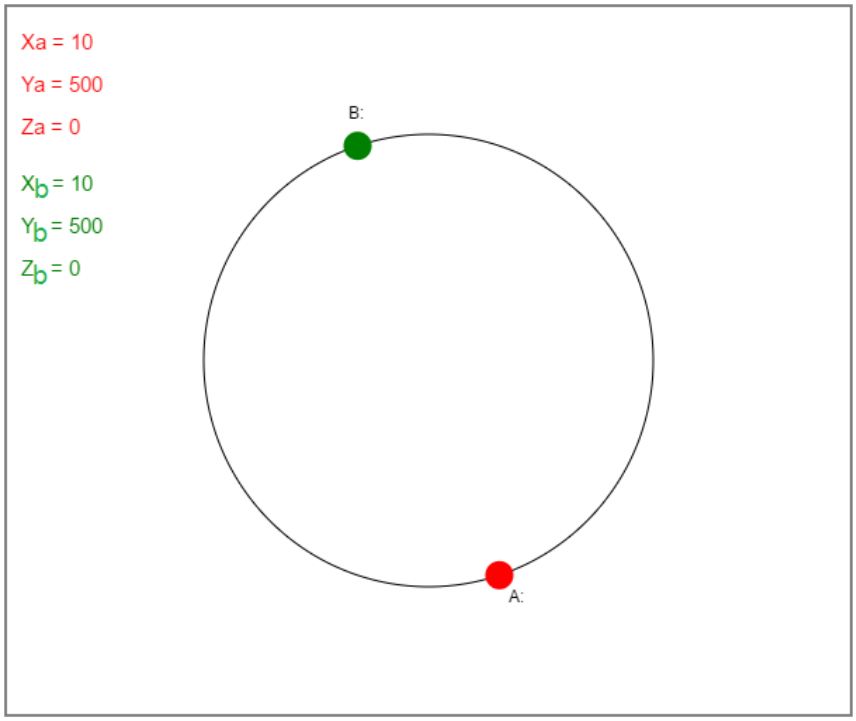
Contemporary Data Processing Technology (CCOD)

Lab 1 (September 09, 2016)

Savchuk Artem, AS-36

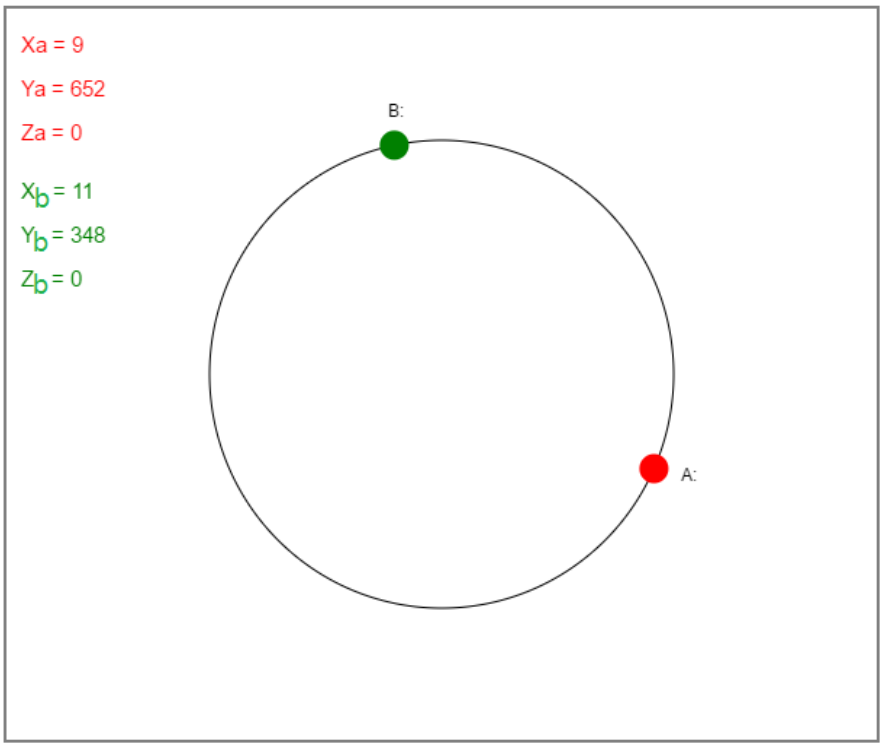
Controll two metro cars using Speed, Distance, and Brake. In this work we are design a virtual loop with 1000 pixels on which two metro cars(A – first train car and B – second train car). We don't assume any stations. That is, both train always run. The speed of train is denoted as X_a and X_b. Speed ranges from 0 to 20 pixels per unit of time. The distance from train A to B is denoted as Y_a and from train B to train A is Y_b, so $Y_a + Y_b = 1000$. Both train move in a clockwise direction. At the beginning speed of trains is 10 pixels per unit of time ($X_a = 10$, $X_b = 10$) and $Y_a = 500$, $Y_b = 500$; Step by step the speed changes by -1, 0, +1 randomly. Simulations stops when train collide. We also need to calculate the value membership function for speed.

I wrote a program that simulates this virtual metro. The result of the program shown at the bottom.



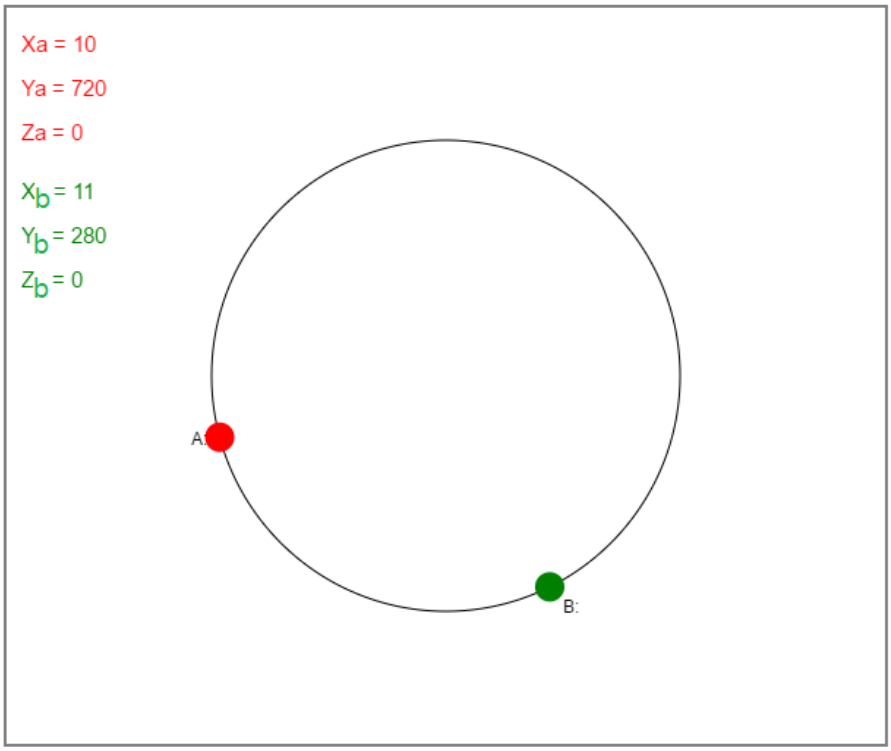
A: Membership function for speed:
very slow: **0**
slow: **0**
medium: **1**
fast: **0**
very fast: **0**

B: Membership function for speed:
very slow: **0**
slow: **0**
medium: **1**
fast: **0**
very fast: **0**



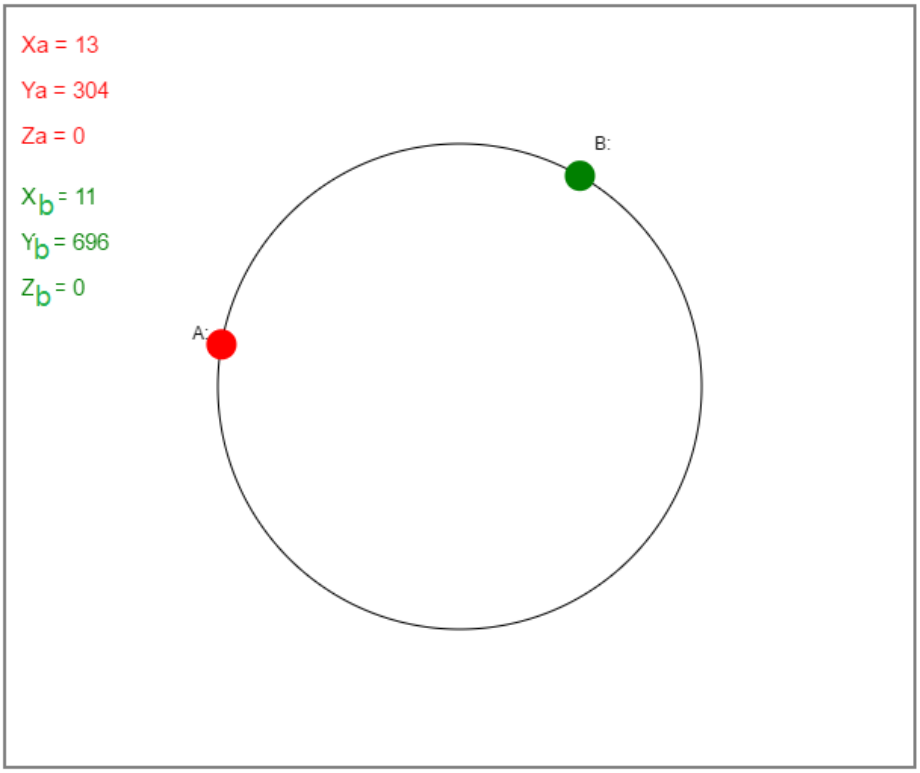
A: Membership function for speed:
very slow: **0**
slow: **0.25**
medium: **0.75**
fast: **0**
very fast: **0**

B: Membership function for speed:
very slow: **0**
slow: **0**
medium: **0.75**
fast: **0.25**
very fast: **0**



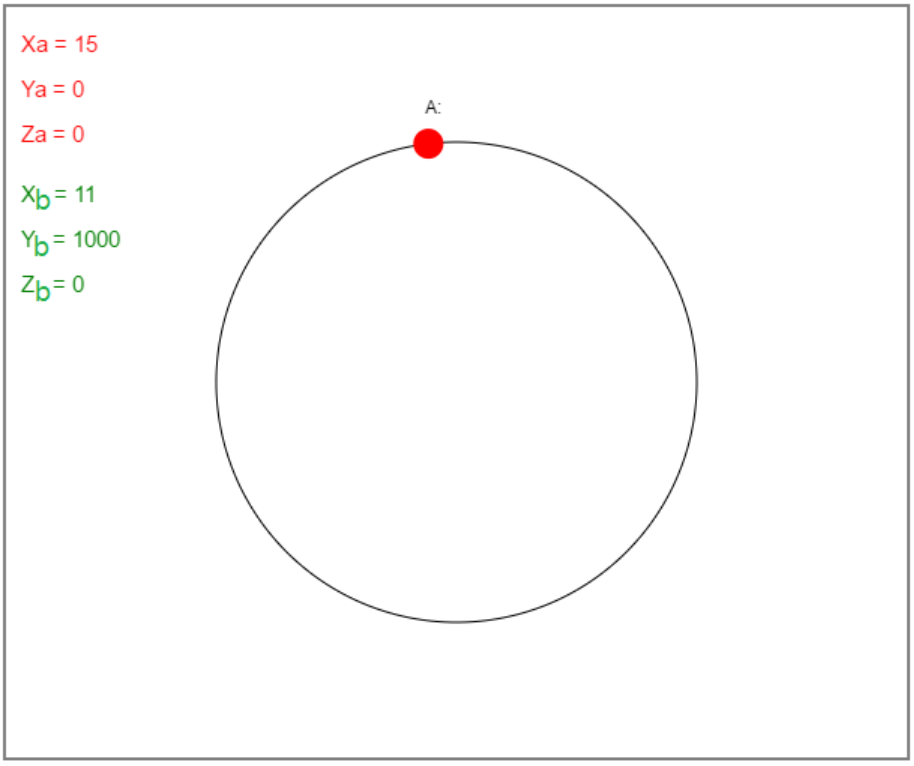
A: Membership function for speed:
very slow: **0**
slow: **0**
medium: **1**
fast: **0**
very fast: **0**

B: Membership function for speed:
very slow: **0**
slow: **0**
medium: **0.75**
fast: **0.25**
very fast: **0**



A: Membership function for speed:
very slow: **0**
slow: **0**
medium: **0.25**
fast: **0.75**
very fast: **0**

B: Membership function for speed:
very slow: **0**
slow: **0**
medium: **0.75**
fast: **0.25**
very fast: **0**



A: Membership function for speed:
very slow: **0**
slow: **0**
medium: **0**
fast: **0.75**
very fast: **0.25**

B: Membership function for speed:
very slow: **0**
slow: **0**
medium: **0.75**
fast: **0.25**
very fast: **0**

Source code:

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Canvas</title>
        <link rel="stylesheet" type="text/css" href="style/style.css">
    </head>
    <body>
        <canvas id="canvas" width="600" height="500" style="border:2px
solid gray;"></canvas>
        <div class = "Container">
            <div class = "title">
               A: Membership function for speed:
            </div>
            <div>
                 very slow: <br />
                 slow: <br />
                 medium: <br />
                 fast: <br />
                 very fast: <br />
            </div>
        </div>
<div class = "Container">
            <div class = "title">
               B: Membership function for speed:
            </div>
            <div>
                 very slow: <br />
                 slow: <br />
                 medium: <br />
                 fast: <br />
                 very fast: <br />
            </div>
        </div>
 <script src = "script/index.js"></script>
    </body></html>
```

script/index.js

```
"use strict";
let cnv = document.getElementById("canvas");
let ctx = cnv.getContext("2d");

const Width = 600;
const Height = 500;

const CenterX = 300;
const CenterY = 250;
const Radius = 160;
let lengthOfCircle =  2 * Math.PI * Radius;/*1000*/
```

```
const TrainsColor1 = "red";
const TrainsColor2 = "green";

const RadOfTrain = 10;

let currAngleOfTrain1 = 0;
let currAngleOfTrain2 = 180;

let step1 = 10;
let step2 = 10;

let intervalOfCh = 1700;

let distance1;
let distance2;

let stop;

ctx.beginPath();
ctx.arc(CenterX, CenterY, Radius, 0, 2*Math.PI);
ctx.stroke();


function drawTrainByAngle(angle, trainsColor) {
    let {x,y} = getCoordFromAngle(angle, Radius);

    ctx.beginPath();
    ctx.arc(x, y, RadOfTrain, 0, 2*Math.PI);
    ctx.fillStyle = trainsColor;
    ctx.fill();
}

function drawDiscriptionOfTrain(angle, text) {
    let {x,y} = getCoordFromAngle(angle, Radius + 20);
    ctx.fillStyle = "black";
    ctx.beginPath();
    ctx.font = "12px Arial";

    ctx.fillText(text ,x,y);

}

function getCoordFromAngle(angle, radius ) {
    let x = Math.cos( angle * Math.PI / 180) * radius + CenterX;
    let y = Math.sin( angle * Math.PI / 180) * radius + CenterY;
 //console.log("angle = %d", Math.cos( angle * Math.PI / 180) *
Radius);
 //console.log("x = %d\ny = %d",x,y);
    return {x,y};
```

```
}
function speedEffect2() {
    if(Math.random()<0.5)
        step2 += Math.random()*3;
    else
        step2 -= Math.random()*3;
    if(step2 < 2) step2 = 2;
    if(step2>20)step2 = 20;
    //return step2;
}

//setInterval(speedEffect2, intervalOfCh);
function pixelToAngle(pixel) {
    let k = pixel / lengthOfCircle;
    return k * 360;
}

function angleToPixel( angle ) {
    // let k = angle / 360;
    // return k * lengthOfCircle;
    return Math.PI * Radius * angle / 180;
}


function simulate() {

    ctx.clearRect(0,0,Width,Height);

    ctx.beginPath();
    ctx.arc(CenterX, CenterY, Radius, 0, 2*Math.PI);
    ctx.stroke();

    drawTrainByAngle(currAngleOfTrain1, TrainsColor1);
    drawTrainByAngle(currAngleOfTrain2, TrainsColor2);

    drawDiscriptionOfTrain(currAngleOfTrain1, "A:");
    drawDiscriptionOfTrain(currAngleOfTrain2, "B:");

    // let currStep1 = speedEffect(step1);
    // let currStep2 = speedEffect(step2);

    console.log("speed1 = %d; angle1 = %d", step1, currAngleOfTrain1);
    console.log("speed2 = %d; angle2 = %d", step2, currAngleOfTrain2);
    console.log("");

    let diffrence = currAngleOfTrain2 - currAngleOfTrain1;
    if(diffrence > 0) {
        distance1 = angleToPixel( diffrence );
        distance2 = lengthOfCircle - distance1;
    }
```

```
    else {
        diffrence = -diffrence;
        distance2 = angleToPixel( diffrence );
        distance1 = lengthOfCircle - distance2;
    }

    distance1 -= 2.655;/** 2,655 - погрешность расчётов */
    distance2 -= 2.655;

    if(distance1 < 0 || distance2 < 0)stopSimulate();

    ctx.font = "15px Arial";
    ctx.fillStyle = "red";
    ctx.fillText("Xa = " + step1 ,10,30);
    ctx.fillText("Ya = " + ( distance1.toFixed(0) ) ,10,60);
    ctx.fillText("Za = " + 0  ,10,90);

    ctx.fillStyle = "green";
    ctx.fillText("Xb = " + step2 ,10,130);
    ctx.fillText("Yb = " + (distance2.toFixed(0) ) ,10,160);
    ctx.fillText("Zb = " + 0 ,10,190);



    currAngleOfTrain1 += pixelToAngle( step1 );
    currAngleOfTrain2 += pixelToAngle( step2 );

     if(distance1 < 0 || distance2 < 0)stopSimulate();

    if(currAngleOfTrain1 > 360)currAngleOfTrain1 -= 360;
    if(currAngleOfTrain2 > 360)currAngleOfTrain2 -= 360;

}



function speedEffect() {
    step1 += 1 - Math.floor( Math.random() * 3 );
    step2 += 1 - Math.floor( Math.random() * 3 );

    if(step1<0) step1 = 0;
    else if(step1>20) step1 = 20;

    if(step2<0) step2 = 0;
    else if(step2>20) step2 = 20;
}

function stopSimulate() {
    ctx.clearRect(0,0,Width,Height);
```

```
    ctx.beginPath();
    ctx.arc(CenterX, CenterY, Radius, 0, 2*Math.PI);
    ctx.stroke();

    drawTrainByAngle(currAngleOfTrain1, TrainsColor1);


    drawDiscriptionOfTrain(currAngleOfTrain1, "A:");

    ctx.font = "15px Arial";
    ctx.fillStyle = "red";
    ctx.fillText("Xa = " + step1 ,10,30);
    ctx.fillText("Ya = " + 0  ,10,60);
    ctx.fillText("Za = " + 0  ,10,90);

    ctx.fillStyle = "green";
    ctx.fillText("Xa = " + step2 ,10,130);
    ctx.fillText("Ya = " + 1000  ,10,160);
    ctx.fillText("Za = " + 0 ,10,190);

    clearInterval(stop);
}

stop = setInterval(simulate, 50);

setInterval(speedEffect, intervalOfCh);
```