Universidad Tecnológica de la Mixteca

Agentes: Equipo de Fútbol

Integrantes:

1. Martínez Guzmán Erika Michelle
2. Payán Rosales Carlos Antonio
3. Ramírez González Metztli
4. Vásquez Tejada Victor Antonio

Carrera: Ingeniería en computación

Materia: Inteligencia Artificial

Grupo: 802-A

A viernes 21 de mayo de 2021

Índice

Lista de figuras

Introducción

El presente reporte explica la realización de la práctica de Inteligencia Artificial sobre un agentes de un equipo de fútbol; contiene la explicación del funcionamiento que tienen los agentes que forman cada equipo y el código fuente elaborado para estos, el cual fue realizado en el lenguaje de programación JavaScript y se usó la librería Crafty. Además se incluyen imágenes de la interfaz resultante en acción y las conclusiones obtenidas sobre el trabajo realizado. También exponen conceptos claves necesarios para entender mejor la práctica.

Funcionamiento

Sabemos que un agente percibe su ambiente y se encuentra dentro de un entorno de trabajo que está formado por el ambiente. los actuadores y los sensores del propio agente. En esta práctica se simula el comportamiento de un entorno multiagente, donde funcionan dos conjuntos de agentes jugadores que representan a dos equipos de fútbol enfrentándose en un partido, cada equipo tiene 7 jugadores. Este encuentro tiene un tiempo límite de 10 minutos y se pueden anotar tantos goles como los agentes puedan meter.

Cada agente jugador se mueve a una velocidad diferente en el eje x y y, para poder desplazarse por la cancha. Los defensas tiene una restricción que no pueden pasar más allá de la mitad de la cancha correspondiente a su equipo y los porteros no pueden ir más allá del rectángulo menor. Por otro lado tenemos al balón tiene un movimiento aleatorio, que cambia su dirección en sentido contrario cada vez que es tocado por un agente jugador.

El ambiente está creado por un una cancha de fútbol , por donde se podrá desplazar el agente. En un inicio los jugadores de ambos equipos tienen una

posición determinada repartidos en toda la cancha y el balón se ubica en una posición aleatoria, una vez se inicia el programa los jugadores empiezan a moverse y el balón es interceptado por estos tantas veces como lo encuentren.

Los agentes juegan durante 10 minutos sin limite de goles y gana el que al final tenga más goles.

Código Fuente

La implementación del proyecto se llevó a cabo con el lenguaje JavaScript. El código se compone de el head y el body, en la primera parte se declara el estilo que tomará el campo y cada botón para la funcionalidades de start, pausa y stop. El body se compone primeramente de la definición de las funcionalidades de los botones mencionados anteriormente, seguidamente se ubican los botones de lado derecho de la pantalla, mientras que el timer y el marcador de goles se ubican en la parte superior. Después encontramos la inicialización del balón, la creación de las porterías y los jugadores, donde se define el espacio que le corresponde a cada uno para que en funciones posteriores se pueda detectar en qué parte se encuentra el balón y cada jugador, y así tomar la funcionalidad que debe.

Crafty fue la colección de librerías de Javascript que se utilizó para manejar el comportamiento representado por cada jugador (agente). Crafty fue diseñado para desarrollar videojuegos fácilmente en gráficos 2D, cuenta con un entorno de trabajo para construir juegos basados en el navegador con tecnología web estándar (HTML5). Los dos elementos estructurales más importantes de un juego en Crafty son los *componentes* y las *entidades*. Una *entidad* es cualquier objeto que existe en el tu juego y tiene un comportamiento de cualquier tipo dentro de él, por otro lado,

un *componente* es el que especifica el grupo de datos y/o comportamientos que pueden ser aplicados dichas entidades.

Las siguientes imágenes muestran el código que compone el juego desarrollado.

```
<html>
  <head>
    <style>
        .campo{
            background-image:url("https://i.pinimg.com/originals/e9/66
/7f/e9667f36f88477226cf6e485d4f76eca.jpg");
            background-size: cover;
            width:1024px;
            height: 640px;
            float: left;
            box-shadow: 10px 10px ;
            border: 2px solid;
            border-color: black;
            border-radius: 50px;
        }
        .pausa{
            width: 70px;
            height: 36px;
            line-height: 2.25rem;
            float: right;
            margin-right:150px;
            margin-top:20px;
            padding: 0 10px 0 18px;
            border-radius: 50px;
            color: #eee;
            font-family: Roboto, sans-serif;
            font-size: 0.875rem;
            font-weight: 510;
            letter-spacing: 0.06em;
            text-transform: uppercase;
            border: 2px solid;
            border-color:yellow;
            cursor: pointer;
        }
        .pausa:hover,
        .pausa:focus {
            box-shadow: 0 0.5em 0.5em -0.4em var(--hover);
            -webkit-transform: translateY(-0.25em);
            transform: translateY(-0.25em);
            border-bottom: 5px solid;
            border-color:yellow;
        }
```

Figura 1: HeadPt1

```
        .start{
            width: 70px;
            height: 36px;
            line-height: 2.25rem;
            float: right;
            margin-right:150px;
            margin-top:100px;
            padding: 0 5px 0 20px;
            border-radius: 50px;
            color: #eee;
            font-family: Roboto, sans-serif;
            font-size: 0.875rem;
            font-weight: 510;
            letter-spacing: 0.06em;
            text-transform: uppercase;
            border: 2px solid;
            border-color:yellow;
            cursor: pointer;
        }
        .start:hover,
        .start:focus {
            box-shadow: 0 0.5em 0.5em -0.4em var(--hover);
            -webkit-transform: translateY(-0.25em);
            transform: translateY(-0.25em);
            border-bottom: 5px solid;
            border-color:yellow;
        }
        .stop{
            width: 70px;
            height: 36px;
            line-height: 2.25rem;
            float: right;
            margin-right:150px;
            margin-top:20px;
            padding: 0 5px 0 20px;
            border-radius: 50px;
            color: #eee;
            font-family: Roboto, sans-serif;
            font-size: 0.875rem;
            font-weight: 510;
            letter-spacing: 0.06em;
            text-transform: uppercase;
            border: 2px solid;
            border-color:yellow;
            cursor: pointer;
        }
        .stop:hover,
        .stop:focus {
            box-shadow: 0 0.5em 0.5em -0.4em var(--hover);
            -webkit-transform: translateY(-0.25em);
            transform: translateY(-0.25em);
            border-bottom: 5px solid;
            border-color:yellow;
        }
        body{
            background: linear-gradient(#e66465, #9198e5);
        }
    </style>
</head>
```

Figura 2: HeadPt2

```
<body>
    <div style="display:float;">
        <div class="campo"><div id="game" ></div></div>

        //Funcionalidad de los botones
        <div class="start" onclick="initPosition();Crafty.init();ball.attr({x: 494, y: 302, w: 20, h:
          20});">Start</div>
        <div id="pausa" class="pausa" onclick="Crafty.pause();changeBoton();">Pause</div>
        <div class="stop" onclick="time=0; initPosition();Crafty.stop();go1=0;go2=0;marcador.text('Team
          1: '+go1+' vs Team 2: '+go2);">Stop</div>
    </div>
    <script type="text/javascript" src="https://rawgithub.com/craftyjs/Crafty/release/dist/crafty-
        min.js"></script>
    <script>

    changeBoton=()=>{
        if(document.getElementById('pausa').innerHTML == 'Pause'){
            document.getElementById('pausa').innerHTML= 'Resume';
        }else{document.getElementById('pausa').innerHTML = 'Pause';}
    }

    //Inicialización del campo
      Crafty.init(1024,640, document.getElementById('game'));

    //TEXTO Y BOTONES
    //Ubicacion del timer
    var time=0
    var Tiempo = Crafty.e('2D, DOM, Text')
      .attr({
        x: 60,
        y: 13,
        w:265,
        h:50
      });

    //Formato del tiempo
    Tiempo.textFont({
      size: '20px',
    }).textColor('white');

    //Tiempo en formato mm:ss
    Tiempo.bind("UpdateFrame", function(eventData){
        time+=eventData.dt/1000;
        var minutos=parseInt((time/60),10);
        if(minutos<10)minutos='0'+minutos;
        var segundos=parseInt((time%60),10);
        if(segundos<10)segundos='0'+segundos;
        if(minutos==10){
            Crafty.stop();go1=0;go2=0;time=0;
            this.text('Se acabo el partido');
        }else{
            this.text('Tiempo: '+minutos+':'+segundos);
        }
    })

    //Ubicacion del marcador
    var marcador = Crafty.e('2D, DOM, Text')
      .attr({
        x: 380,
        y: 10,
        w:265,
        h:50
      });

    //Formato marcador de goles
    marcador.textFont({
        size: '25px',
      }).textColor('white');
      var go1=0;
      var go2=0;
      marcador.text('Team 1: '+go1+' vs Team 2: '+go2);
```

Figura 3: Body

La figura 4 muestra la definición de los jugadores, asignación de su tamaño e imagen, así como la inicialización del balón, también con su tamaño y representación.

```
//INICIALIZACIÓN DE LOS JUGADORES
//equipo1
    var assetsObj = {
    "sprites": {
        "http://i.imgur.com/e50rhHP.png": {
            tile: 104,
            tileh: 114,
            map: {
                walker_start: [0, 0],
                walker_middle: [7, 0],
                walker_end: [7, 1]
            }
        }
    }
};
Crafty.load(assetsObj);

//equipo2
    var assetsObj2 = {
    "sprites": {
        "image.png": {
            tile: 104,
            tileh: 114,
            map: {
                walker2_start: [0, 0],
                walker2_middle: [7, 0],
                walker2_end: [7, 1]
            }
        }
    }
};
Crafty.load(assetsObj2);

//INICIALIZACIÓN DEL BALÓN
    var assetsObjBall = {
    "sprites": {
        "https://i.imgur.com/VXse9yk.png": {
            tile: 306,
            tileh: 306,
            map: {
                ball_start: [0, 0],
                ball_middle: [0, 0],
                ball_end: [0, 0]
            }
        }
    }
};
Crafty.load(assetsObjBall);
```

Figura 4: Inicialización de jugadores y balón

La creación de las porterías y del balón se realizó por medio de la librería Crafty, esto para poder detectar el movimiento por medio de sus coordenadas y el método collision(). Si se detectaba el balón dentro de la portería, desde aquí mismo se actualizaba el marcador, tal como se muestra en la siguiente imagen.

```
//CREACIÓN DE LAS PORTERIAS
//Porteria equipo1
var red1 = Crafty
                .e("2D, Canvas, Color, Collision")
                .attr({x: 28, y: 280, w: 28, h: 82})
                .color('green')
                .collision()
                .checkHits("ball_start")
                .bind("HitOn", function(hitData) {
                    if ((hitDatas = this.hit('ball_start'))) {
                        go2+=1;
                        Crafty.stop();
                        initPosition();
                        setTimeout(function(){ Crafty.init(); }, 1000);
                        marcador.text('Team 1: '+go1+' vs Team 2: '+go2);
                    }
                });
//Porteria equipo2
var red2 = Crafty
                .e("2D, Canvas, Color, Collision")
                .attr({x: 969, y: 280, w: 28, h: 82})
                .color('green')
                .collision()
                .checkHits("ball_start")
                .bind("HitOn", function(hitData) {
                    if ((hitDatas = this.hit('ball_start'))) {
                        go1+=1;
                        Crafty.stop();
                        initPosition();
                        setTimeout(function(){ Crafty.init(); }, 1000);
                        marcador.text('Team 1: '+go1+' vs Team 2: '+go2);
                    }
                });
```

Figura 5: Creación de porterías

Dentro de la creación de balón se definieron los tipos de movimientos que podía tomar los cuales son hacia arriba, abajo y central, estos dependen de donde era tocado el balón y por cual equipo.

```
//CREACIÓN DEL BALÓN
var ball = Crafty
                .e('2D, Canvas, ball_start, SpriteAnimation, Collision')
                .attr({x: 494, y: 302, w: 20, h: 20})
                .collision()
                .checkHits('Collision')
                .bind("HitOn", function(hitData) {
                    //Desplazamientos
                    if ((hitDatas = this.hit('arriba'))) {
                        this.y_move=false;
                        this.y_tiro=false;
                    }
                    if ((hitDatas = this.hit('abajo'))) {
                        this.y_move=true;
                        this.y_tiro=false;
                    }
                    if ((hitDatas = this.hit('centro'))) {
                        this.y_tiro=true;
                    }
                    if ((hitDatas = this.hit('walker_start'))) {
                        this.x_move=true;
                    }
                    if ((hitDatas = this.hit('walker2_start'))) {
                        this.x_move=false;
                    }
                });
ball.reel("walking", 500, [
    [0, 0]
]);
ball.animate("walking", -1);
```

Figura 6: Creación del balón.

La siguiente imagen muestra la creación e inicialización de los jugadores, además de que se define el espacio por donde podrá desplazarse cada jugador.

```
//CREACIÓN DE LOS JUGADORES

var team1 = [];
var team2 = [];
//Equipo1
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, centro').attr({x: 48, y:
270, w: 50, h: 50}).collision());
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, arriba').attr({x: 200, y:
190, w: 50, h: 50}).collision());
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, abajo').attr({x: 200, y:
350, w: 50, h: 50}).collision());
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, centro').attr({x: 310, y:
270, w: 50, h: 50}).collision());
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, arriba').attr({x: 310, y:
90, w: 50, h: 50}).collision());
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, abajo').attr({x: 310, y:
450, w: 50, h: 50}).collision());
team1.push(Crafty.e('2D, Canvas, walker_start, SpriteAnimation, Collision, centro').attr({x: 400, y:
270, w: 50, h: 50}).collision());

//equipo2
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, centro').attr({x: 904, y:
270, w: 50, h: 50}).collision());
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, arriba').attr({x: 750, y:
190, w: 50, h: 50}).collision());
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, abajo').attr({x: 750, y:
350, w: 50, h: 50}).collision());
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, centro').attr({x: 645, y:
270, w: 50, h: 50}).collision());
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, arriba').attr({x: 645, y:
90, w: 50, h: 50}).collision());
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, abajo').attr({x: 645, y:
450, w: 50, h: 50}).collision());
team2.push(Crafty.e('2D, Canvas, walker2_start, SpriteAnimation, Collision, centro').attr({x: 545, y:
270, w: 50, h: 50}).collision());

//Posicionamiento de los jugadores
//equipo1
var initPosition = ()=>{
team1[0].attr({x: 48, y: 270, w: 50, h: 50});
team1[1].attr({x: 200, y: 190, w: 50, h: 50});
team1[2].attr({x: 200, y: 350, w: 50, h: 50});
team1[3].attr({x: 310, y: 270, w: 50, h: 50});
team1[4].attr({x: 310, y: 90, w: 50, h: 50});
team1[5].attr({x: 310, y: 450, w: 50, h: 50});
team1[6].attr({x: 400, y: 270, w: 50, h: 50});

//equipo2
team2[0].attr({x: 904, y: 270, w: 50, h: 50});
team2[1].attr({x: 750, y: 190, w: 50, h: 50});
team2[2].attr({x: 750, y: 350, w: 50, h: 50});
team2[3].attr({x: 645, y: 270, w: 50, h: 50});
team2[4].attr({x: 645, y: 90, w: 50, h: 50});
team2[5].attr({x: 645, y: 450, w: 50, h: 50});
team2[6].attr({x: 545, y: 270, w: 50, h: 50});
}

//Espacio de desplazamiento establecido para cada jugador
for(var i=0;i<7;i++){
    //espacio jugador equipo1
    team1[i].reel("walking", 500, [
                [0, 0], [1, 0], [2, 0], [3, 0], [4, 0], [5, 0], [6, 0], [7, 0],
                [0, 1], [1, 1], [2, 1], [3, 1], [4, 1], [5, 1], [6, 1], [7, 1]
            ]);
    team1[i].animate("walking", -1);

    //espacio jugador equipo2
    team2[i].reel("walking", 500, [
                [0, 0], [1, 0], [2, 0], [3, 0], [4, 0], [5, 0], [6, 0], [7, 0],
                [0, 1], [1, 1], [2, 1], [3, 1], [4, 1], [5, 1], [6, 1], [7, 1]
            ]);
    team2[i].animate("walking", -1);
}
```

Figura 7: Creación de los jugadores

Manejo del movimiento del balón, inicialmente comienza en el centro del campo de otro modo se calcula el ángulo dependiendo donde lo golpearon, aquí también se define la rapidez que tomará conforme al jugador por el cual fue pateado.

```
//Movimiento del balón
let xball;
let yball;
ball.origin("center");
ball.bind("UpdateFrame", function(eventData){
    if(this.x > 950 || !this.x_move){
        xball=true;
        this.x_move=false;
    }
    if(this.x < 55 || this.x_move){
        xball=false;
        this.x_move=true;
    }
    if(xball){
        this.x = this.x - 222 * (eventData.dt / 1000);
        this.rotation = this.rotation - 8;
    }
    else{
        this.x = this.x + 222 * (eventData.dt / 1000);
        this.rotation = this.rotation + 8;
    }

    if(this.y >= 595 || this.y_move){
        yball=true;
        this.y_move=true;
    }
    if(this.y <= 17 || !this.y_move){
        yball=false;
        this.y_move=false;
    }
    let velocidad=200;
    if(this.y_tiro){
        velocidad=22;
    }
    if(yball){
        this.y = this.y - velocidad * (eventData.dt / 1000);}
    else{
        this.y = this.y + velocidad * (eventData.dt / 1000);}
});
```

Figura 8: Movimiento del balón

El movimiento de los jugadores se detectó por medio del método eventData. Los jugadores se desplazan con movimientos aleatorios por todo tu espacio establecido anteriormente. La figura 9 muestra el tratamiento de dos jugadores, los demás se manejaron de la misma manera con sus respectivos espacios, para no alargar el reporte solo se incluye el siguiente ejemplo:

```
//Movimiento de los jugadores
    //Equipo 1
    let y1;
    let x1;
team1[0].bind("UpdateFrame", function(eventData) {
    if(this.x > 68)
        x1=true;
    if(this.x < 28)
        x1=false;
    if(x1)
        this.x = this.x - 30 * (eventData.dt / 1000);
    else
        this.x = this.x + 30 * (eventData.dt / 1000);

    if(this.y >= 350){
        y1=true;}
    if(this.y <= 190){
        y1=false;}
    if(y1){
        this.y = this.y - 68 * (eventData.dt / 1000);}
    else{
        this.y = this.y + 73 * (eventData.dt / 1000);}
  });
    let y2;
    let x2;
  team1[1].bind("UpdateFrame", function(eventData) {
    if(this.x > 240)
        x2=true;
    if(this.x < 140)
        x2=false;
    if(x2)
        this.x = this.x - 40 * (eventData.dt / 1000);
    else
        this.x = this.x + 40 * (eventData.dt / 1000);

    if(this.y >= 270){
        y2=true;}
    if(this.y <= 2){
        y2=false;}
    if(y2){
        this.y = this.y - 75 * (eventData.dt / 1000);}
    else{
        this.y = this.y + 60 * (eventData.dt / 1000);}
  });
```

Figura 9: Movimiento de los jugadores

Pruebas de ejecución

A continuación presentamos las pantallas que corresponden a la ejecución del programa elaborado para la práctica.

La figura 10 muestra el menú que controla el juego. El botón “Start” sirve para iniciar el juego e inicia la ejecución del programa. El botón “Pause” se utiliza para pausar el juego, todos los agentes se quedan quietos en la misma posición, entonces el botón cambia a “Resume” y este se presiona cuando se quiere volver a reanudar el juego. Por último el botón “Stop” para detener el juego por completo y terminarlo, este estado no se recupera y entonces solo se puede dar “start” para comenzar un juego nuevo.

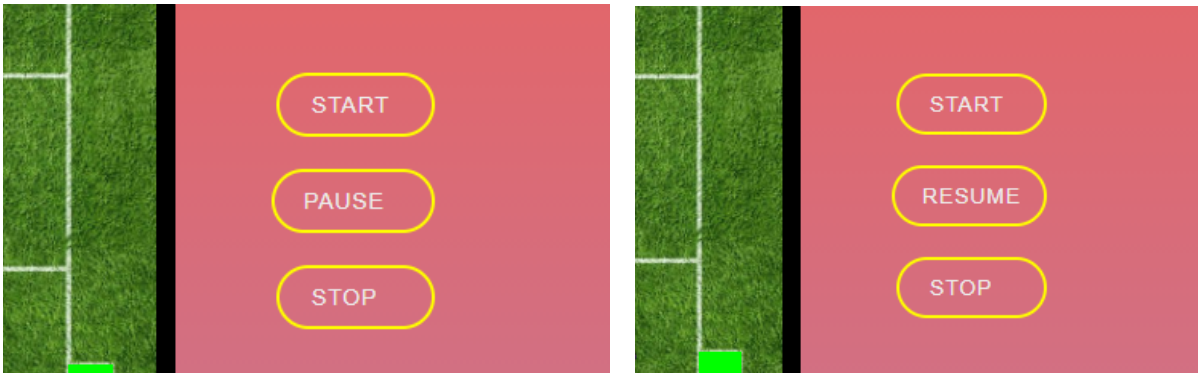

Figura 10: Menú principal

La figura 11 muestra la pantalla del juego completo. En la esquina superior izquierda se encuentra el timer que va contando cuando tiempo de partido lleva el juego, el máximo es 10 minutos y entonces termina el juego. Se ven los agentes del “Team 1” que son color negro y los agentes del “Team 2” que son de color rosa. También vemos la pelota cerca de la portería del Team 1.

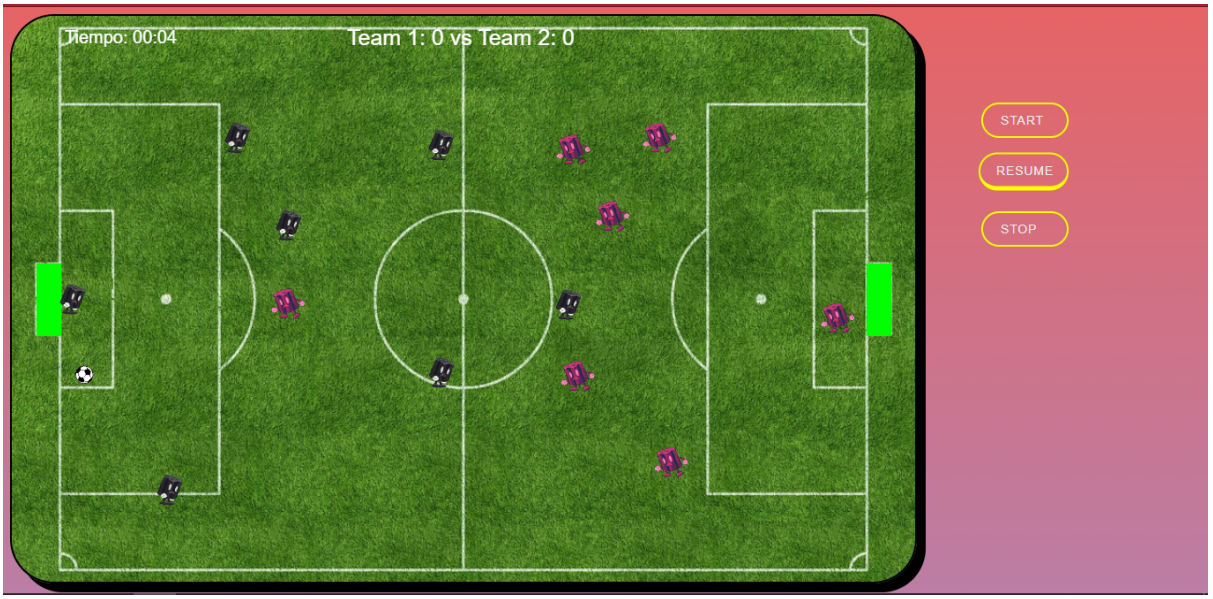

Figura 11: Agentes en acción

En la figura 12 el juego fue detenido con el botón “Stop”, entonces todos los agentes se quedan detenidos donde estén. Para un juego nuevo se debe dar en “Start”.

Figura 12: Juego en Stop

En la figura 13 se aprecia como es un juego pausado, entonces el botón “Resume” se usa para reanudar el juego. Podemos ver que el Team 2 ya ha anotado un gol en los 19 segundos de juego que llevan.

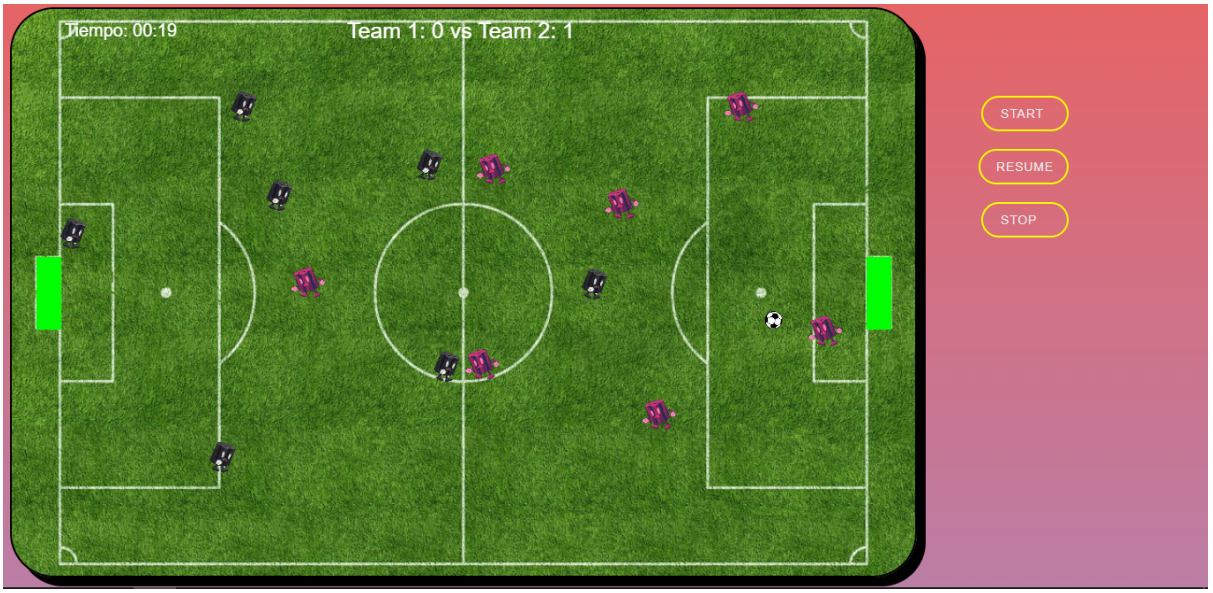

Figura 13: Juego en pausa

En la figura 14 el juego ha avanzado y ya lleva 6 minutos y 39 segundos, en los cuales el team 1 ha anotado 4 goles y el team 2 ha anotado 6 goles.

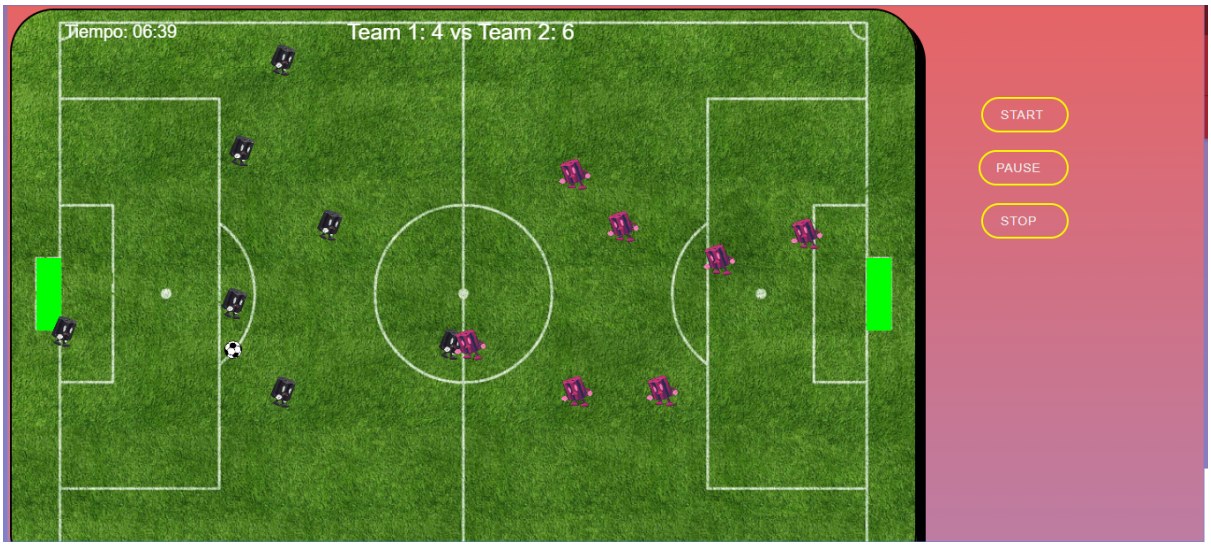

Figura 14: Juego en acción

En la figura 15 el juego ya lleva 7 minutos y 59 segundos, en los cuales el team 1 ha anotado 4 goles y el team 2 ha anotado 7 goles.

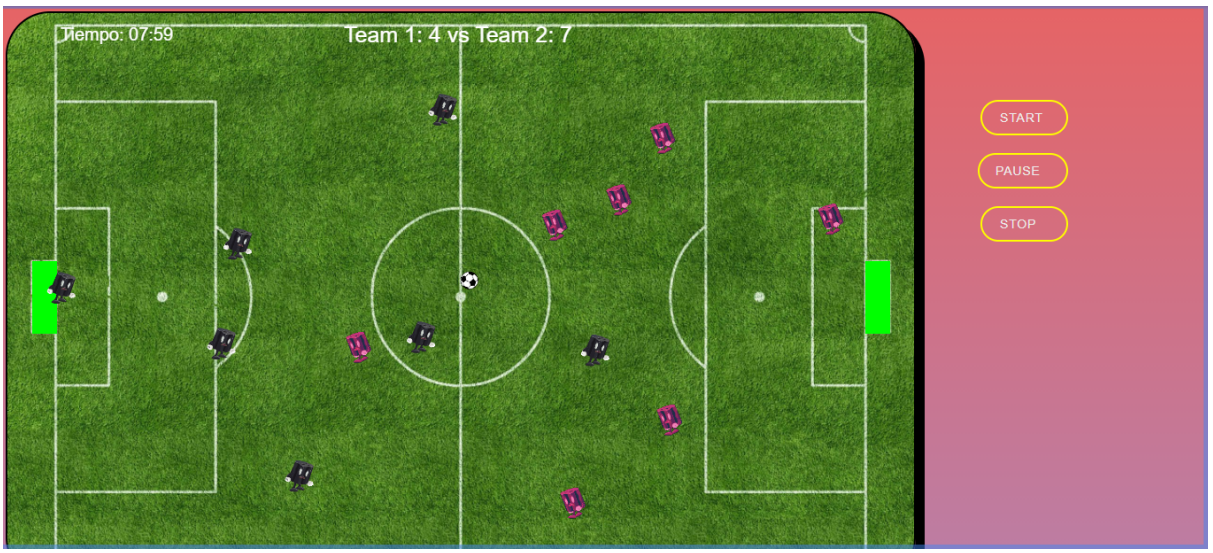

Figura 15: Juego en acción

Como se ve en la figura 16, el juego ha alcanzado el tiempo límite de 10 minutos y entonces pasa a estado finalizado, como se puede leer en el mensaje en la esquina superior izquierda “Se acabó el partido”. Vemos que ganó el team 2 pues anotó 8 goles y el team 1 anotó 7 goles. Para iniciar un partido nuevo se debe presionar el botón “Start”.

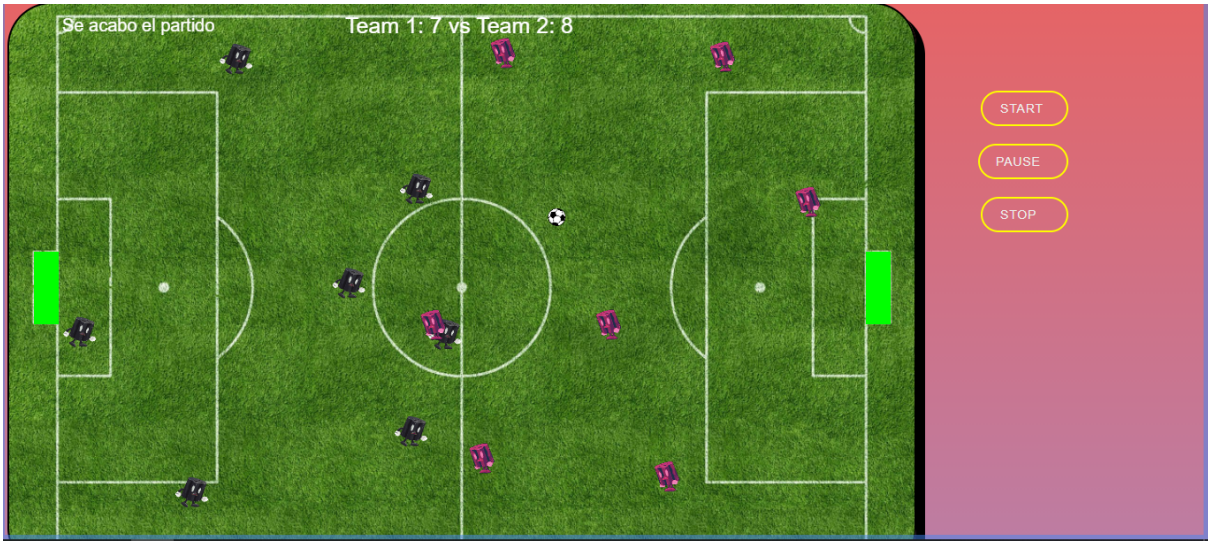

Figura 16: Juego finalizado

Conclusiones

La realización de esta práctica resultó interesante y nos permitió reafirmar los conceptos de un entorno multiagente agente y cómo se relacionan, los cuales son indispensables entender en esta área de inteligencia artificial. Además comprendimos como es el funcionamiento de un producto que en estos tiempos ha ido abriéndose mercado y puede aplicarse en muchos problemas y sus soluciones.

En conclusión, como computologos simular este tipo de sistemas nos permite aprender y entender de mejor manera los comportamientos de sistemas inteligentes para poder desarrollarlos en un futuro.