Controlling game through sound input

J Component Project Report for the course

CSE3002 Internet and Web programming

by

Sibi Akkash (18BCE1170)

Prakash Kannan (18BCE1300)

Submitted to

Dr. Sandhya P

School of Computer science and Engineering

VELLORE INSTITUTE OF TECHNOLOGY

CHENNAI - 600127

June 2020

Certificate

This is to certify that the Project work titled “Controlling game through sound input” is being submitted by ***Sibi Akkash*** **(18BCE1170),** ***Prakash kannan*** **(18BCE1300)** for the course **Internet and Web programming**, is a record of bonafide work done under my guidance. The contents of this project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University.

Dr.Sandhya P

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Sandhya P S,** Associate Professor, School of Computer Science and Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.
We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.
We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

SIBI AKKASH **PRAKASH KANNAN**

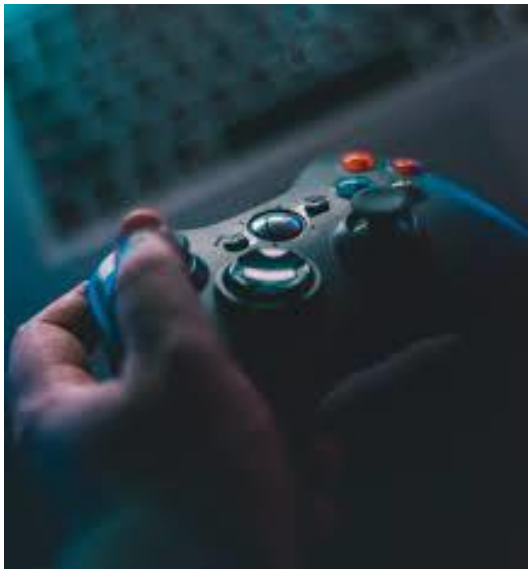
Table of Contents

Problem Statement

The aim of this project is to use sound as input to the game controls. This is achieved by methods of sound classification, which try to understand your voice input and control the game accordingly.

Novelty

Sound classification is an interesting field of machine learning as it combines various fields of ML. Traditionally, games have been controlled with controllers, sticks, keyboard mouse etc… To improve the interactivity and intuitiveness of games, many developments such as motion control, steering wheels for vehicle sims, Virtual reality etc… was introduced. Using sound as an intuitive, hands-free control is new and one that has a lot of scope for improvement. Machine learning helps in this area as work like assistants have already laid the base for this work.



Implementation

A basic chrome dino like game is created using the browser canvas functionality. A sound classification set called MobileNet is used for the sound classification purposes. The animations use a library called *p5.js* and the algorithms part is using a library called *ml5.js*

Player.js

```
class Player {
    constructor() {
        this.x = 50;
        this.height = 50;
        this.y = height - this.height;
        this.width = 50;
        this.velocity = 0;
        this.gravity = 0.7;
        this.thrustVelocity = -11;
    }

    canJump() {
        return this.y == height - this.height - 50;
    }

    show() {
        fill(35, 52, 247);
        rect(this.x, this.y, this.width, this.height);
    }

    update() {
        this.velocity += this.gravity;
        this.y += this.velocity;
        this.y = constrain(this.y, 0, height - this.height - 50);
    }

    jump() {
        if(this.canJump()) {
            this.velocity = this.thrustVelocity;     // upward thrust
        }
    }
}
```

Obstacle.js

```
class Obstacle {
    constructor() {
        this.width = 30;
        this.height = 30;
        this.x = width; // canvas Width
        this.y = height - this.height;
        this.y -= 50;   // to look like its on the ground in the bg image
        this.xspeed = -10; // move towards player
    }

    show() {
        fill(222, 4, 52);
        rect(this.x, this.y, this.width, this.height);
    }

    update() {
        this.x += this.xspeed;
    }

    collides(player) {
    if (
        this.x < player.x + player.width &&
        this.x + this.width > player.x &&
        this.y < player.y + player.height &&
        this.y + this.height > player.y
    ) {
        return true;
    }
    return false;
    }
}
```

Game.js (controls the player and obstacles)

```
// import { Player } from './player.js';
// import { Obstacle } from './obst.js';

class Game {
    constructor() {
        this.playing = false;

        this.obstacles = [];
```

```
        this.classifier;
        // this.options = options;

        this.isModelReady = false;
        this.useSoundInput = true;

        this.refs();

    }

    refs() {
        console.log(this.options);
        this.startRef = document.querySelector('.start-btn');
        this.startScreen = document.querySelector('.start-screen');
        this.endScreen = document.querySelector('.game-end-screen');
        this.scoreDiv = document.querySelector('.score');
        this.highScoreDiv = document.getElementById('highScore');
    }

    loadModel() {
        this.classifier = ml5.soundClassifier('SpeechCommands18w', options
, modelReady);
    }

    init() {
        // this.loadModel();
        this.playing = false;
        this.player = new Player();
        this.obstacles = [];
        this.score = 0;
        this.setup();
        this.showScore();
    }

    start() {
        this.playing = true;
        if(this.playing) {
            this.startScreen.style.display = 'none';
        }
    }

    setup() {
        this.startRef.addEventListener('click', () => this.start());
        this.highscore = localStorage.getItem('highscore');
        if(!this.highscore) {
```

```
            localStorage.setItem('highscore', 0);
            this.highscore = 0;
        }
        if(!this.playing) {
            this.startScreen.style.display = '';
        }

    }

    updateScore() {
        this.score++;
        if(this.score > this.highscore) {
            localStorage.setItem('highscore', this.score);
            this.highscore = this.score;
        }
        this.showScore();
    }

    showScore() {
        this.scoreDiv.textContent = `Score: ${this.score}`;
        this.highScoreDiv.textContent = `HighScore: ${this.highscore}`;
    }
}
```

Sketch.js (starts the game loop)

```
let game, classifier;
let bg;
let bgX1 = 0;
let bgX2;
let options;


function preload() {
    classifier = ml5.soundClassifier('SpeechCommands18w', options, modelRe
ady);
    bg = loadImage('../assets/bg-2.jpg');
}

function setup() {
    createCanvas(800, 400);
    game = new Game();
    options = {
        probabilityThreshold: 0.95,
        scrollSpeed: -5
```

```
    };
    game.init();
    bgX2 = width;
}

function restart() {
    console.log('restatubg');
    game.init();
    loop();
}

function draw() {
    background(0);
    if(game.playing) {
        image(bg, bgX1, 0, width, height);
        image(bg, bgX2, 0, width, height)
        game.player.show();
        game.player.update();
    }

    if(game.obstacles) {
        for(let i = game.obstacles.length - 1; i >= 0 ; i--) {
            game.obstacles[i].show();
            game.obstacles[i].update();

            if(game.obstacles[i].collides(game.player)) {
                console.log('game over');

                game.playing = false;
                noLoop();
                // restart screen
            }

            if(game.obstacles[i].x < 0) {
                game.obstacles.splice(i, 1);
                game.updateScore();
            }
        }
    }

    bgX1 += options.scrollSpeed;
    bgX2 += options.scrollSpeed;

    if(bgX1 <= - width) bgX1 = width;
    if(bgX2 <= - width) bgX2 = width;
}
```

```
// keyboard input
function keyPressed() {
    if(keyCode === 32) {
        console.log('jump');
        game.player.jump();
    }
    if(keyCode === UP_ARROW) {
        game.obstacles.push(new Obstacle());
    }
}

//model
function modelReady() {
    isModelReady = true;
    console.log('model ready');
    classifier.classify(gotResult);
}

function gotResult(error, result) {
    if(error) {
        console.error(error)
        return;
    }
    console.log(`${result[0].label}, confidence: ${result[0].confidence}`)
;

    if(result[0].label === 'up' ||  result[0].label === 'left') {
        player.jump();
    }
}
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <title>Document</title>
    <link rel="stylesheet" href="./css/index.css" />
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.8.0/p5.js"
></script>
    <script src="https://unpkg.com/ml5@0.4.3/dist/ml5.min.js"></script>
```

```
    <script src="sketch.js" ></script>
    <script src="player.js" ></script>
    <script src="obst.js" ></script>
    <script src='game.js'></script>
  </head>
  <body>
  <button class='reset' onclick='restart()'>Restart</button>
  <div class='start-screen'>
    <div>Start Game</div>
    <img src="../assets/start.png" alt="Start button" class='start-btn'>
  </div>
  <div class='game-end-screen'></div>
  <div class='scoreContainer'>
    <div class='score' id='score'></div>
    <div class='score' id='highScore'></div>
  </div>

  </body>
</html>
```

Style.css

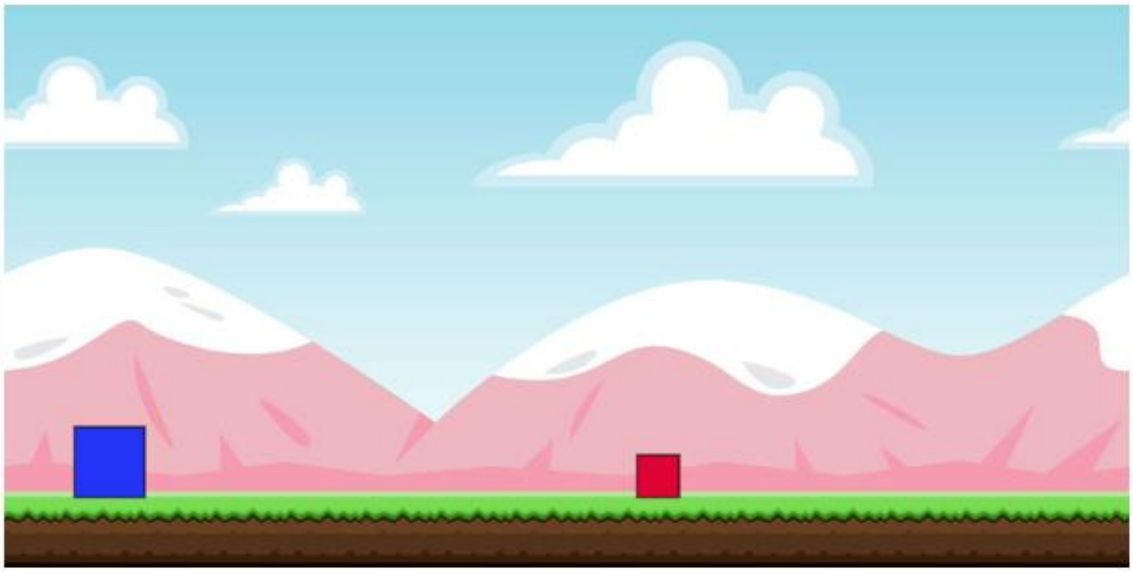
```
.reset {
    border: none;
    text-decoration: none;
    padding: 10px 10px;
    font-size: 20px;
}

.score {
    font-size: 30px;
}

canvas {
    margin: 0 auto
}

.scoreContainer {
    display: flex-end;
    padding: 10px;
}

.start-screen {
    display: flex;
    justify-content: center;
```

```
    align-items: center;
    position: absolute;
    top: 139px;
    background-color: rgba(247, 31, 229);
    z-index: 2;
    width: 802px;
    height: 402px;
}

.start-btn {
    display: block;
    mix-blend-mode: multiply;
    width:70px;
}

.start-btn:hover {
    cursor: progress;
}

.start-screen > div {
    /* border: 2px solid black; */
    display: block;
    font-family: monospace;
    font-size: 50px;
    padding: 20px;
}
```

Output

Restart

Score: 0
HighScore: 59



Restart

Score: 1
HighScore: 59