Controlarea vehiculelor cu roti utilizând kit-ul Lego Mindstorm NXT – partea a II-a

Un tip comun de roboti este vehiculul cu doua roti cu motoare controlate independent. Acest tip de structură foloseşte viraj diferenţial şi se poate roti în loc. LeJOS NXJ contine cateva clase pentru a-l controla.

Pilot

Clasa Pilot vireaza vehiculul prin controlarea directiei si vitezei de rotatie a motoarelor acestuia. Pilot trebuie sa stie diagram electrica a robotului, spre exemplu la care porturi sunt motoarele conectate si daca conducand motoarele inainte vor face robotul sa se miste inainte sau inapoi (reverse). Trebuie de asemenea stiut diametrul rotilor si largimea traseului, spre exemplu distant intre centrele rutelor celor doua roti. Clasa Pilot foloseste diametrul rotii pentru a calcula distant pe care acesta a parcurs-o.

Se foloseste ratia pentru a calcula cat de departe aceasta s-a rotit. Evident, ambii parametrii trebuie sa fie in aceleasi unitati de masura, dra ei pot fi orice iti doresti. Cu reglarea corespunzatoare a acestor parametrii, erorile in distanta parcursa si unghiul de rotatie pot fi tinute la 2% sau poate mai putin. Aceste informatii sunt transmise constructorului pilot.

Constructori:

- Pilot(float wheelDiameter, float trackWidth,M otor leftMotor, Motor rightMotor)
- Pilot(float wheelDiameter, float trackWidth, Motor leftMotor, Motor rightMotor, boolean reverse)

  Folositi acest constructor daca doriti sa setati variabila boolean *reverse* la valoarea ***true***.

Miscarea in linie dreapta

Pentru a controla robotul sa mearga in linie dreapta, folositi:

- void setSpeed(int speed)

  seteaza viteza motoarelor in grade/secunda

- void forward()

  porneste miscarea robotului in fata

- void backward()
- void stop()

Pentru a controla distant pe care robotul o parcurge, folositi:

- void travel(float distance)
- void travel(float distance, boolean immediateReturn)

  distance are aceeasi unitate de masura ca si diametrul rotii; o distant negative inseamna sa mearga in spate. Puteti afla cat de departe a mers un robot apeland

- int getTravelDistance(); - returneaza distant parcursa pana la ultima apelare a resetTachoCount();

Exemplu:

```
import lejos.nxt.*;
import lejos.navigation.Pilot;

/**
 * Robotul care se opreste daca loveste ceva in calea lui, isi termina
  *  calatoria.
 */
public class TravelTest {
  Pilot pilot;
  TouchSensor bump = new TouchSensor(SensorPort.S1);

  public void go() {
    pilot.travel(20, true);
    while (pilot.isMoving()) {
      if (bump.isPressed()) pilot.stop();
    }
    System.out.println(" "+pilot.getTravelDistance());
    Button.ewaitForPress();
  }

  public static void main(String[] args) {
    TravelTest traveler = new TravelTest();
    traveler.pilot = new Pilot(2.25f, 5.5f, Motor.A, Motor,C);
    traveler.go();
  }
}
```

Puteti seta robotul sa se roteasca in loc specificand unghiul si folosind

- void rotate(int degrees)

  Trebuie sa aveti valori exacte pentru wheelDiameter si trackWidth ca aceasta metoda sa produca rezultate exacte.

Program SquareTracer

Scrieti un program care foloseste o clasa Pilot pentru inconjurarea unui careu, folosind metodele travel si rotate.

```
import lejos.nxt.*;
import lejos.navigation.Pilot;

/**
 * Inconjoara un careu
 * @author Roger
 */
public class SquareTracer
{
   Pilot pilot ;
   public void  drawSquare(float length)
   {
      for(int i = 0; i<4 ; i++)
      {
         pilot.travel(length);
         pilot.rotate(90);
      }
   }
   public static void main(String[] args)
   {
      SquareTracer sq = new SquareTracer();
      sq.pilot = new Pilot(2.25f, 5.5f, Motor.A, Motor.C);
      sq.drawSquare(20);
   }
}
```

Program SquareTracer2

Scrieti un program care inconjoara 2 careuri cu unghiul la colturi marit, dupa care reface aceeasi cale in directia opusa. Modificati metoda traceSquare a programului Pilot 1 astfel incat sa poata inconjura un careu in oricare directive, si folositi asta in acest program. Acesta este un test strict al acuratetei diametrului rotii si al constantei largimii rutei folosita in pilot.

```
import lejos.nxt.*;
import lejos.navigation.Pilot;

/**
 * Inconjoara doua careuri, de doua ori.
 * @author Roger
 */
public class SquareTracer2
{
     Pilot pilot ;
   public void  drawSquare(float length)
   {
      byte direction = 1;
      if(length < 0 )
      {
         direction = -1;
         length = -length;
      }
      for(int i = 0; i<4 ; i++)
      {
         pilot.travel(length);
         pilot.rotate(direction * 90);
      }
   }
   public static void main( String[] args)
   {
      System.out.println(" Square Tracer 2");
      Button.waitForPress();
      SquareTracer2 sq = new SquareTracer2();
```

```
        sq.pilot = new Pilot(2.25f, 5.5f, Motor.A, Motor.C);
        byte direction = 1;
        int length = 20;
        for(int i = 0; i<4; i++)
        {
           sq.drawSquare(direction * length );
           if( i == 1)
           {
              sq.pilot.rotate( 90);
              direction = -1;
           }
        }
    }
}
```

Miscarea de-a lungul unei cai curbate

Clasa pilot poate intoarce robotul in loc conducand o roata in fata si alta in spate. Metodele care realizeaza acest lucru sunt:

- void rotate(int angle)
- void rotate(int angle, boolean immediateReturn)

  Daca ***angle*** este pozitiv, robotul se intoarce spre stanga. Parametrul ***immediateReturn*** lucreaza ca si in metoda Motor – permite thread – ului apelant sa lucreze altceva in timp ce cerinta de rotatie este in desfasurare.

Clasa Pilot poate controla robotul sa se miste pe o cale circulara folosind urmatoarele metode:

- void steer(int turnRate) – urmeaza o cale circulara pana cand alta metoda este executata
- void steer(int turnRate, int angle)
- void steer(int turnRate, int angle, boolean immediateReturn)

Parametrul ***turnRate*** determina raza caii. O valoare pozitiva inseamna ca central cercului se afla la stanga robotului (deci motorul din stanga

actioneaza roata din interior). O valoare negative inseamna ca motorul stang actioneaza roata din exterior. Valoarea absoluta este intre 0 si 200, si determina ratia vitezei intre motorul din interior si cel exterior. Motorul exterior ruleaza la viteza setata a robotului; motorul interior este incetinit pentru a face robotu sa se intoarca. La rata de intoarcere 0, ratia vitezei este 1.0 si robotul merge in linie dreapta. La rata de intoarcere 200, ratia vitezei este -1 si robotul se invarte in loc. Rata de intoarcere 100 da o ratie a vitezei de 0, deci motorul interior se opreste. Formula este: speed ratio = 100 – abs (turnRate).

Parametrul ***angle*** determina unghiul de rotatie la care robotul se opreste. Daca unghiul este negative, robotul urmeaza calea circulara definita de rata de intoarcere, dar se misca in spate.

- getAngle() – returneaza unghiul de rotatie al vehiculului pana la ultima apelare a resetTachoCount()

Program SteerTester

Scrieti un program care foloseste ButtonCounter pentru a introduce variabilele ratei de intoarcere si a unghiului, dupa care apeleaza metoda steer(). Toate astea se vor face intr-o bucla ca sa puteti incerca valori diferite pentru acesti parametrii pentru a controla calea robotului.

```
import lejos.nxt.*;
import lejos.navigation.*;
import lejos.util.*;

/**
 * Testeaza metoda turn()
 * Butonul stang introduce 100's, dreptul - 10's.
 */
public class SteerTest
{
   public static void main( String[] args)
   {
      Pilot pilot = new Pilot(2.25f, 4.8f, Motor.A, Motor.C); //units: inches
      ButtonCounter bc = new ButtonCounter();
```

```
        while(true)
        {
           bc.count("Turn Rate x10");
           int turnRate = 100 * bc.getLeftCount() + 10 * bc.getRightCount();
           bc.count("Angle x 10");
           int angle = 100 * bc.getLeftCount() + 10 * bc.getRightCount();
           pilot.steer(turnRate,angle);
        }
    }
}
```

Alte metode pentru Pilot

- void resetTachocount()

  Reseteaza ambele motoare. Aceasta metoda NU este apelata de nicio alta metoda a clasei Pilot. Aceasta metoda trebuie apelata daca se doresc rezultate folositoare de la metoda getTravelDistance() sau de la metoda getAngle();

- void regulateSpeed(Boolean yes)

  Trebuie sa luati in considerare oprirea reglarii vitezei daca folositi sensor feedback si metoda steer() pentru a controla miscarea robotului.

- boolean isMoving()

  Returneaza true daca vreun motor se misca. Este folositoare daca ati folosit parametrul immediateReturn si trebuie sa stiti daca cerinta este in desfasurare.

- boolean stalled()

  Returneaza true daca viteza actuala a oricarui motor este zero. Amintiti-va, viteza actuala este calculate la fiecare 100ms. Deci stalled() va returna ***true*** pentru primele 100ms dupa ce robotul isi incepe miscarea.

Daca aveti nevoie sa folositi motoare individuale, puteti folosi:

- Motor getLeft()
- Motor getRight()