TC2017.1 Electricidad y Magnetismo

Proyecto Final

Profesor Edgar René

Pablo Muñoz Haro A01222422
Andrés Barro

November 9, 2016

Contents

1 Manifest Constants

```
var WINDOW_WIDTH  = 970;
var WINDOW_HEIGHT = 720;

const PERMITIVITY     = 9 * Math.pow(10, 9);
const ELECTRON_CHARGE = -1.602 * Math.pow(10, -19);
const PROTON_CHARGE   = -ELECTRON_CHARGE;
const PROTON_MASS     = 1.6727 * Math.pow(10, -27);
const NEUTRON_MASS    = 1.6750 * Math.pow(10, -27);
const ELECTRON_MASS   =  9.110 * Math.pow(10, -31);
```

2 Partículas

2.1 class Particle

```
class Particle {
    // x, y, radius
    constructor(args) {
        _.assign(this, _.defaults(args, {
            x: WINDOW_WIDTH / 2,
            y: WINDOW_HEIGHT / 2,
            radius: 8,
            velocityX: 0,            // m/s
            velocityY: 0,            // m/s
            accelX: 0,               // m/s
            accelY: 0,               // m/s
            charge: ELECTRON_CHARGE, // C
            mass: ELECTRON_MASS      // kg
        }));

        if (this.charge > 0) {
            this.color = 'red';
        } else if (this.charge < 0) {
            this.color = 'blue';
        }
    }

    draw() {
        this.circle = new Path.Circle(new Point(this.x, this.y), this.radius);
        this.circle.fillColor = this.color;
    }

    on(event, func) {
        this.circle['on' + _.capitalize(event)] = func;
    }
```

```
31
32      reactToElectricFieldDueTo(otherParticle) {
33          const distanceX = (this.circle.position.x - otherParticle.circle.position.x); // 1 pxl = 1cm
34          const distanceY = (this.circle.position.y - otherParticle.circle.position.y); // 1 pyl = 1cm
35          var forceX = 0,
36              forceY = 0;
37
38          if (distanceX != 0) {
39              forceX = PERMITIVITY * (Math.abs((this.charge * otherParticle.charge))
40                                      / Math.pow(distanceX, 2));
41          }
42          if (distanceY != 0) {
43              forceY = PERMITIVITY * (Math.abs((this.charge * otherParticle.charge))
44                                      / Math.pow(distanceY, 2));
45          }
46
47          this.accelX = forceX / this.mass;
48          this.accelY = forceY / this.mass;
49
50          // Still need to come up with proper sign
51          const sameChargeType = Math.sign(this.charge) == Math.sign(otherParticle.charge);
52
53          if (sameChargeType) {
54              // Repel each other
55              if (otherParticle.circle.position.x > this.circle.position.x) {
56                  this.accelX *= -1;
57              }
58              if (otherParticle.circle.position.y > this.circle.position.y) {
59                  this.accely *= -1;
60              }
61          } else {
62              if (otherParticle.circle.position.x < this.circle.position.x) {
63                  this.accelX *= -1;
64              }
65              if (otherParticle.circle.position.y < this.circle.position.y) {
66                  this.accely *= -1;
67              }
68          }
69      }
70
71      advanceTime(milliseconds) {
72          const seconds = milliseconds / 1000;
73          this.velocityX += this.accelX * seconds;
74          this.velocityY += this.accelY * seconds;
75          this.circle.translate(new Point(this.velocityX, this.velocityY));
76      }
77
78  }
```

3 Funciones de utilería

4 Simulación de Ley de Coulomb

Para las simulaciones de la ley de Coulomb se realizan las siguientes suposiciones:

- Las partículos se encuentran inicialmente en reposo (velocidad = acelearación = 0)
- Cada pixel equivaldrá a 1m del mundo real.

4.1 Interacción entre 2 cargas puntuales

```
class TwoPointChargeSystem {
    constructor() {
        this.frameMillis = 1000 / 60;

        this.p0 = new Particle({
            x: WINDOW_WIDTH / 2 - 30,
        });
        this.p0.draw();

        this.p1 = new Particle({
            x: WINDOW_WIDTH / 2 + 30,
        });
        this.p1.draw();
    }

    start() {
        this.refreshIntervalId = setInterval(_.bind(function() {
            this.p0.reactToElectricFieldDueTo(this.p1);
            this.p1.reactToElectricFieldDueTo(this.p0);
            this.p0.advanceTime(this.frameMillis);
            this.p1.advanceTime(this.frameMillis);
        }, this), this.frameMillis);
    }

    stop() {
    }
}
```