# Interactive Graphics A.Y. 2018/19
# Final Project
# “Vaporwave Endurance”
Abbate Giovanbattista 1875271
Di Spazio Fabio 1876540

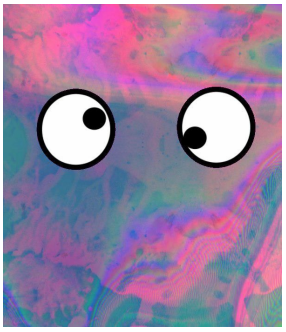
## Abstract

Vaporwave Endurance is based on tower defense game style, but it is reinterpreted in a first-person perspective with a Vaporwave touch. Usually, tower defense games involve defending a base by building towers, which fire at waves of enemies as they cross from one side of the battlefield to another. Usually these enemies are very varied and some may take multiple hits to kill or may have special properties. Our reinterpretation takes these fundamental properties with the difference that you are the firing tower! There are different kind of enemies, upgradable bullets and a tower to defend.
The player earns score by killing enemies and he wins if he reaches 50’000 points.

# Elements

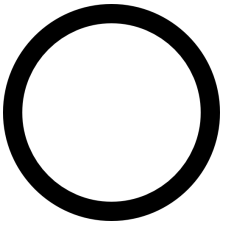
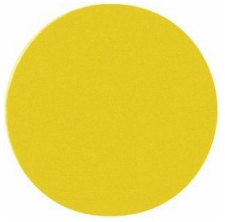
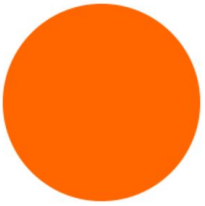
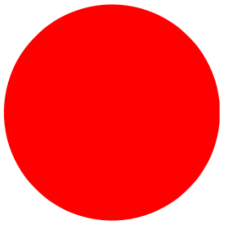
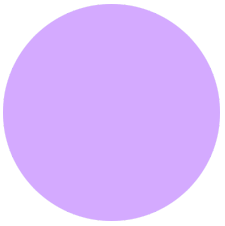
## Enemies

There are four types of enemies based on a different characteristic that spawn making up the waves. Each four waves, their characteristics increase.

| | |
|---|---|
|  | "Easy" composes the first wave and as suggest the name this type is quite simple to kill.<br>Info (first spawn):<br>-Health: 75 hp<br>-Speed: 35<br>-Amount 10 |
|  | “Medium” has a similar speed of the previous one but it got more health.<br>Info (first spawn):<br>-Health: 400 hp<br>-Speed: 25<br>-Amount: 6 |
|  | “Hard” composes a category of very fast enemies.<br>Info (first spawn):<br>-Health: 120 hp<br>-Speed: 50<br>-Amount: 25 |
|  | Finally there is the “Super” type which is very durable but slower than others.<br>Info (first spawn):<br>-Health: 1000 hp<br>-Speed: 30<br>-Amount: 1 |

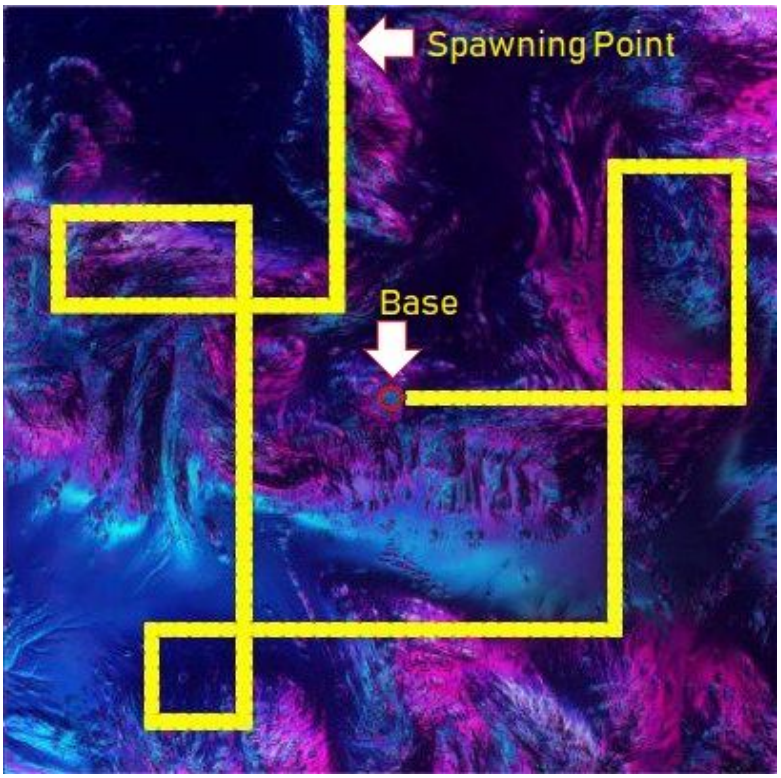
## Bullets

With increasing of cash, the player can update the bullets shot to fight enemies. There are five type of bullets corresponding to five upgrades.

| | |
|---|---|
|  | White Bullet<br>Power: 100<br>Cost: 0<br>Score to unlock: 0 |
|  | Yellow Bullet<br>Power: 120<br>Cost: 500<br>Score to unlock: 6500 |
|  | Orange Bullet<br>Power: 156<br>Cost: 1200<br>Score to unlock: 15000 |
|  | Red Bullet<br>Power: 218<br>Cost: 1300<br>Score to unlock: 26000 |
|  | Violet Bullet<br>Power: 327<br>Cost: 1400<br>Score to unlock: 39800 |

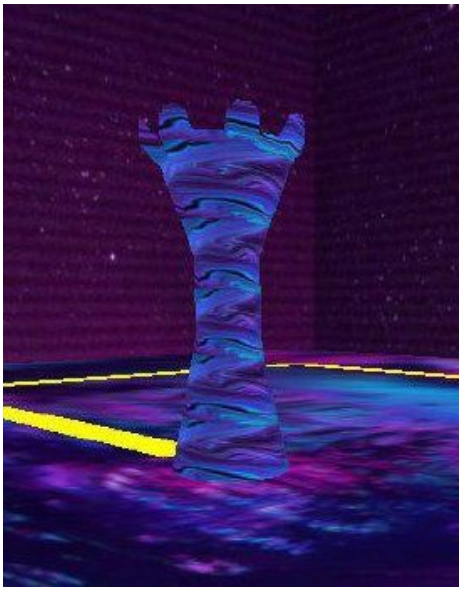
## Battlefield

The battlefield is composed of the map and the path where enemies move.
In the figure are indicated the spawning point and where the base is placed.




## Base

The base to be defended is constituted by a tower at the top of which is placed a firing cannon.



# Explanation of the code

Our code folder is structured in this way:

```
root
├─── js/
│    ├─── Creep.js
│    ├─── Detector.js
│    ├─── jquery.min.js
│    ├─── Map.js
│    └─── Score.js
├─── ThreeJS/
│    └─── ...
├─── models/
│    └─── ...
├─── textures/
│    └─── ...
└─── HTML files
```

## Map.js

This script produces the map on which the entire gameplay is based.  It is divided in 2 main functions.

### Map.initialize()

This function initializes some variables that will be used in the next functions, and an array that stores the coordinates of the path.

### Map.generate()

This function at first draws the entire map plane (using a basic material and a texture produced from an image) on which the path will be placed. After this part,  the path on which the creeps will walk on is drawn step-by-step manually by direction. In this way, it is possible to create loops. For each square composing the path, the coordinates are pushed in an array that will represent the path for the creeps.

## Detector.js, jquery.min.js and Three.js

This javascript file is taken from the projects on the site https://alteredqualia.com/. The script simply gives a nice output in case WebGL is not detected on the target device. The file jquery.min.js is the standard jquery library v1.6.3 available on the site http://jquery.com/. The Three.js library is available at https://threejs.org/.

## Score.js

This script manages all the variables related to the damage of the cannon, the health of the tower, the personal score, the amount of cash, the win/lose condition.

### Score.initialize()

All the variables that will be used in the other Score functions are intialized.

### Score.setScore()

This function increases cash and score according to the correspective killed creep, and returns the win condition when the score is more than 50000.

### Score.setHealth()

This function decreases the health of the main tower when a creep reaches the end of the path. Returns the lose condition if the health reaches 0.

### Score.upgradeCheck()

This function prompts to the user whenever an upgrade is possible.

### Score.upgrade()

This function increases the cannon's damage, changes bullet's color and decreases cash when conditions are met. This function is triggered during the game by pressing U key.

### Score[.getScore() | .getDamage() | .getCash() | .getHealth()]

This four functions return the correspective parameter, in order to prompt it to the user during the game.

## Creep.js

This script manages all the interactions between the creeps and the world.

### Creep.initialize()

This function initializes all the variables that will be used in the script, along with the definition of the creeps' stats for the first wave

### Creep.runLevel()

This function defines the progression of the creeps' stats along the different waves. Each 4 waves, the stats are increased through some parameters.

### Creep.create()

This function creates the creeps. A different texture is applied to each different type of creep. Creeps are basically a sphere buffer geometry, shaped in order to let them seem ghosts. Creeps represent a first hierarchical model of the project, formed by the body of the creep and the health bar, defined as a child of the body. An array is initialized both for creeps and healthbars.

### Creep.update()

In this function, the animation for the creeps is defined. It is possible to see that health bar doesn't require adjustments, because it moves along with the body of the creeps. The function also checks if creeps reach the base, removing them and decreasing the health of the tower. It checks the health of the creeps too, and if the value reaches 0, the creep is declared dead.

### Creep.isHit()

This function is triggered when a collision between a bullet and a creep is defined. The health of the creep is decreased, and the health bar is scaled proportionally to the remaining creep's health.

### Creep.isDead()

This function is triggered when the health of a creep reaches 0. Score and cash are increased according to the killed creep, and the element is removed both from creep and health bar arrays.

### Creep.getWave()

This function simply returns the number of the current wave, in order to print it during the game.

## VaporwaveEndurance.html

This file contains the main page of the game. In the head of the file, some CSS properties are defined for body and the text bar on the top of the window. Then, all the required js files are imported, and the main script is declared.

### Main JS Script

This is the core of the whole project. Since it is impossible to comment on the documentation every single step, some comments are added in the parts that are not reported in the following lines.

### init()

In this function, the parts of the main window are defined. Each part is commented on the code. Other functions will be explained in the following part.

### checkCollision()

This function checks the collision between a bullet and a creep. The starting idea was taken from the project “Black Entity” published on Piazza, adding the check on the third dimension and re-adapting it to work on this project.

### onMouseDown()

This function is the function of the homonym event listener. A bullet is generated, and if the win/lose condition is met, the game ends.

### onWindowResize()

This function adapts the game when resizing the browser.

### drawBox()

This function draws the cube where the entire project’s world is located in.

### drawElements()

This function draws the cannon and the tower from predefined models, applying to them a texture generated from two different images. The material of this two models is lambert material, in order to use different material from the map to allow the user see the difference of the light effect on them.

### refreshValues()

This function updates the values on the text bar on the top of the page.

### animate()

Simple function for animation. Each function that requires real-time update is put here.

### render()

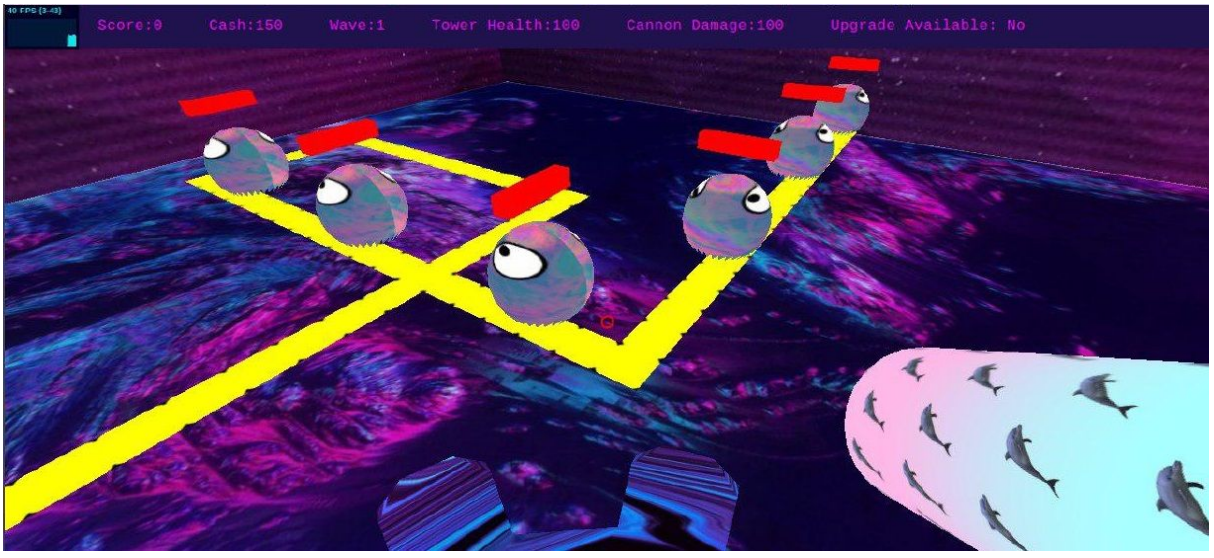
In this function, the position of the bullets in checked, removing them when they cross the plan. The animation of the creeps is performed in this function, along with the win/lose condition check.

# Controls

Once the button “Play” is clicked, the player must click anywhere on the canvas to permit to the browser to catch the mouse entering in full-screen mode. When the page got the control, the user can rotate the camera by moving the mouse and shoot the cannon by left-clicking mouse button. Moreover, if the condition is satisfied, the player can upgrade the bullet by simply clicking the “U” key. (The value to reach to upgrade the bullet depends by score and cash). Finally the player can pause the game anytime by pressing the “P” key and he can also gain the mouse control pressing the “Esc” button.

# Hud

The head-up display presents all important data. In the figure we can see a game screen:




The data starting from left to right are:

- FPS (Frame per second): frame rate expressed in fps.

- Score: score increase killing enemies.
- Cash: credit earned needed to upgrade bullets.
- Wave: the number of current wave of enemies.
- Tower Health: initially it is 100. If one or more enemy reaches the base your tower will lose health points. Once hp goes to 0, the player lose.
- Cannon Damage: measures how much damage the cannon deals to enemies.
- Upgrade Available: yes/no. It indicates to user if he reached the value to upgrade bullets.

# References


[1] https://alteredqualia.com/

[2] http://jquery.com/

[3] https://threejs.org/

[4] https://marcoschaerfcourses.github.io/Black-Entity/

[5] https://github.com/ligerzero459/3DTowerDefense

[6] https://stackoverflow.com/questions/50965025/three-js-shooting-bullet

[7] https://clara.io/

[8] https://www.youtube.com/watch?v=WdrbxCRLNgA