Contemporary Data Processing Systems
Lab 2 of 15/09/2016
AS-36, Savitsky Anton

Control 2 metro trains using Speed, Distance, and Brake. What we have:

-a virtual loop with 1000 px on which there are

-two metro trains(1 – first train car and 2 – second train car)

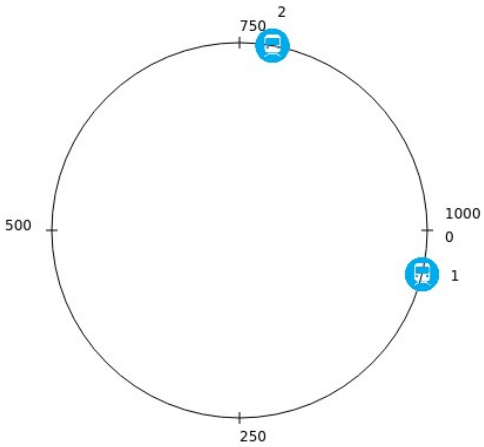
Both train always run. The speed of trains is denoted as X1 and X2 . Speed ranges from 0 to 20 px per unit of time. The distance from train 1 to 2 is denoted as Y1 and from train 2 to train 1 is Y2 , so Y1 + Y2 = 1000. Both trains move in a clockwise direction. At the beginning speed of trains is 10 pixels per unit of time (X1 = 10, X2 = 10) and Y2 = 500, Y1 = 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 1 and 2 is denoted as Z1 and Z2. And we change break value each 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 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}



X1 = 17	X2 = 13
Y1 = 742	Y2 = 258
Z1 = 1	Z2 = 0
M1 = 0	M2 = 0

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

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

1	speed	μ	Distance	μ	break	μ	total μ
	17	0	742	0	1	0	0

2	speed	μ	Distance	μ	break	μ	total μ
	13	0.25	491	0	0	0	0

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

1	speed	μ	Distance	μ	break	μ	total μ
	17	0	742	0	1	0	0

2	speed	μ	Distance	μ	break	μ	total μ
	13	0.25	491	0,5	0	0	0

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

1	speed	μ	Distance	μ	break	μ	total μ
	17	0	742	0.25	1	0	0

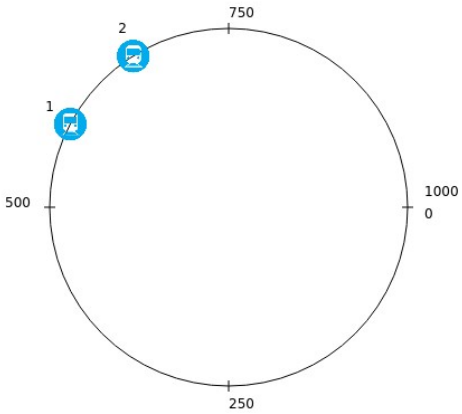
2	speed	μ	Distance	μ	break	μ	total μ
	13	0.25	491	0,5	0	0	0

Membership function value:

for 1 train: rule 1 or rule 2 or rule 3 => 0

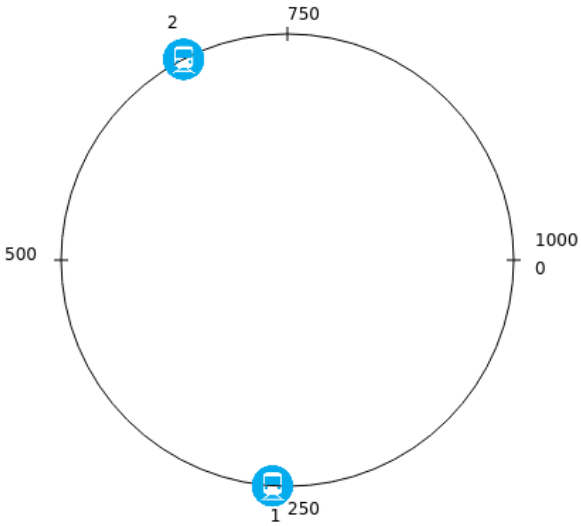
for 2 train: rule 1 or rule 2 or rule 3 => 0

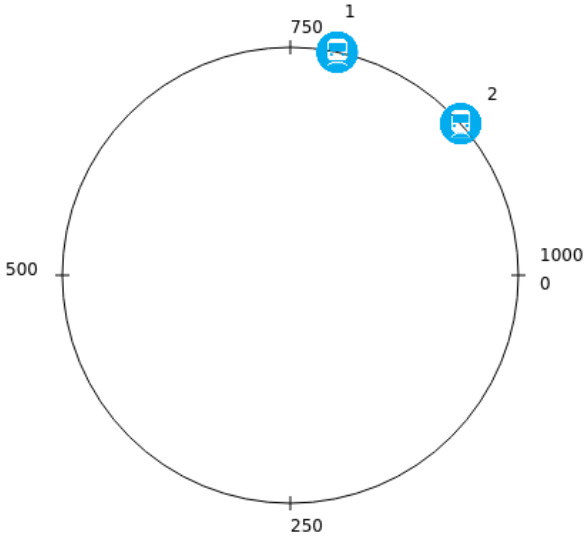
Other snapshots:



X1 = 4
Y1 = 76
Z1 = 0
M1 = 0

X2 = 12
Y2 = 924
Z2 = 0
M2 = 0





X1 = 11
Y1 = 414
Z1 = 0
M1 = 0

X2 = 8
Y2 = 587
Z2 = 0
M2 = 0

X1 = 0
Y1 = 97
Z1 = 0
M1 = 0

X2 = 12
Y2 = 903
Z2 = 0
M2 = 0

Source code was written in html5 with canvas graphics:

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Lab2 - 15_09_2016</title>
    <link rel="stylesheet" type="text/css" href="style/style.css">
  </head>
  <body>
    <canvas id="metro-canvas" width="600" height="700"></canvas>
    <img src="train.png" id="train-img" style="width: 30px; height: 30px; display: none;">
    <script src = "index.js"></script>
  </body>
</html>
```

index.js

```
"use strict";

var canvas = document.getElementById("metro-canvas");
var context = canvas.getContext("2d");
var centerOfX = 300;
var centerOfY = 250;
var R = 160;

var circleLength =  2 * Math.PI * R;

var trainR = 15;
```

```
var angleOfA = 0;
var angleOfB = 180;

var stepOf1 = 10;/*ms*/
var stepOf2 = 10;/*ms*/

var width = 600;
var height = 500;

var stepInterval = 10;/*ms*/

var dist1To2;
var dist2To1;

var z1 = 0; //break
var z2 = 0;

var stop1;
var stop2;

var c = 0;

drawMetroLine();

function showData()
{
   context.clearRect(0,450,width,height);
   context.font = "20px Arial";
   context.fillStyle = "blue";
   context.fillText("X1 = " + stepOf1 ,150,530);
   context.fillText("Y1 = " + ( dist1To2.toFixed(0) ) ,150,560);
   context.fillText("Z1 = " + z1  ,150,590);
   context.fillText("M1 = 0"  ,150,620);

   context.fillStyle = "red";
   context.fillText("X2 = " + stepOf2 ,400,530);
   context.fillText("Y2 = " + (dist2To1.toFixed(0) ) ,400,560);
   context.fillText("Z2 = " + z2 ,400,590);
   context.fillText("M2 = 0"  ,400,620);
}

function drawMetroLine() {
   context.linewidth = 2;
   context.beginPath();
   context.arc(centerOfX, centerOfY, R, 0, 2*Math.PI);
   context.stroke();

   context.fillStyle = 'black';
   context.font = "12px Arial";

   let c = angleToCoordinate(0, R)
   context.beginPath();
   context.moveTo( c.x-5, c.y );
   context.lineTo( c.x+5, c.y );
   context.stroke();
   context.fillText("1000" , c.x+15, c.y-10);
   context.fillText("0" , c.x+15, c.y+10);

   c = angleToCoordinate(90, R)
   context.beginPath();
   context.moveTo( c.x, c.y-5 );
   context.lineTo( c.x, c.y+5 );
   context.stroke();
```

```
    context.fillText('250' , c.x, c.y+20);

    c = angleToCoordinate(180, R)
    context.beginPath();
    context.moveTo( c.x-5, c.y );
    context.lineTo( c.x+5, c.y );
    context.stroke();
    context.fillText('500' , c.x-40, c.y);

    c = angleToCoordinate(270, R)
    context.beginPath();
    context.moveTo( c.x, c.y-5 );
    context.lineTo( c.x, c.y+5 );
    context.stroke();
    context.fillText('750' , c.x, c.y-10);

}

let speed = {
    set stepOf1(value) {
        if(value < 0) stepOf1 = 0;
        else if(value > 20) stepOf1 = 20;
        else stepOf1 = value;
    },
    get stepOf1(){
        return stepOf1;
    },

    set stepOf2(value) {
        if(value < 0) stepOf2 = 0;
        else if(value > 20) stepOf2 = 20;
        else stepOf2 = value;
    },
    get stepOf2(){
        return stepOf2;
    }
}

function drawTrainByAngle(angle) {
    let {x,y} = angleToCoordinate(angle, R);

    var img=document.getElementById("train-img");

    context.beginPath();
    context.drawImage(img,x-15,y-15, 30, 30);
}

function drawTrainInfo(angle, text) {
    let {x,y} = angleToCoordinate(angle, R + 25);
    context.fillStyle = "black";
    context.beginPath();
    context.font = "12px Arial";

    context.fillText(text ,x,y);

}

function angleToCoordinate(angle, radius) {
    let x = Math.cos( angle * Math.PI / 180) * radius + centerOfX;
    let y = Math.sin( angle * Math.PI / 180) * radius + centerOfY;
    return {x,y};
}
```

```
function changeSpeedOf2() {
   if(Math.random()<0.5)
      stepOf2 += Math.random()*3;
   else
      stepOf2 -= Math.random()*3;
   if(stepOf2 < 2) stepOf2 = 2;
   if(stepOf2>20)stepOf2 = 20;

}

function pxToAngle(pixel) {
   let k = pixel / circleLength;
   return k * 360;
}

function angleToPx( angle ) {
   return Math.PI * R * angle / 180;
}

function calculateZ() {
    if(stepOf1 >=14)
      z1 = 1;
   else if(z1 === 1) z1 = 0;
   if(stepOf2 >=14)
      z2 = 1;
   else if(z2 === 1) z2 = 0;


   if(dist1To2 <= 50)
    z1 = 9;
   else if(z1 == 9 ) z1 = 0;
   if(dist2To1 <= 50)
    z2 = 9;
   else if(z2 == 9 ) z2 = 0;
}


function simulate() {

   context.clearRect(100,0,width,height);

   drawMetroLine();

   drawTrainByAngle(angleOfA);
   drawTrainByAngle(angleOfB);

   drawTrainInfo(angleOfA, "1");
   drawTrainInfo(angleOfB, "2");

   let diffirence = angleOfB - angleOfA;
   if(diffirence > 0) {
      dist1To2 = angleToPx( diffirence );
      dist2To1 = circleLength - dist1To2;
   }
   else {
      diffirence = -diffirence;
      dist2To1 = angleToPx( diffirence );
      dist1To2 = circleLength - dist2To1;
   }
```

```
    dist1To2 -= 2.6;
    dist2To1 -= 2.6;

    if(dist1To2 < 0 || dist2To1 < 0){
      stopSimulate();
      return;
    }

    angleOfA += (pxToAngle( stepOf1 ))/stepInterval;
    angleOfB += (pxToAngle( stepOf2 ))/stepInterval;

    if(angleOfA > 360)
      angleOfA -= 360;
    if(angleOfB > 360)
      angleOfB -= 360;

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

        effectSpeedOf1();
        calculateZ();

        speed.stepOf1 = stepOf1 - z1;
        speed.stepOf2 = stepOf2 - z2;

        showData();
    }
    c++;
    if(c === 1000) c = 1;


}

function effectSpeedOf1() {
    speed.stepOf1 = stepOf1 + (1 - Math.floor( Math.random() * 3 )) * 2;
    speed.stepOf2 = stepOf2 + (1 - Math.floor( Math.random() * 3 )) * 2;

    console.log("stepOf1 = " + stepOf1 + '  ' + "stepOf2 = " + stepOf2);
}


stop = setInterval(simulate, 25);



function stopSimulate() {
    context.clearRect(0,0,width,height);

    drawMetroLine();

    drawTrainByAngle(angleOfA, TrainsColor);

    drawTrainInfo(angleOfA, "1");

    drawTrainInfo(angleOfB, "2");

    clearInterval(stop1);
    clearInterval(stop2);
}
```