CS7038 Group B

Hao Guan, Yaffy Koyo, Saloni Sharma & Jeremiah Dunn

1 Theme

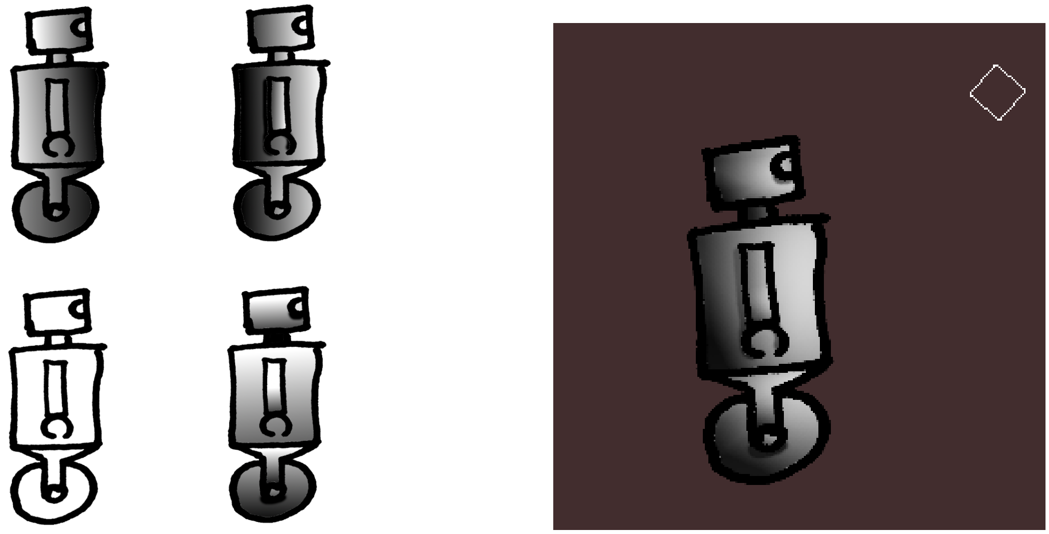
In our game the player controls a maintenance robot on board a spaceship on a long journey. While on the journey the ship malfunctions and begins shutting down, the whole crew are in stasis and it is up to the robot to restore power.

The player must traverse the ship, reactivating components and lighting the ship up in the process. The gameplay takes inspiration from exploration based platformer games such as *Spelunker*.

2 Gameplay

Movement and jumping will be implemented using Unity's 2D physics engine to handle.

Re-activating components of the ship has a probability of activating some of the ship's security measures. These will include activating traps or spawning hostile defence drones. On reaching the next component these security systems can be deactivated.



(a) input files including a base file and the same file drawn as if it was lit from a handful of different directions

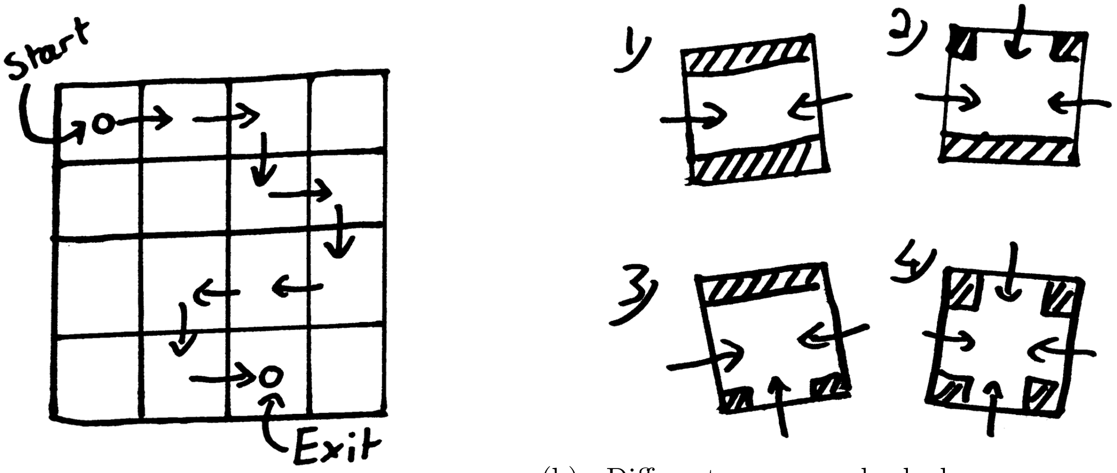
(b) output file reacting to a dynamic light shown by the white cube

Figure 1: Player character mock-up

We plan to use 2d sprites, with normal maps created using *SpriteLamp* for the game's lighting effects. Figure 1 shows an example of it's use.

3 Level Generation

The game will use elements of procedural generation to create levels. To do this, first a grid of *rooms* is created as shown in figure 2a. A random position on the top of the grid is chosen to be the starting room. From here there is a probability of stepping left/right or down. If the stepping algorithm reaches a wall it will automatically step down. At the bottom of the grid the probability of stepping down is replaced with the probability of creating a level end.



(a) Room grid

(b) Different rooms, hashed areas represent floor/roof

Figure 2: Level Generation

After the room grid is created the individual rooms are populated with pre-existing room tiles. To get a basic version of the game running only the four room tiles show in figure 2b would be needed with solid rooms for the rooms off the solution path. Individual rooms will be tile based and will be stored in text files which the game will read in and then populate.

4 Management & Version Control Software

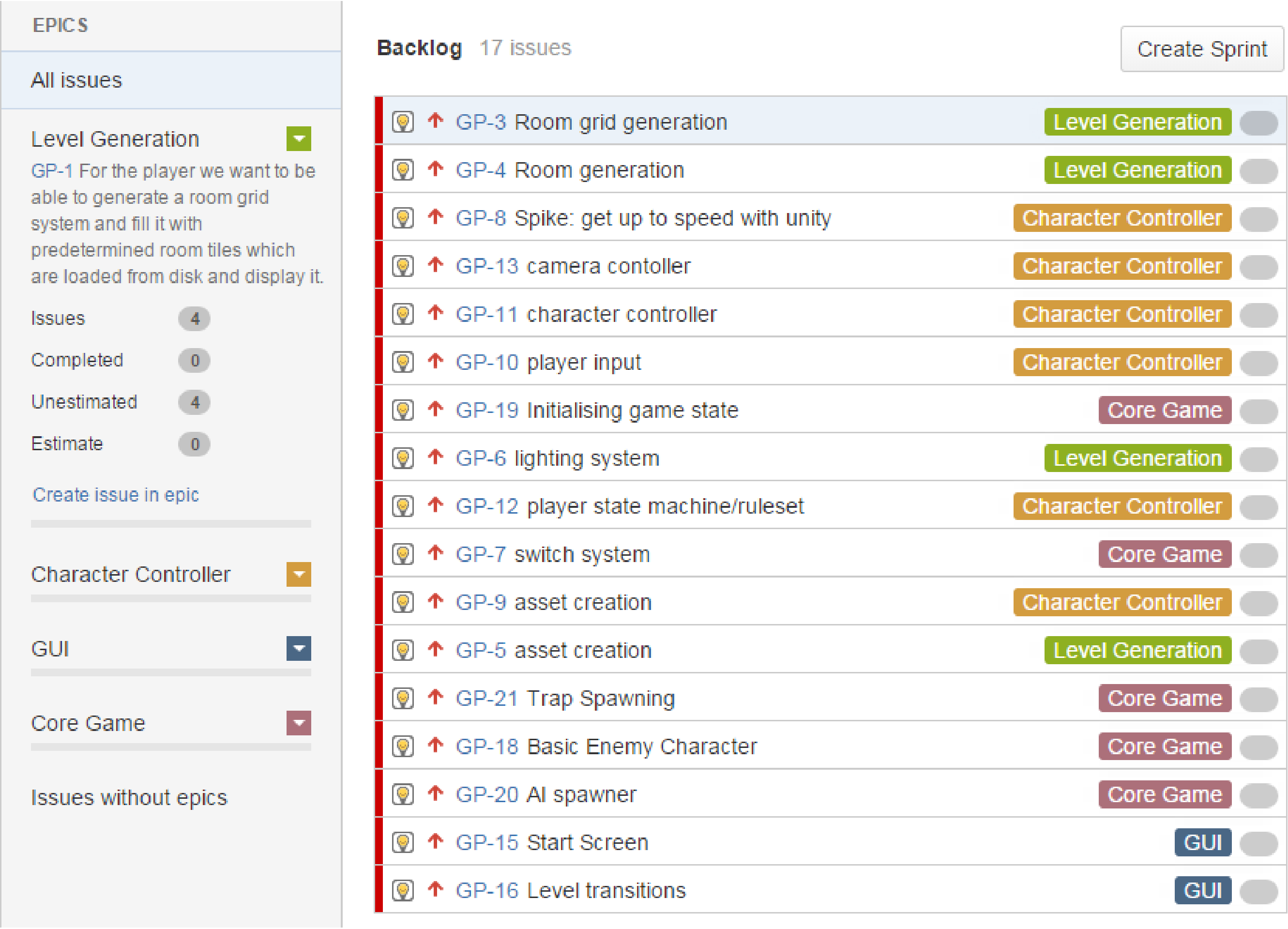
For version control we are using git. To manage creating stories, logging time and creating burn down charts we are using JIRA which is deployed on one of our machines.

5 Project management

After a week of discussion on the game project idea, all main tasks were subdivided into Epics: *level generation, GUI, game logic* and the *character controller*.

The epics are spread over the duration of the project and are broken down into a number of smaller stories as shown in figure 3. Tasks for each sprints will be chosen from the backlog based on their priority and requirements.

Project would be divided into 4 sprints. At the start of each week we will have a grooming and kick-off meeting. In the grooming meeting we will look through the backlog and see what stories we want to work on, assign story points and flesh out the descriptions or subtask them as required. In the kick-off we will assign tasks and give time estimates.



EPICS

All issues

Level Generation

GP-1 For the player we want to be able to generate a room grid system and fill it with predetermined room tiles which are loaded from disk and display it.

Issues	4
Completed	0
Unestimated	4
Estimate	0

Create issue in epic

Character Controller

GUI

Core Game

Issues without epics

GP-3 Room grid generation	Level Generation
GP-4 Room generation	Level Generation
GP-8 Spike: get up to speed with unity	Character Controller
GP-13 camera contoller	Character Controller
GP-11 character controller	Character Controller
GP-10 player input	Character Controller
GP-19 Initialising game state	Core Game
GP-6 lighting system	Level Generation
GP-12 player state machine/ruleset	Character Controller
GP-7 switch system	Core Game
GP-9 asset creation	Character Controller
GP-5 asset creation	Level Generation
GP-21 Trap Spawning	Core Game
GP-18 Basic Enemy Character	Core Game
GP-20 AI spawner	Core Game
GP-15 Start Screen	GUI
GP-16 Level transitions	GUI

Figure 3: Screenshot of Storyboard on Jira

6 Meeting Plan

Sprint-Plan meetings would organized in the beginning of each sprint to discuss about the execution plan for the sprint tasks and Sprint-Demo meetings would be organised to discuss and demonstrate the executed sprint tasks.

The team will have stand-ups or kick-off/grooming meetings three times a week. Stand ups will aim to last for 10 minutes with any additional discussions happening in separate meetings outside this time.