Contemporary Data Processing Technology (CCOD)
Lab 2 (September 15, 2016)
Savchuk Artem, AS-36

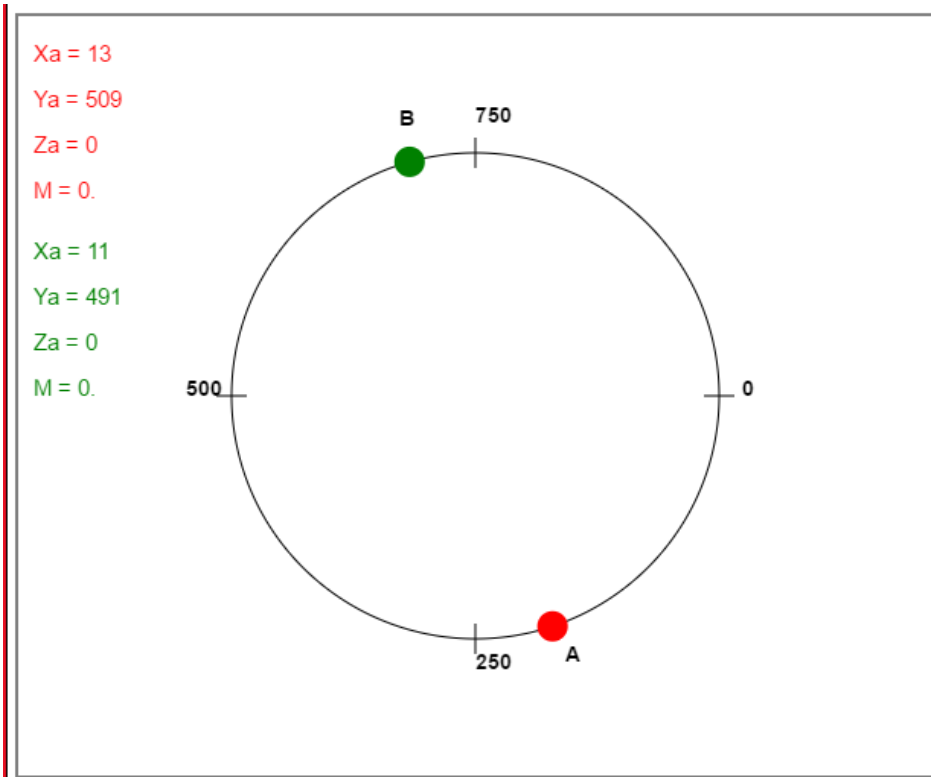
Controll two metro cars using Speed, Distance, and Brake. In this work we are design a virtual loop with 1000 pixels on which two metro cars(A – first train car and B – second train car). We don't assume any stations. That is, both train always run. The speed of train is denoted as Xa and Xb . Speed ranges from 0 to 20 pixels per unit of time. The distance from train A to B is denoted as Ya and from train B to train A is Yb , so Ya + Yb = 1000. Both train move in a clockwise direction. At the beginning speed of trains is 10 pixels per unit of time (Xa = 10, Xb = 10) and Ya = 500, Yb = 500; Step by step the speed changes by -2, 0, +2 randomly. Speed will be controlled by the distance to the train in front via its break. The break of train A and B is denoted as Za and Zb. And we change break every step with the following algorithm:

```
{if x>=14 then z=1, x=x-1} & {if y<=50 then z=9, x=x-9}
```

We also need to show the membership value of 3 rules:

```
{IF x=medium AND y=small THEN z=strong} OR {IF x=medium AND y=medium THEN
z=medium} OR {IF x=medium AND y=large THEN z=week}
```

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



Calculation of membership value of 3 rule according to the screenshot above.

IF x=medium AND y=small THEN z=strong

A	speed	μ	Distance	μ	break	μ	total μ
	13	0.25	509	0	0	0	0

B	speed	μ	Distance	μ	break	μ	total μ
	11	0.75	491	0	0	0	0

IF x=medium AND y=medium THEN z=medium

A	speed	μ	Distance	μ	break	μ	total μ
	13	0.25	509	0.455	0	0	0

B	speed	μ	Distance	μ	break	μ	total μ
	11	0.75	491	0.545	0	0	0

IF x=medium AND y=large THEN z=week

A	speed	μ	Distance	μ	break	μ	total μ
	13	0.25	509	0.555	0	0	0

B	speed	μ	Distance	μ	break	μ	total μ
	11	0.75	491	0,455	0	0	0

Membership value:

A: Rule1 or Rule2 or Rule3 => 0

B: Rule1 or Rule2 or Rule3 => 0

Xa = 6

Ya = 775

Za = 0

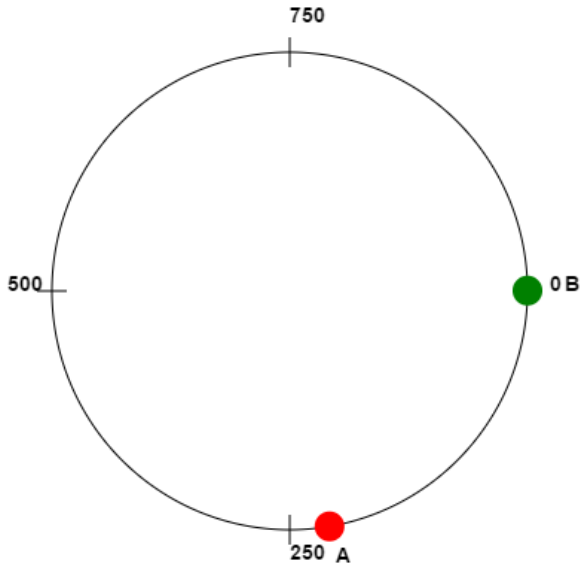
M = 0.

Xa = 11

Ya = 225

Za = 0

M = 0.



Xa = 2

Ya = 963

Za = 0

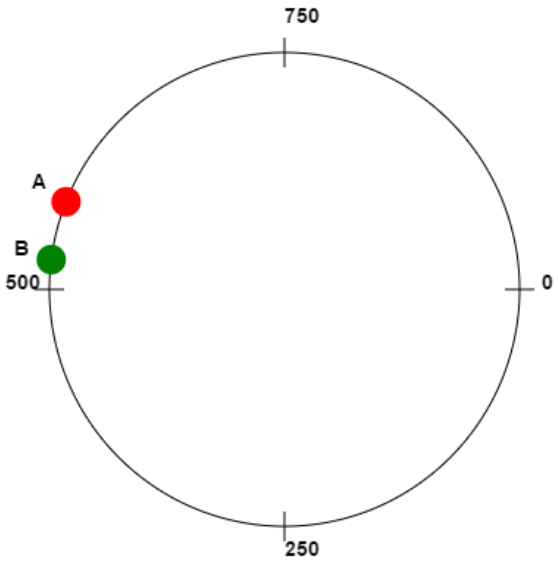
M = 0.

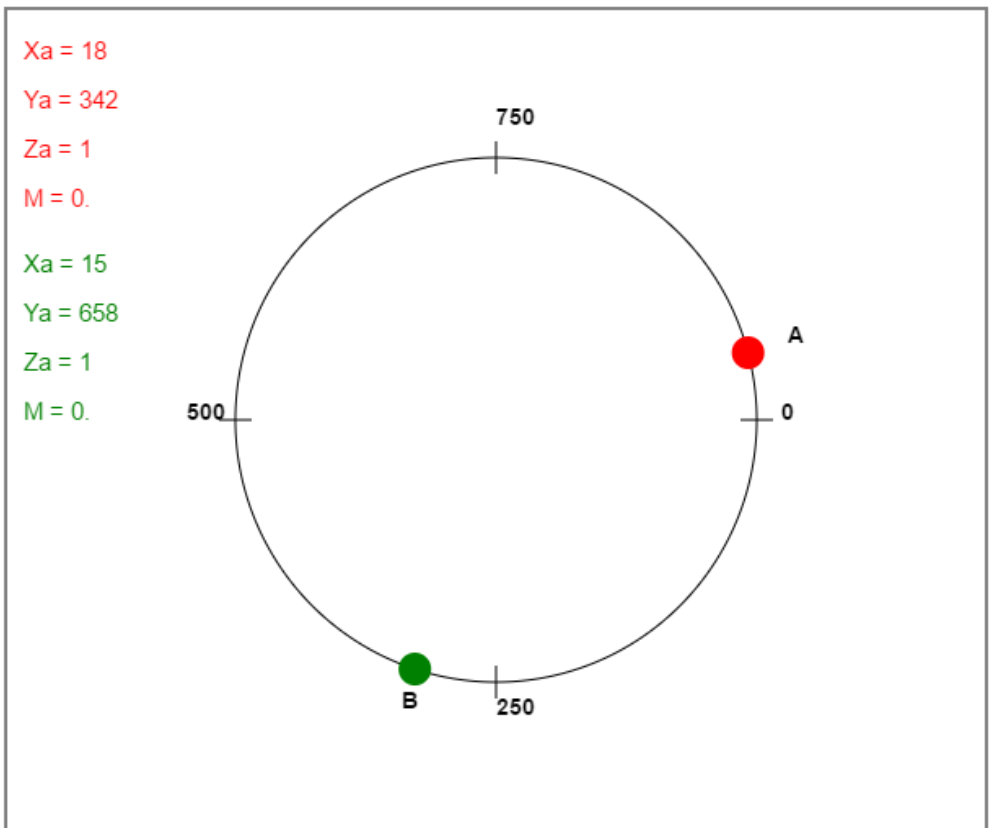
Xa = 0

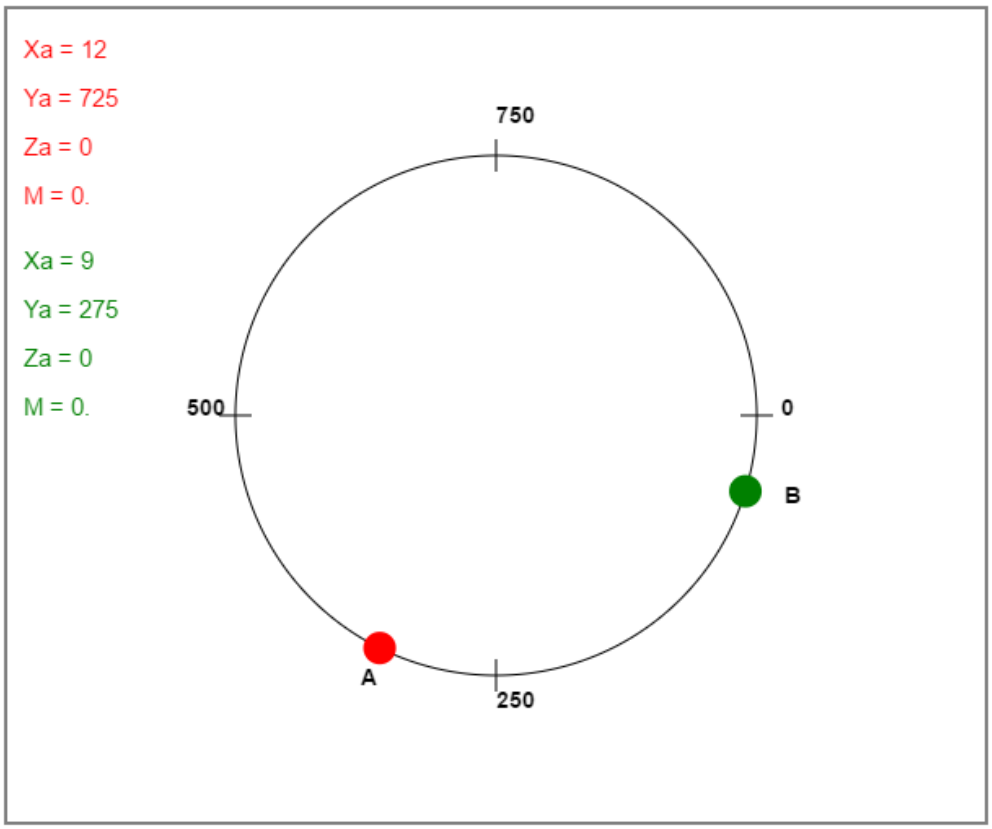
Ya = 37

Za = 9

M = 0.







As shown above, in every situation the membership value is equal to 0. Because of Z only takes the following values: 0, 1, 9. Membership function strong, medium and weak is always equal to zero when break takes these values.

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>
        <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 stepInterval = 15; /*  10 * 25 = 250  */

let speed = {
    set step1(value) {
        if(value < 0) step1 = 0;
        else if(value > 20) step1 = 20;
        else step1 = value;
        //alert(step1);

    },
    get step1(){
        return step1;
    },
```

```
    set step2(value) {
        if(value < 0) step2 = 0;
        else if(value > 20) step2 = 20;
        else step2 = value;
        //alert(step2);
    },
    get step2(){
        return step2;
    }


}



let intervalOfCh = 500;

let distance1; /**distance between red and green train */
let distance2; /**distance between green and red train */

let z1 = 0;
let z2 = 0;

let stop;
let stop2;

let count = 0;

drawRailway();

function showInfo()
{
    ctx.clearRect(0,0,100,Height);
    ctx.font = "15px Arial";
    ctx.fillStyle = "red";
    ctx.fillText("Xa = " + step1 ,10,30);
    ctx.fillText("Ya = " + ( distance1.toFixed(0) ) ,10,60);
    ctx.fillText("Za = " + z1  ,10,90);
    ctx.fillText("M = 0."  ,10,120);

    ctx.fillStyle = "green";
    ctx.fillText("Xa = " + step2 ,10,160);
    ctx.fillText("Ya = " + (distance2.toFixed(0) ) ,10,190);
    ctx.fillText("Za = " + z2 ,10,220);
    ctx.fillText("M = 0."  ,10,250);
}

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

    let c = getCoordFromAngle(0, Radius)
    ctx.beginPath();
    ctx.moveTo( c.x-10, c.y );
    ctx.lineTo( c.x+10, c.y );
```

```
    ctx.stroke();
    ctx.fillStyle = 'black';
    ctx.font = "bold 14px Arial";
    ctx.fillText('0' , c.x+15, c.y);

    c = getCoordFromAngle(90, Radius)
    ctx.beginPath();
    ctx.moveTo( c.x, c.y-10 );
    ctx.lineTo( c.x, c.y+10 );
    ctx.stroke();
    ctx.fillStyle = 'black';
    ctx.font = "bold 14px Arial";
    ctx.fillText('250' , c.x, c.y+20);

    c = getCoordFromAngle(180, Radius)
    ctx.beginPath();
    ctx.moveTo( c.x-10, c.y );
    ctx.lineTo( c.x+10, c.y );
    ctx.stroke();
    ctx.fillStyle = 'black';
    ctx.font = "bold 14px Arial";
    ctx.fillText('500' , c.x-30, c.y);

    c = getCoordFromAngle(270, Radius)
    ctx.beginPath();
    ctx.moveTo( c.x, c.y-10 );
    ctx.lineTo( c.x, c.y+10 );
    ctx.stroke();
    ctx.fillStyle = 'black';
    ctx.font = "bold 14px Arial";
    ctx.fillText('750' , c.x, c.y-20);

}

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 + 25);
    ctx.fillStyle = "black";
    ctx.beginPath();
    ctx.font = "bold 14px 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;
```

```
    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;

}

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 calculateZ() {
     if(step1 >=14) {
        z1 = 1;
        //speed.step1 = step1 - z1;

    }
    else if(z1 === 1) z1 = 0;
    if(step2 >=14) {
        z2 = 1;
        //speed.step2 = step2 - z2;

    }
    else if(z2 === 1) z2 = 0;


    if(distance1 <= 50){z1 = 9; /*speed.step1 = step1 - 9;*/}
    else if(z1 == 9 ) z1 = 0;
    if(distance2 <= 50) {z2 = 9; /*speed.step2 = step2 - 9;*/}
    else if(z2 == 9 ) z2 = 0;
}


function simulate() {

    ctx.clearRect(100,0,Width,Height);

    drawRailway();

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

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

    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(); return;}


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


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

    if((count % stepInterval) == 0) {


        speedEffect();
        calculateZ();

        speed.step1 = step1 - z1;
        speed.step2 = step2 - z2;

        showInfo();

        //count = 0;
    }
    count++;
    if(count === 1000) count = 1;


}

function speedEffect() {

    speed.step1 = step1 + (1 - Math.floor( Math.random() * 3 )) * 2;
    speed.step2 = step2 + (1 - Math.floor( Math.random() * 3 )) * 2;

    console.log("step1 = " + step1 + '  ' + "step2 = " + step2);

}

function stopSimulate() {
```

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

    drawRailway();

    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 = " + z1  ,10,90);

    ctx.fillStyle = "green";
    ctx.fillText("Xa = " + step2 ,10,130);
    ctx.fillText("Ya = " + 1000  ,10,160);
    ctx.fillText("Za = " + z2 ,10,190);

    clearInterval(stop);
    clearInterval(stop2);
}

stop = setInterval(simulate, 25);
```