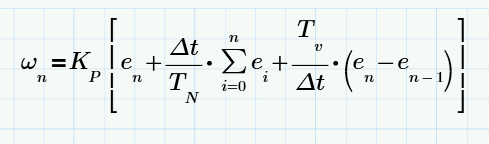
**Stunde vom 12.11.2014**

1. Aufgabenstellung:



ω

Ges: KP, TN, Tv

Programmieren des NXT als Line Follower mit einem PD-Regler.

1. Durchführung

* Eclipse zum Laufen gebracht
  + Lejos plugin installiert

1. Ergebnis

**Main.java**

1. Probleme + Lösungsvorschläge/Lösungen

* Problem beim Eclipse starten da falsche Eclipse version (64bit 🡪 32bit)
  + GELÖST: neue Eclipse version installiert (32bit)
    - Plugins funktionieren
  + Java 32 bit installieren
  + LeJos installieren

**Stunde vom 19.11.2014**

1. Aufgabenstellung

**Siehe Stunde vom 12.11.2014!**

1. Durchführung

**Programmcode: IFollower.java**

**import** lejos**.**nxt**.**Button**;**

**import** lejos**.**nxt**.**LCD**;**

**import** lejos**.**nxt**.**LightSensor**;**

**import** lejos**.**nxt**.**Motor**;**

**import** lejos**.**nxt**.**SensorPort**;**

public class IFollower **{**

public static void main**(**String**[]** args**)** **{**

LightSensor sensLeft **=** **new** LightSensor**(**SensorPort**.**S1**);**

LightSensor sensRight **=** **new** LightSensor**(**SensorPort**.**S2**);**

sensLeft**.**setFloodlight**(true);**

sensRight**.**setFloodlight**(true);**

double Tv **=** 0.05**;**//1.8;

double Tn **=** 1.5**;**

double Kp **=** 0.18**;**

double ei **=** 0**;**

double el **=** 0**;**

double diff **=** calibrate**(**sensLeft**,** sensRight**);**

double tt **=** 0**;**

double fspeed **=** 300**;**

double t **=** 0.05**;**

**while** **(true)** **{**

int left **=** sensLeft**.**getNormalizedLightValue**();**

int right **=** sensRight**.**getNormalizedLightValue**();**

double e **=** left **-** right **-** diff**;**

**if(**Math**.**abs**(**e**)<**10**)** **{**

e **=** 0**;**

**}**

ei **+=** e**;**

tt**-=**System**.**currentTimeMillis**()/**1000**;**

double w **=** Kp **\*** **(**e **+** t**/**Tn **\*** ei **+** Tv**/**t **\*** **(**e **-** el**));**

LCD**.**drawString**(**"l:" **+** left **+** " diff="**+**diff**+**"\nr:" **+** right **+** " tt="**+**tt**+**"\ne:" **+** e **+** "\nw:" **+** w**,** 0**,** 0**);**

tt**=** System**.**currentTimeMillis**()/**1000**;**

Motor**.**A**.**setSpeed**((**int**)** **(**fspeed**+**w**));**

Motor**.**B**.**setSpeed**((**int**)** **(**fspeed**-**w**));**

**if(**w**>**fspeed**)** **{**

LCD**.**drawString**(**"ERROR" **+** w**,** 30**,** 0**);**

**}**

Motor**.**A**.**forward**();**

Motor**.**B**.**forward**();**

el **=** e**;**

**try** **{**

Thread**.**sleep**((**long**)** **(**t**\***1000**));**

**}** **catch** **(**InterruptedException e1**)** **{**

e1**.**printStackTrace**();**

**}**

**}**

**}**

private static double calibrate**(**LightSensor sensLeft**,** LightSensor sensRight**)** **{**

LCD**.**drawString**(**"Calibrating \n(All white)"**,** 0**,** 0**);**

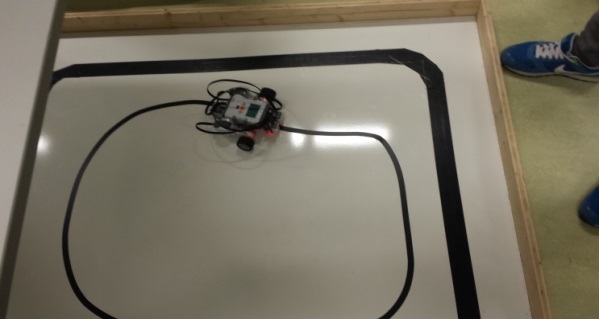
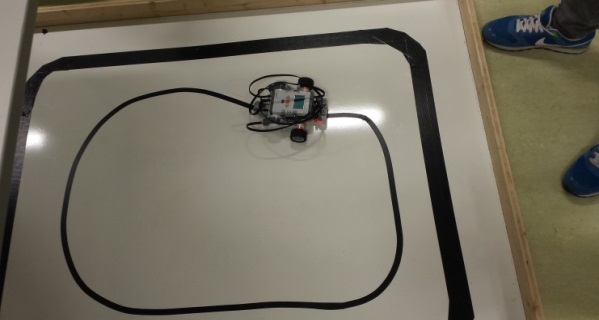
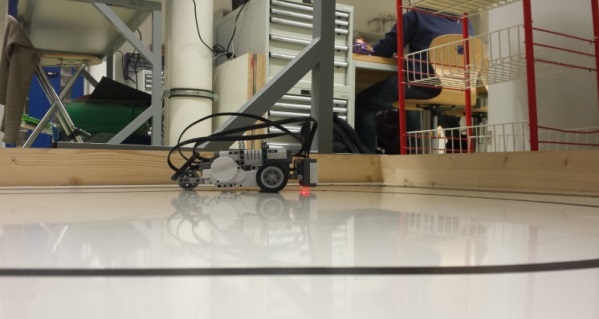
Button**.**waitForAnyPress**();**

**return** sensLeft**.**getNormalizedLightValue**()-**sensRight**.**getNormalizedLightValue**();**

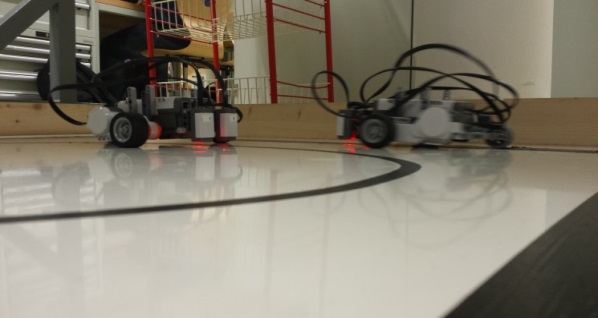
**}**

**}**

* **Durchgeführte Testläufe**







1. Ergebnis

**IFollower.java**

1. Probleme + Lösungsvorschläge / Lösungen

* In der Formel aus der Aufgebenstellung falsches DELTA (δ)
  + LÖSUNG: Aus DELTA δ 🡪 Δ
  + Δ : die Summe der zweiten [partiellen Ableitungen](http://de.wikipedia.org/wiki/Partielle_Ableitung) in jeder Raumrichtung
  + <http://de.wikipedia.org/wiki/Laplace-Operator>

**Stunde vom 26.11.2014**

1. Aufgabenstellung

**Siehe Stunde vom 12.11.2014!**

1. Durchführung

**Programmcode: IFollower.java**

**import** lejos**.**nxt**.**Button**;**

**import** lejos**.**nxt**.**LCD**;**

**import** lejos**.**nxt**.**LightSensor**;**

**import** lejos**.**nxt**.**Motor**;**

**import** lejos**.**nxt**.**SensorPort**;**

public class IFollower **{**

public static void main**(**String**[]** args**)** **{**

LightSensor sensLeft **=** **new** LightSensor**(**SensorPort**.**S1**);**

LightSensor sensRight **=** **new** LightSensor**(**SensorPort**.**S2**);**

sensLeft**.**setFloodlight**(true);**

sensRight**.**setFloodlight**(true);**

double Tv **=** 0.003**;**

double Tn **=** 1.5**;**

double Kp **=** 2.0**;**

double ei **=** 0**;**

double el **=** 0**;**

double diff **=** calibrate**(**sensLeft**,** sensRight**);**

double tt **=** 0**;**

double fspeed **=** 600**;**

double t **=** 0.02**;**

**while** **(true)** **{**

int left **=** sensLeft**.**getNormalizedLightValue**();**

int right **=** sensRight**.**getNormalizedLightValue**();**

double e **=** left **-** right **-** diff**;**

**if(**Math**.**abs**(**e**)<**10**)** **{**

e **=** 0**;**

**}**

**if(**Button**.**RIGHT**.**isDown**())** **{**

Tv **+=** 0.001**;**

**try** **{**

Thread**.**sleep**((**long**)** **(**500**));**

**}** **catch** **(**InterruptedException e1**)** **{**

e1**.**printStackTrace**();**

**}**

**}**

**if(**Button**.**LEFT**.**isDown**())** **{**

Tv **-=** 0.001**;**

**try** **{**

Thread**.**sleep**((**long**)** **(**500**));**

**}** **catch** **(**InterruptedException e1**)** **{**

e1**.**printStackTrace**();**

**}**

**}**

**if(**Button**.**ENTER**.**isDown**())** **{**

Kp **+=** 0.01**;**

**try** **{**

Thread**.**sleep**((**long**)** **(**500**));**

**}** **catch** **(**InterruptedException e1**)** **{**

e1**.**printStackTrace**();**

**}**

**}**

**if(**Button**.**ESCAPE**.**isDown**())** **{**

Kp **-=** 0.01**;**

**try** **{**

Thread**.**sleep**((**long**)** **(**500**));**

**}** **catch** **(**InterruptedException e1**)** **{**

e1**.**printStackTrace**();**

**}**

**}**

ei **+=** e**;**

LCD**.**clear**();**

tt**-=**System**.**currentTimeMillis**()/**1000**;**

double w **=** Kp **\*** **(**e **+** Tv**/**t **\*** **(**e **-** el**));** // Kp \* (e + t/Tn \* ei + Tv/t \* (e - el))

LCD**.**drawString**(**"l:" **+** left **+** " diff="**+**diff**+**"\nr:" **+** right **+** " tt="**+**tt**+**"\ne:" **+** e **+** "\nw:" **+** w **+** "\nTv:" **+** Tv **+** "\nKp:" **+** Kp**,** 0**,** 0**);**

tt**=** System**.**currentTimeMillis**()/**1000**;**

Motor**.**A**.**setSpeed**((**int**)** **(**fspeed**+**w**));**

Motor**.**B**.**setSpeed**((**int**)** **(**fspeed**-**w**));**

**if(**w**>**fspeed**)** **{**

LCD**.**drawString**(**"ERROR" **+** w**,** 30**,** 0**);**

**}**

Motor**.**A**.**forward**();**

Motor**.**B**.**forward**();**

el **=** e**;**

**try** **{**

Thread**.**sleep**((**long**)** **(**t**\***1000**));**

**}** **catch** **(**InterruptedException e1**)** **{**

e1**.**printStackTrace**();**

**}**

**}**

**}**

private static double calibrate**(**LightSensor sensLeft**,** LightSensor sensRight**)** **{**

LCD**.**drawString**(**"Calibrating \n(All white)"**,** 0**,** 0**);**

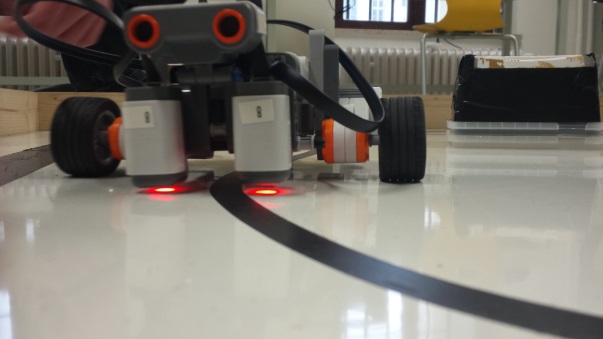
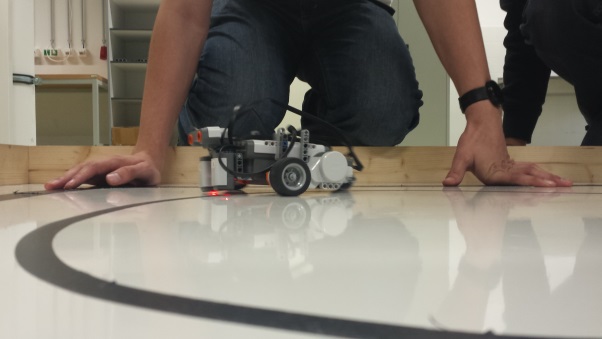
Button**.**waitForAnyPress**();**

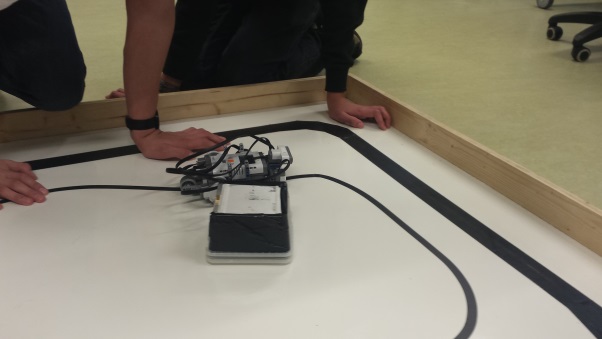
**return** sensLeft**.**getNormalizedLightValue**()-**sensRight**.**getNormalizedLightValue**();**

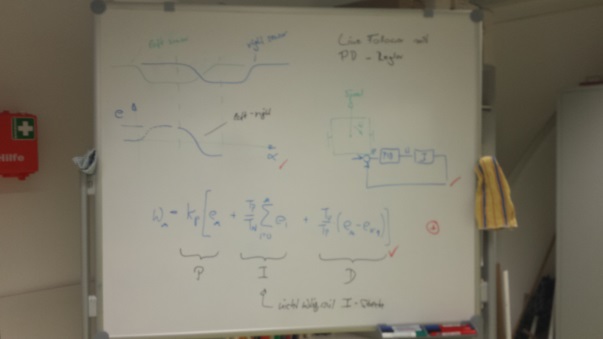
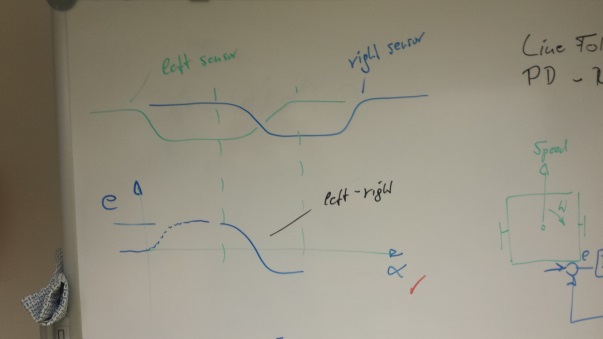
**}**

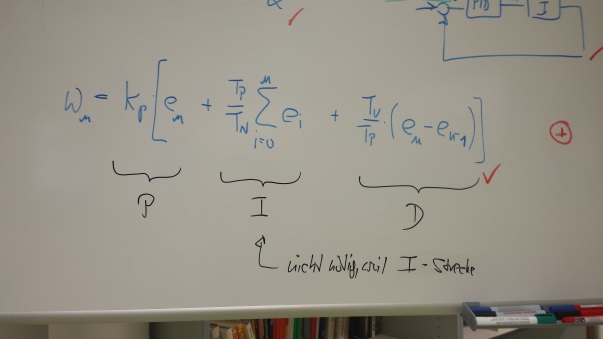
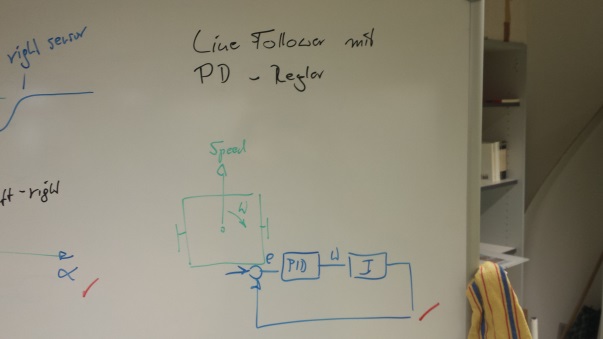
**}**

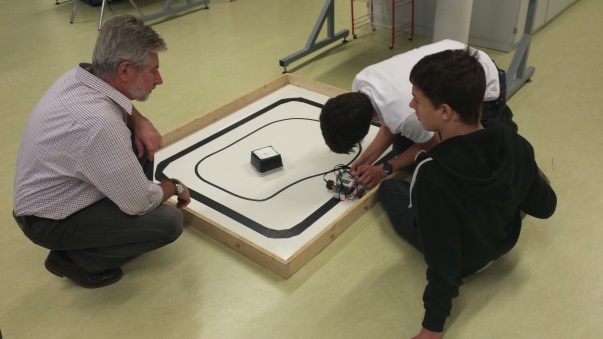
* **Durchgeführte Testläufe**

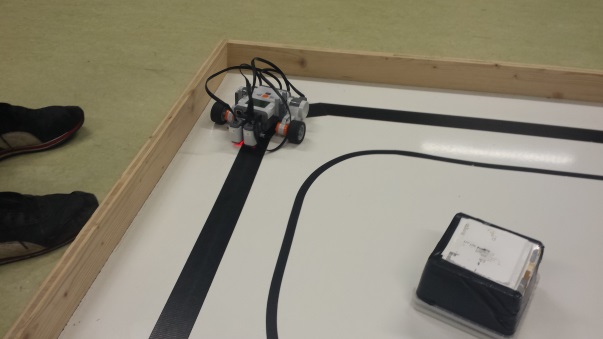
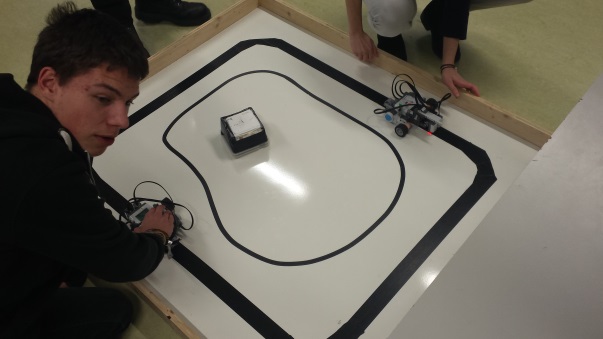


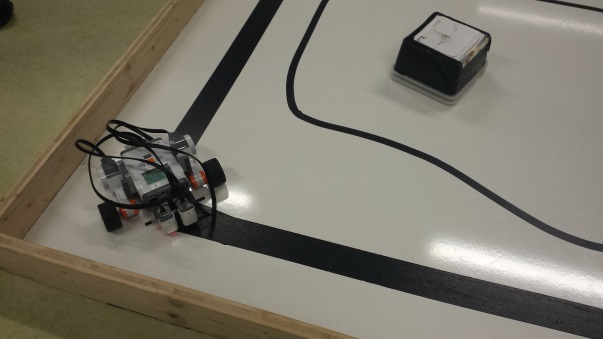


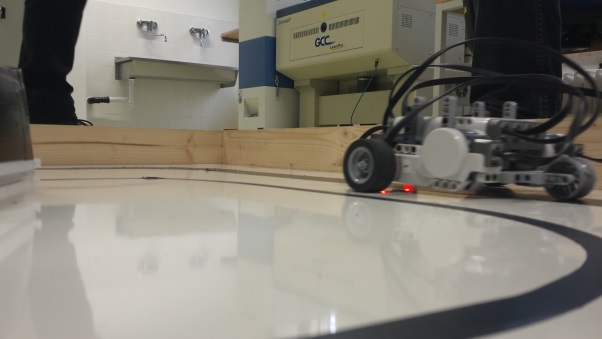


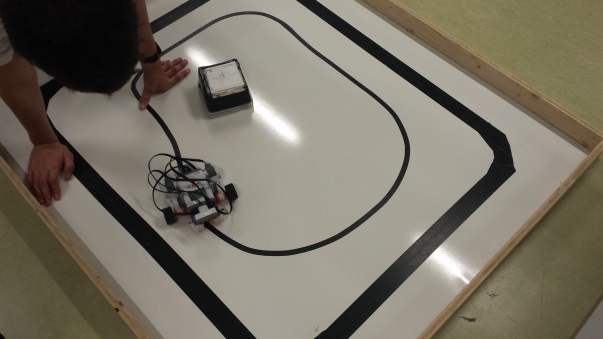
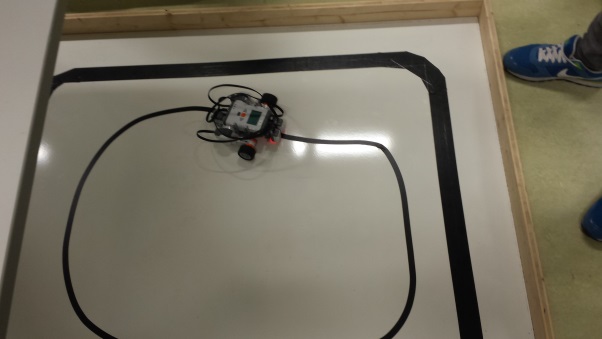












1. Ergebnis

**IFollower.java**

1. Problem + Lösungsvorschläge / Lösung

-

* ANMERKUNG:

Von unserer Arbeit kann man sich auch seit dem 26.11.2014 auf YouTube überzeugen.

Hier der Link zu unserem Video: <https://www.youtube.com/watch?v=OO6mlWeMbU8&feature=youtu.be>