Computer Science Project: Core Project Document

Game Development Group 3

3 December 2021

1 Group information

Game name: Beat & Boost
Group number: GD 3

2 Theme and interpretation

The chosen themes are: "No where to hide", "From a magnificent creature". The game will be a fighting game where players fight on platforms floating in the air, in several rounds. As the user loses a round he shall receive a random power-ups, from a set of available ones (e.g. more damage). In essence, the player will have no where to hide from the magnificent creature that is the strengthened opponent.

3 Game idea

Genre : Round Based Platform Fighter
Our game will be a platform fighter game. The platform fighter genre is beloved by many, we decided to put our own spin on it by substituting stocks with a round based system (i.e. first to X round wins). However, to make things more interesting, a power-up is selected every round from a random assortment, but the previous round winner does not get one. These power-ups stack throughout the rounds, so if a person loses every round, they should have a power-up advantage to compensate. There are different 'classes' of characters you can choose between, before the fight. The characters will be divided into multiple offensive and defensive character archetypes. There will be a variety of stages and stage types. After the fight, so after multiple rounds, a new fight can begin, and you can choose an other character and stage.

4 Student information

Group GD 3 exists of 5 students. You can read all about them and their roles below.

- Producer/Game Tester — Aleksander Śliwiński (5016266)
  A.J.Sliwinski@student.tudelft.nl
- Lead Artist/Game Tester — Bas van Beelen (4924797)
  b.p.e.vanbeelen@student.tudelft.nl
- Lead Programmer/Game Tester — Hashim Karim (4811518)
  H.Karim-2@student.tudelft.nl
- Game Designer/Game Tester — Simon van Leent (5097363)
  S.J.vanLeent@student.tudelft.nl
- World Builder/Game Tester — Vincent Goossens (5186625)
  v.a.goossens@student.tudelft.nl

5 MoSCoW prioritization analysis

Must have

- Movement mechanics (such as wave dashing)
- Simple pathfinding
- Local multiplayer
- Character animations + power-up/ability animations
- Variety of power-ups
- (Close range) combat

Should have

- Animated textures
- Custom game art
- Main menu
- Knock percentage (as opposed to a health system)
- Save relevant information from game during play: hits given, attacks missed etc.
- AI opponents

- Simple AI: AI follows and attacks player
- Different Character Archetypes (both offensive and defensive)
- Match settings (stocks/items/rounds/etc.....)
- Hue/saturation of textures (multiple skins)
- Multiple AI's
- Complex Ai: AI responds differently to character types and players with powerups
- Level heatmap
- Collect and show highscores remotely
- Detailed game analytics (play by play, heatmap etc)
- Power-ups are Archetype specific
- Camera effects
- High scores

Could have

- Xinput/Dinput support (controller support)
- Dynamic Maps: Stages that move, activated stages, stages that appear and disappear, stages with stat effects (give off damage or slow down player)
- Leaderboard/ranked ladder
- Item mode
- Match history
- Different skins (possibly a user/community made content manager)
- Replay viewer
- Online multiplayer
- Match chat
- Phone controller

Won't Have

- Machine learning AI
- Ray tracing
- Voice Communication
- 3 dimensional movement and combat
- Mobile support (playing native on Android/iOS)
- Profanity filter

6 Feature highlight

The core of the game consists of combat in a 2D stage, and power-ups players get after every round.

6.1 Combat

The most important feature of the game will be local multiplayer, close range combat. This can be expanded to longer ranged combat, but this is not essential for the game to function. All other features will be made to support this game play, in essence the game will be a fighting game on a 2D stage.

To realize this, a stage of bar-shaped platforms will be built in Blender to later import to Unity. Initially the platforms will have no material on it and just consist of basic grey blocks. The platforms itself will be locked in position. Also, a basic model of the character controlled by the player will be made in Blender. After importing both in Unity, the player will be given the option to move the character sideways and jump upwards, so one movement direction will be locked. Gravity should be introduced, and the platforms should prevent players from falling down. At first the camera position shall be locked, too.

A second character will then be added to the scene, controllable with different keys than the first one. The characters will be given basic attacks which will have a given damage value, as well as a vector to give the attack a direction. Pressing the attack keys will trigger a player animation. If the attacking player collides with another player while doing the attack, a function call will be sent to the attacked player. This will take the damage and vector variables from the attack as inputs and will accomplish two things: First, the health variable of the player will increase with the damage of the attack multiplied by a damage taken variable. Second, the player will 'fly-off' in the direction of the attack vector. The magnitude of this impulse will be dependent on a Knockback multiplier and the health variable of the player.The round ends when a player collides with the boundaries of the stage (which are defined by planes). These boundaries will be a bit further away from the visible edges in the main camera, so they will not be visible. After the round is over, the winner of the round will get a point, while the other player will get a powerup.

Also, they will be given materials on the surface to make it more visually appealing. The amount of attacks can be extended with heavier but slower, of lighter and faster attacks, or some ranged attacks. New types of characters can be introduces which are capable of different attacks, such as attacks with higher range. Furtermore, each of these characters will have distinct variable values to make them more class like. For example, a 'heavy' character will have a low 'damage taken' and low walking speed in order to make him play differently.

6.2 Power-ups

The feature that distinguishes the game from example given Super Smash Bros. is the possibility for the losing player to choose and use power-ups that apply in the next round. A set will consist of multiple rounds, and after each round the losing player may choose a power-up from a set of options. The options will be randomly generated with the System.Random() function, to create an element of luck in the game. The power-ups will initially consist of stat boosts, such as raising damage, range or damage resistance. Also there will be power-ups that allow the player to jump higher and move faster, for example. These will be implemented in the next round. The chosen power-up will be passed on to the next round.

At first ideas will be made for power-ups and simple images (for example png or svg files) will be made to represent the power-ups in screen. A scene for selecting power-ups will be made in Unity, the Round Menu scene, and the power-up images will be shown and will be made selectable. After a player chooses one power-up, it should modify a or some variables of the player such as damage per attack. This is done in the fighting scene. Initially a player will be given a few, for example three, standard options of power-ups and will be able to select one each round. A set could consist of three rounds. After each round the winning player will receive a score point, and if a player reaches more than half the maximum amount of rounds in score points the set will be ended. When a new round starts, the players will have combat again but with the effects of the power-ups applied. The new round scene will detect the chosen power-up and apply it to the player that has chosen it. There, the losing player now can jump higher for example, if this player has chosen Jump Higher as power-up. The power-ups only last one round, so the next round they disappear and the new power-up is applied.

Later on the amount of power-ups will be expanded. A player will then be given random options for power-ups after each round. The power-ups will not be equal in how beneficial they are for the player, to introduce a factor of luck. The amount of rounds and available options for power-ups could be reconsidered as well.

7 Rough timeline

Week number	Dates	Tasks
3	22-28 Nov.	Finish menu features Introduce movement (jumping) Introduce health system Finish stage and movement prototypes Finish CPD extension 1 **(Deadline: 26 Nov. 23:59)** Implement basic attack (punch or kick)
4	29. Nov. - 5 Dec.	Implemenent movement and jumps Introduce health system Design and implement first power-up (damage boost) Make base character model for player Finish the first stage design and make platforms in Blender CPD Extension 2 **(Deadline: 3 Dec. 23:59)**
5	6-12 Dec.	Introduce multiple extra power ups (speed, attack range, size) Start implementing should have's Work on AI: add a computer-controlled player Start on game analytics: scoreboard, leaderboard Playtesting of the game Finish all must have's Make designs (not yet models) for characters and stage Peer review other groups' games Add game analytics
6	13-19 Dec.	Playtesting Implement game analytics Make art for game play backgrounds Release early access game **(Deadline: 17 Dec. 23:59)**
7	20-26 Dec.	Finish all should haves and must haves Playtesting Refining character design
Holidays	20 Dec. - 9 Jan.	Improve AI Refine character and stage models and animations Make extra player models and stages
8	10-16 Jan.	Polish game play using game analytics Refine/fix should have's and must have's if necessary Work on a could have feature Bug fixes and tweaks Release beta game **(Deadline: 13 Jan. 23:59)**
9	17-23 Jan.	Playtesting Bug fixes and tweaks Devide presentation roles (who presents what)
10	24-30 Jan.	Prepare final presentation Release game (final) **(Deadline: 25 Jan. 23:59)**

8 Star point distribution

The table below shows the distribution and amount of stars for each specific feature, each category and the overall total of stars.

Category	Feature	Stars	Total
Computer Graphics (CG)	Our own 3D assets (player models, stage platforms)	3	**10**
	Custum charachter and world/stage art	3	
	Camera movement (shake, movement)	1	
	Animated textures (custom)	1	
	Our own skeletal animation	2	
Artificial Intelligence (AI)	Simple AI (only attack and walk)	1	**7**
	Advanced AI (uses specific attacks based on distance to player, can block attacks)	3	
	AI pathfinding for the AI player	2	
	Complex AI (can respond, e.g. with blocks)	1	
Game Analytics (GA)	Save information during play with global variables	1	**9**
	Collect and show highscores remotely	1	
	Creating remote server for data storage	3	
	Level heatmap by checking where the player collides	2	
	Web-based replay store/level explorer	2	
Programming (PR)	Movement/attack mechanics (custom component)	3	**18**
	Use unity's full physics simulations	2	
	FPS independent	1	
	Dynamic environment (platforms move, objects destroyable)	1	
	Power-ups (custom component)	3	
	Controller support (custom component, e.g. extra damage)	1	
	Local multiplayer (two players fight)	2	
	Basic menu (main menu) for player settings, names	2	
	High scores with a high score counter	1	
	Option/customization menu (choose player model, stage)	1	
	Self made UI-animations (animations after rounds)	1	
Total stars			**44**

9 Art

9.1 Moodboard

The art style will consist of characters, platforms and backgrounds in the stage with round shapes, and occasionally geometric shapes. These shapes are considered relatively ease to make in Blender, and give a friendly look to the game

Figure 1: Mood board for player characters, stage platforms, menus and backgrounds during game play.

9.2 Assets

The game, at the time of writing, only has one general asset for players with animations for movement and attacks. This asset will serve as the basis for future hominoid player models. The game will later have assets for stage objects (platforms) and multiple player models. The background during game play will be an image, but the camera will move so the visible part of the image will change slightly during game play. The menus will also consist of more round shapes. See the 2 for the overall mood board of the game.

9.3 Non-visual art

At the time of writing, the game does not include any sounds yet. If time would not allow making custom music and sound effects, copyright free music from example given YouTube can be uses. However, if time allows it, it is desirable to make several hit sounds for players hitting each other, and players knocking another player off the stage. Also, self-made background music could be used in the main menu, the 'next-round'-menu, and during game play itself. This means in total three times music (main menu, next round, game play), and at least two times short sounds (hit, knock off stage) have to be made. Since one of the group members already has access to basic audio editing software (Adobe Audition, Ableton Live Lite), it could be possible to compose simple music and edit self-recorded hit sounds with those software.

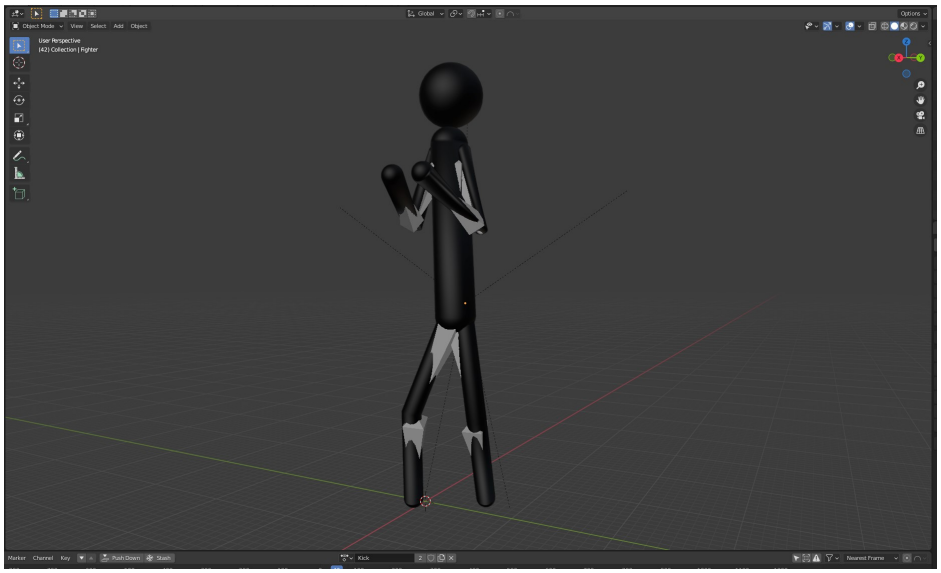

Figure 2: The base humanoid blender model for the player, consisting of a body with movable bones which, their turn, influence the positions of the body parts. Later, the model will be modified for multiple specific player models a user of the game can choose form.

10 User interface and controls

10.1 Menu's

The first interface the player encounters is the main menu. As the game is intended for mainly local 2-player multiplayer, the main menu gives users the option to pick player names. This could be useful when displaying scores of the individual players. The players also have the option to choose a stage (the area the fight takes place). The game should feature multiple, at least two, different stages to choose form. The current menus not consistent with the desired art style yet, but before release it will be rearranged.

Besides the main menu, the other menu a player can see is the 'next round'-menu. This menu displays score and what power up the losing player of the previous round obtains. Users of the game can also choose a different stage when they become available.

10.2 Game interface

The interface during game play consists of a stage with several platforms, with the two players on it. The background will be a still image, but the camera will show slight movement and zoom towards the area the players are fighting. In figure 5 the current basic interface is shown, where the blue block and the white

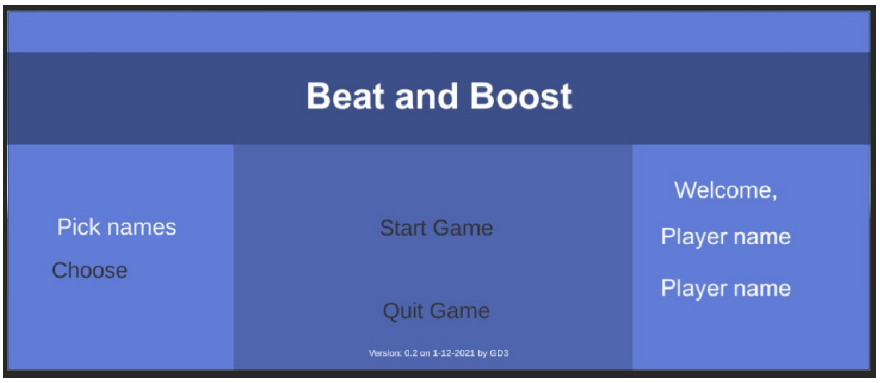

Figure 3: The current layout for the main menu, the first screen a player sees when starting the game.

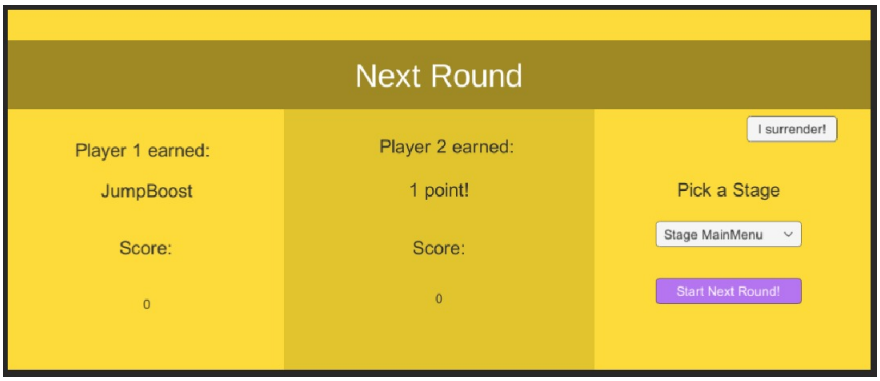

Figure 4: The current 'next round'-menu, used to tell the winner of the previous round, show what power up the losing player obtains, and to optianally select a different stage.

humanoid object are to be changed for player models. If two players would use the same model, they will have different colours to differentiate them more easily.

10.3 Controls

The game is designed to be played by two players on the same computer. A single player option is also available, where the player battles against an AI. It is intended the players both use a controller, for example a Playstation 3 or 4, or an XBox One or 360 controller connected via wire to the computer. Players navigate with the four keys on the left of the controller, or with the left stick. Players can jump with the most northern button on the right set of four buttons. The other three buttons on the right side can be used for attacking

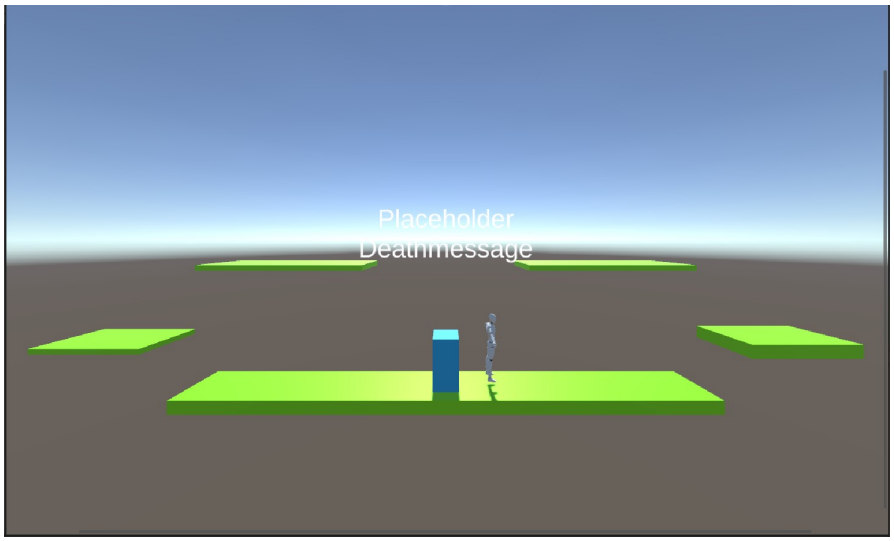

Figure 5: The current basic player interface. Players can use the green platforms to move and jump on. The blue block and the white humanoid object are in this case the two players. Later on they will be substituted with a custom player model (see figure 2).

the other player, of blocking attacks from another player.

Using a keyboard instead of a controller is also possible. A player then navigates upwards, left, down and right with the WASD-keys. A player can jump with space bar. A player can attack or block using the JKL-keys.