Shooting-Game

Sandeep Singh Sardar and Madhusudhana Reddy Makireddy
{s.sardar001@umb.edu},{m.makireddy001@umb.edu}
University of Massachusetts Boston

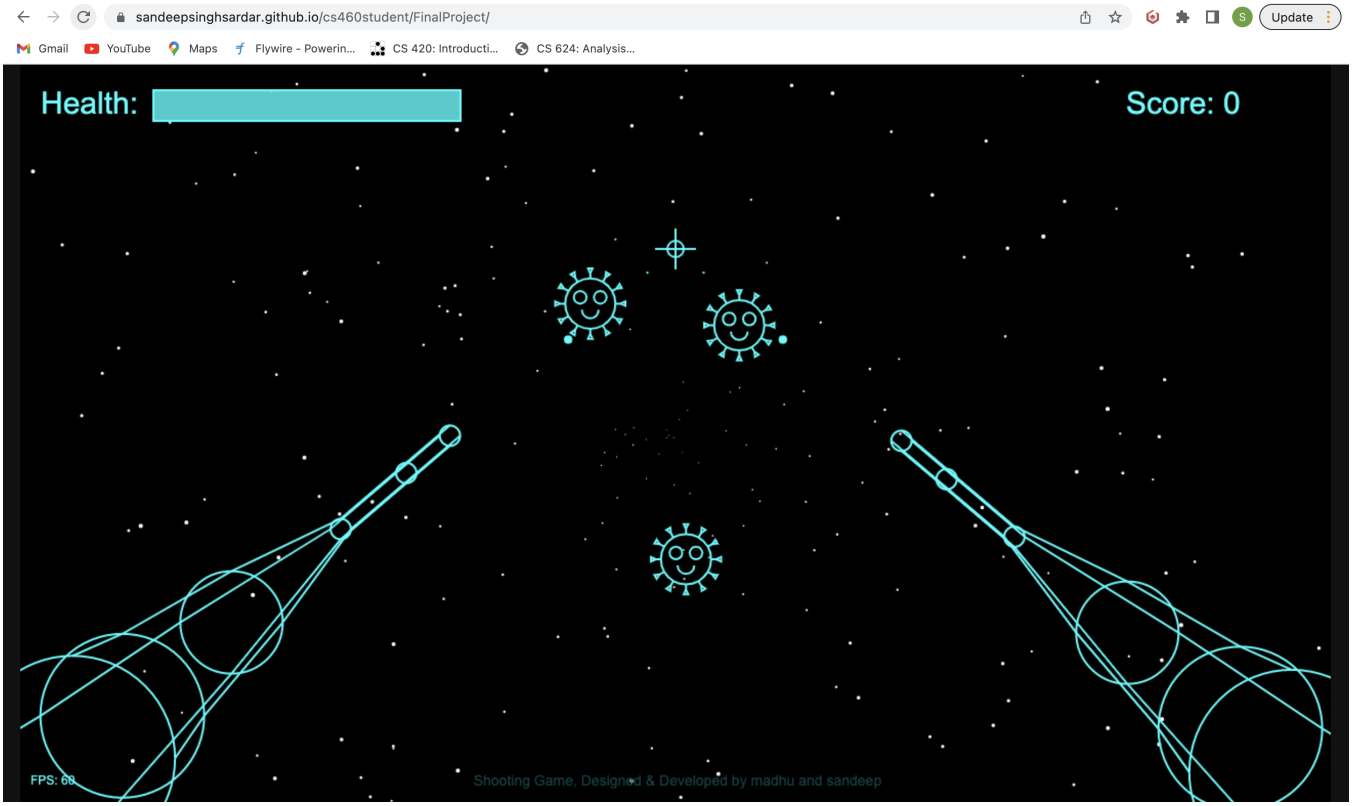

Figure 1: Shooting-Game.

ABSTRACT

The main objective of this Shooting-Game is to kill all the obstacles by shooting them using the two guns. The main goal is to survive and get the maximum score.

KEYWORDS

Three.js, WebGL

ACM Reference Format:
Sandeep Singh Sardar and Madhusudhana Reddy Makireddy. 2022. Shooting-Game. In *CS460: Computer Graphics at UMass Boston, Fall 2022*. Boston, MA, USA, 4 pages. https://CS460.org

CS460, Fall 2022, Boston, MA

ACM ISBN 1337.
https://CS460.org

1 INTRODUCTION

TODO: We decided to do this project because we were inspired by one of our previous assignments where we created a fun game and the main objective of the game was to avoid the obstacles in front of the plane.
In addition to our previous assignment code, we contributed the following things in our project:
- Health Bar and Health Obstacle.
- Bomb Obstacle.
- Score Card.

2 RELATED WORK

This game is based on Three.js and WebGL.
Credits:
- https://threejs.org/
- https://github.com/michaljaz/webmc

3 METHOD

The Shooting-Game is relatively a simple game but as the score increases it becomes complicated. It has lot of objects such as virus obstacle, health obstacle, bomb obstacle and a score counter. To start the game, you need to just open the website. Soon after opening the website the game would start and the virus obstacle, bomb obstacle and health obstacle will automatically start coming towards the screen. The main goal of this game is to avoid the virus obstacle by simply shooting it using the two guns. This actually looks very easy, but if you start playing the game you will find it difficult because the speed of the obstacle increases as the score increases.
When the game starts the virus obstacles comes in blue color towards the screen and if the user does not shoot the blue virus obstacle then it becomes red when it reaches the screen and it drastically starts reducing the health. The user has access to two guns, which he can control using the mouse and hit the obstacles using the bullets of two guns simultaneously. To make the game more interesting we added a bomb obstacle, if the user shoots the bomb obstacle then all the virus obstacles on the screen would be killed and along with this we also added a health bar and health obstacle, by seeing the health bar the user can get a real time health percentage during the game. Likewise, we also added a score card to monitor the score and if the user would kill one virus obstacle then the score gets increased by 10. When the game ends you can just click on the retry button to start the game again.

3.1 Implementation

```
body {background:#111;margin:0;padding:0}
                canvas {background:#000;display:block;cursor: none}
```

Figure 2: This is the implementation of the black background.

```
healthprog += (health-healthprog)/5;
ctx.fillStyle='#00ffff';
ctx.font = "30px arial";
ctx.textAlign = "left";
ctx.textBaseline = "middle";
ctx.fillText("Health: ",20,40);

ctx.fillText("Score: "+score,stage.w-200,40);



// ctx.fillText("Step:   "+(Date.now()-steptime),20,120);
if (health>30) {
    ctx.fillStyle='rgba(0,255,255,0.8)';
    } else {
    ctx.fillStyle='rgba(255,0,0,0.8)';
    }
    ctx.lineWidth = 2;
ctx.fillRect(130,25,healthprog*3,30);
ctx.strokeStyle = "#00ffff";
ctx.strokeRect(130,25,300,30);
```

Figure 3: This is code of health bar and score counter.

```
if (fireact) {
        firetm++;
        if(firetm>5) {

            cannons.push(new Cannon(pointer.x+(stage.w-pointer.x)/2.5,pointer.y+(stage.h-pointer.y)/2.5,pointer.x,pointer.y));
            cannons.push(new Cannon(pointer.x-(pointer.x)/2.5,pointer.y+(stage.h-pointer.y)/2.5,pointer.x,pointer.y));


            firetm=0;
        }

        arm.x=Math.floor(Math.random()*50)-25+stage.w;
        arm.y=Math.floor(Math.random()*50)-25+stage.h;
        arm2.x=Math.floor(Math.random()*30)-15;
        arm2.y=Math.floor(Math.random()*30)-15+stage.h;
    } else {
        arm.x=stage.w;
        arm.y=stage.h;
        arm2.x=0;
        arm2.y=stage.h;
    }
```

Figure 4: This snippet shows the code to implement the cannon functionality.

```
var Star = function() {
        this.a = Math.random()*Math.PI*2;
        this.v = 3+Math.random()*5;
        this.x = stage.w/2;
        this.y = stage.h/2;
        this.r = 0.2;
}

var Power = function() {
        this.type = Math.floor(Math.random()*2)+1;
        this.a = Math.random()*Math.PI*2;
        this.v = 3+Math.random()*5;
        this.x = stage.w/2;
        this.y = stage.h/2;
        this.r = 0.2;
        this.dis = false;
        this.op = 1;
}


var powers = [];
var powertm = 0;
var powermax = Math.random()*800+300;
// powermax = 10;

var stars = [];

for (var i =0;i<200;i++) {
        stars[i] = new Star();
        var st = stars[i];
        var move = Math.random()*400;
```

Figure 5: This snippet shows the code to implement virus obstacle.

3.2 Milestones

How did you structure the development?

3.2.1 Milestone 1. Our first milestone was to create two guns, each on both the bottom corner's of the screen and we created that in scripts.js file which we called in index.html file. For the free movement or rotation of the gun we used three.js. We used the cannon function and fire function to shoot the bullets from both the guns simultaneously.

3.2.2 Milestone 2. Our second milestone was to create the virus obstacle which would come towards the screen in a loop. If the user would not shoot at the virus obstacle and if the obstacle reaches screen then it would change its color to red and starts decreasing the health.

3.2.3 Milestone 3. Our third milestone was to create a health bar to monitor the health and score counter to monitor the score. Every time the user kills the virus obstacle the score would increase by 10.

3.2.4 Milestone 4. Our fourth milestone was to create a heart obstacle which would help the user increase the health in the game. We also created a bomb obstacle, if the user kills the bomb obstacle then it would kill all the virus obstacles on the screen.

3.2.5 Milestone 5. Our last obstacle was to create the background animation which we did using the Webkit animation function. For the user interface we used the three.js UI functionality.

3.3 Challenges

Describe the challenges you faced.

- Challenge 1: The game was very difficult and challenging to create. The reference that we took from our previous assignments helped us and we also took reference from google and YouTube videos.
- Challenge 2: The main problem we faced was to create the two guns each on the left bottom corner and right bottom corner. Then to fire the bullets we used cannon function and fire function which were very difficult to use as they had lot of calculations and this was very new to us but we learned a lot.
- Challenge 3: Initially our plan was to just create the virus obstacle but then the game would not be interesting to users so to make it more engaging we added health obstacle, bomb obstacle, health bar and score counter. Adding all this objects in a single game was very challenging and we had to learn a lot from YouTube videos and google to make all of them run in connection to each other.
- Challenge 4: The next challenge we faced was when the guns were shooting the bullets at the virus, then the virus should be dead which we didn't know how to do it, so we took references from a couple of other projects. We also had problem in turning the obstacle red when they reached the screen which involved a lot of work.
- Challenge 5: The last challenge we faced was when we created the bomb object, and it would destroy all the other virus objects on the screen. so we had to link the killing of all the virus objects on the screen with the killing of single bomb obstacle which was very difficult and challenging.

4 RESULTS

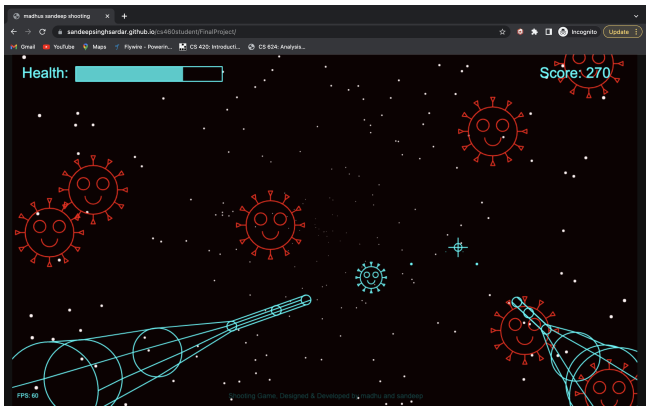

Figure 6: Obstacles turning red when they reach screen.

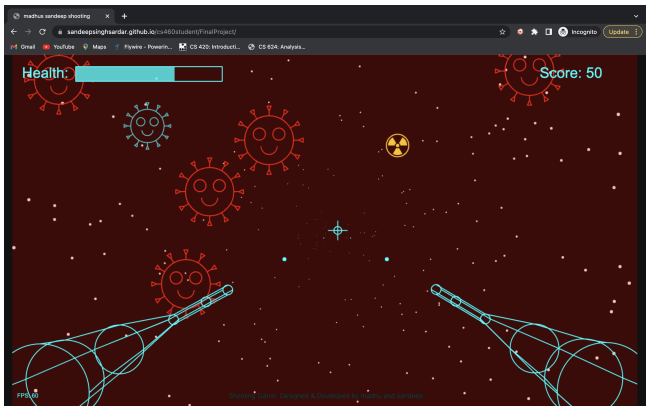

Figure 7: Bomb Obstacle.

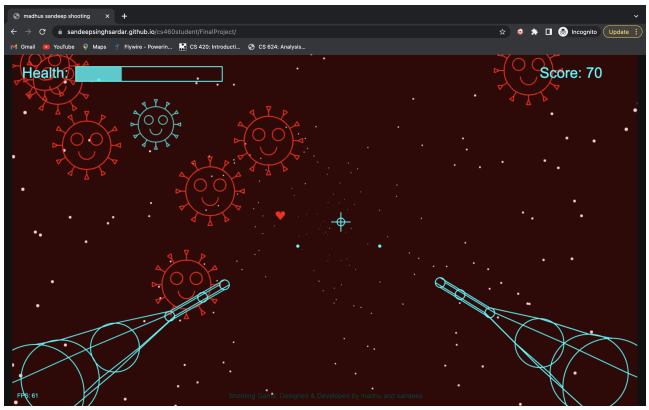

Figure 8: Health Obstacle.

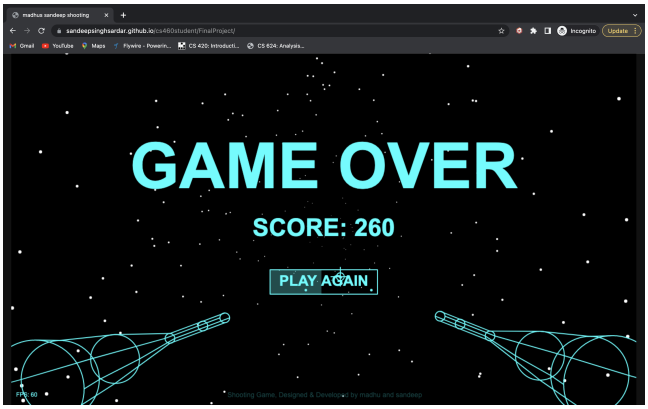

Figure 9: Game Over.

4.1 Our Github pages and Github link for source code are below:

Game link:
https://sandeepsinghsardar.github.io/cs460student/FinalProject/
https://madhumakireddy.github.io/cs460student/FinalProject/

GitHub link: https://github.com/SandeepSinghSardar/cs460student/tree/main/FinalProject
https://github.com/madhumakireddy/cs460student/tree/main/FinalProject

5 CONCLUSIONS

To summarize, making the game was very fun as well as challenging. We learned a lot about many functionalities and objects rendering. Initially we thought that it would be a difficult project but as we created milestones and started achieving them it made further process less complicated. In future we would definitely like to add more features to this game in to make it more appealing to users.

REFERENCES

https://youtube.com/watch?v=X42NGvAtGNk&feature=share
https://threejs.org/
https://www.youtube.com/watch?v=hBiGFpBle7E
https://github.com/verytired/three.js-game