	Script/Component	3-4 hours	3D Objects which are driven by player interaction in the 3D space	Self-made
Game Manager	Script/Component	2-3 hours	Singleton which is responsible for initialising all other objects which require it	Self-made

			and general logic for driving some gameplay elements	
Areas Manager	Script/Component	4-5 hours	Singleton which manages the current 2D playable areas in the scene.	Self-made
Sound Manager	Script/Component	4-5 hours	Singleton which manages the current audio clips being played during gameplay. Driven by other game objects	Self-made
Main Menu Controller	Script/Component		Manages player input concerning the main menu interface, and drives the UI GameObjects responsible for main menu functionality	Self-made
Pause Menu	Script/UI	2-3 hours	Simple pause menu, with graphical and sound options that allow player customisation	Self-made
Player model	3D/Animations	1 hour	Mixamo animations and the necessary animation controller logic for a simple player character	Mixamo/Self-made
2D Tileset	2D Sprite/s	<1 hour	Basic tileset for the 2D playable areas.	Self-made
3D Assets – Office Pack	3D Meshes	<1 hour	Various 3D Meshes from the Unity asset store to accompany self-made meshes – see sources for specific packs.	• Simple City Pack Plain • Snaps Prototype: Office • POLYGON Office Building • Low Poly Office Props • Low Poly Office Props - LITE
Bézier Path Creator	Script/Tool Library	<1 hour	Bézier path creator with editor utilities, used for cables in the 3D scene and	Sebastian Lague's Bézier Path Creator

			camera movement along longer distances by sampling the curve	