

# SP2007 Teamrobotik

## Lego Mindstorms

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

SS 2007

# Unsere Motivationen

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

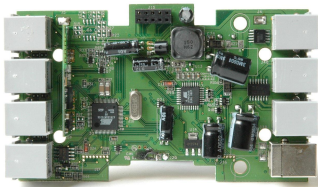
- ▶ Wir sind mit Lego aufgewachsen
- ▶ Interesse für autonome mobile Systeme
- ▶ Projektarbeit im Team

# Technische Spezifikationen des NXT-Brick

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin






- ▶ 32-bit ARM7 microcontroller
- ▶ 256 Kbytes FLASH, 64 Kbytes RAM
- ▶ 8-bit AVR microcontroller
- ▶ 4 Kbytes FLASH, 512 Byte RAM
- ▶ Bluetooth Class II V2.0 kompatibel
- ▶ USB full speed port (12 Mbit/s)
- ▶ 4 Eingänge, 6-adrige Kabel
- ▶ 3 Aufgänge, 6-adrige Kabel
- ▶ 100 x 64 pixel Punktmatrix LC-Anzeige
- ▶ Soundausgabe mit 8-Bit-Auflösung
- ▶ Energieversorgung: 6 AA Batterien



# Übersicht zu Sensoren & Aktoren

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶  2x Berührungssensor
- ▶  1x Ultraschallsensor
- ▶  1x Geräuschsensor
- ▶  1x Lichtsensor
- ▶  3x Servomotor

# Entwicklungsumgebungen

SP2007 Teamrobotik

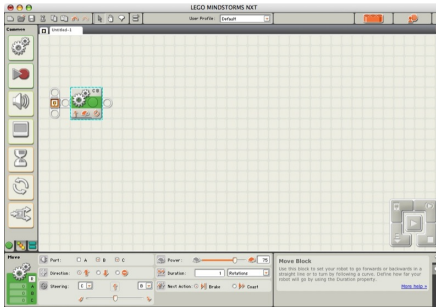
Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ NXT-G
- ▶ Robot C
- ▶ Lejos
- ▶ NXC
- ▶ pbLua

# NXT-G

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ offizielle Entwicklungsumgebung zum Bausatz
- ▶ einfache, graphische Sprache (basiert auf LabView)
- ▶ mit National Instruments entwickelt
- ▶ ausgezeichnet für Programmier-Anfänger geeignet
- ▶ komplexe Programme weniger übersichtlich

# Robot C

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

The screenshot displays the Robot C IDE interface. The main window shows a C program with comments: "The focus of this go and waiting for a signal". A "Debugger" window is open, showing a list of variables and their values. A "Processors Debug" window is also visible, showing "Start" and "Stop" buttons. A "MOTOR CONTROL WINDOW" is open, displaying a table of motor parameters and a "Set Values into NXT" section.

Motor	Speed	Target Pos	Mode	Ring	Step	Type	Mode		
A	0	0	OFF/Pos: 0	none	Idle	0	0	0	0
B	0	0	OFF/Pos: 0	none	Idle	0	0	0	0
C	0	0	OFF/Pos: 0	none	Idle	0	0	0	0

Motor	Speed	Target Pos	Mode	Ring	Step	Type	Mode
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0

Motor	Speed	Target Pos	Mode	Ring	Step	Type	Mode
S1	Raw Value	modeRaw	724	724			
S2	Raw Value	modeRaw	1023	1023			
S3	Raw Value	modeRaw	1023	1023			
S4	Raw Value	modeRaw	1023	1023			

- ▶ weit entwickelt
- ▶ C-Dialekt
- ▶ kommerziell

The screenshot displays the Lejos IDE interface. On the left, there are two panels: 'All Classes' and 'Packages'. The 'All Classes' panel lists various classes such as `AsciiCodec`, `Battery`, `ColorDetectionControl`, `ColorEffect`, `ColorListener`, `Compass`, `CompassNavigator`, `DeviceInfo`, `EvromMessages`, `FileIo`, `FileSystem`, `FirmwareInfo`, `FtpControl`, `FtpEffect`, `Font`, `IRCSensor`, `InputValues`, `Light`, `LightListener`, `MotionDetectionControl`, `MotionDetectionEffect`, and `MotionListener`. The 'Packages' panel shows a tree structure with the following packages: `icommand.nxtcomm`, `icommand.platform.nxt`, `icommand.robotics`, and `icommand.vision`. The main window shows the 'Overview' tab for the selected package, with navigation options like 'PREV', 'NEXT', 'FRAMES', and 'NO FRAMES'.

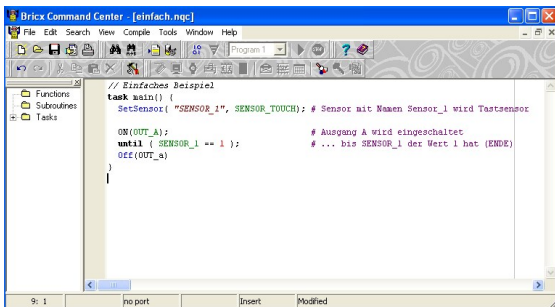
- ▶ Java
- ▶ OpenSource
- ▶ erfordert keinen Mindstorms-Treiber
- ▶ keine eigene IDE
- ▶ Status: Alpha



# NXC (Not eXactly C)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



```
// Einfaches Beispiel
task main() {
  SetSensor( "SENSOR_1", SENSOR_TOUCH); # Sensor mit Namen Sensor_1 wird Tastsensor

  ON(OUT_A);                            # Ausgang A wird eingeschaltet
  until ( SENSOR_1 == 1 );                # ... bis SENSOR_1 der Wert 1 hat (ENDE)
  Off(OUT_a)
}
}
```

- ▶ C-ähnlich
- ▶ IDE für Windows (BricxCC)
- ▶ gute Tutorials
- ▶ mit NBC sehr mächtig
- ▶ OpenSource



- ▶ interpretiert auf NXT, daher kein Compiler nötig
- ▶ gute Bluetooth-Unterstützung
- ▶ OpenSource
- ▶ eigene, schnelle Firmware
- ▶ einsteigerfreundliche Sprache
- ▶ Status: Beta

- ▶ erste Wahl, NXC wegen der guten Umgebung
- ▶ später Umstieg auf Lejos
  - ▶ bessere Manövriermöglichkeiten
  - ▶ Lejos ist strukturierter, Aufteilung in mehrere Klassen möglich

# Infrarot-Ball

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Durchmesser: 7,6cm
- ▶ besitzt 20 Infrarot-Leuchtdioden
- ▶ benötigt zum Betrieb 4 AAA-Batterien
- ▶ Batterielaufzeit:  $\sim$ 1h 30min



- ▶ Inkompatibilitäten mit vielen Adaptern
- ▶ Treiber Updates/Patches könnten hier helfen
- ▶ vor Adapteranschaffung unbedingt informieren
  - ▶ <http://www.vialist.com/users/jgarbers/NXTBluetoothCompatibilityList>

- ▶ Schnittstellen dienen dem Anschluss externer Geräte.
- ▶ Eine Schnittstelle definiert die Festlegung für die physikalischen Eigenschaften der Schnittstellenleitungen.
- ▶ Die Spezifikation einer Schnittstelle enthält Informationen über Übertragungsgeschwindigkeiten, Übertragungsverfahren, zu den Schnittstellenleitungen, dem Stecker, der Buchse oder Steckerleiste und deren Belegung.
- ▶ Sinn einer Spezifikation oder Normierung ist, dass unterschiedliche Geräte verschiedener Hersteller miteinander verbunden werden können.



- ▶ Softwareseitig
  - ▶ `SENSOR_MODE_BOOL` - boolean Wert (0, 1)
  - ▶ `SENSOR_MODE_EDGE` - Anzahl der Flanken
  - ▶ `SENSOR_MODE_PULSE` - Anzahl der Perioden (0-1-0 oder 1-0-1)
  - ▶ `SENSOR_MODE_PERCENT` - Werte von 0% bis 100%
  - ▶ `SENSOR_MODE_RAW` - raw Werte von 0 bis 1023
- ▶ Hardwareseitig
  - ▶ *I<sup>2</sup>C* (Ultraschallsensor)
  - ▶ RCX (Berührungs-, Licht- und Geräuschsensor)



- ▶ Softwareseitig
  - ▶ `SENSOR_MODE_BOOL` - boolean Wert (0, 1)
  - ▶ `SENSOR_MODE_EDGE` - Anzahl der Flanken
  - ▶ `SENSOR_MODE_PULSE` - Anzahl der Perioden (0-1-0 oder 1-0-1)
  - ▶ `SENSOR_MODE_PERCENT` - Werte von 0% bis 100%
  - ▶ `SENSOR_MODE_RAW` - raw Werte von 0 bis 1023
- ▶ Hardwareseitig
  - ▶  $I^2C$  (Ultraschallsensor)
  - ▶ RCX (Berührungs-, Licht- und Geräuschsensor)



# Analog (RCX)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ Wert ergibt sich aus dem Spannungsabfall über der Messschaltung
- ▶ Die Werte liegen im Intervall [0;1023]
- ▶  $D_{RCX} = U * 204,6 \frac{1}{V}$
- ▶  $U = 5V - 10k\Omega * I = \frac{5V}{\frac{10k\Omega}{R} + 1}$
- ▶ 2 Betriebsmodi
  - ▶ passiv  
Messschaltung verhält sich wie ein Widerstand
  - ▶ aktiv  
3 ms liegt Batteriespannung am Sensoreingang an  
0,1 ms wird dann im passiven Modus gemessen

# Analog (RCX)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

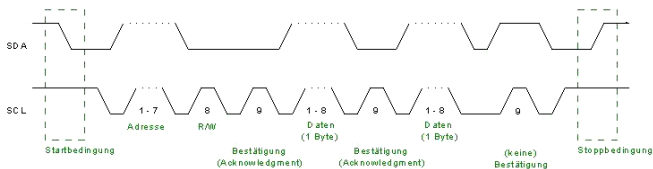
- ▶ Wert ergibt sich aus dem Spannungsabfall über der Messschaltung
- ▶ Die Werte liegen im Intervall [0;1023]
- ▶  $D_{RCX} = U * 204,6 \frac{1}{V}$
- ▶  $U = 5V - 10k\Omega * I = \frac{5V}{\frac{10k\Omega}{R} + 1}$
- ▶ 2 Betriebsmodi
  - ▶ passiv  
Messschaltung verhält sich wie ein Widerstand
  - ▶ aktiv  
3 ms liegt Batteriespannung am Sensoreingang an  
0,1 ms wird dann im passiven Modus gemessen

# Analog (RCX)

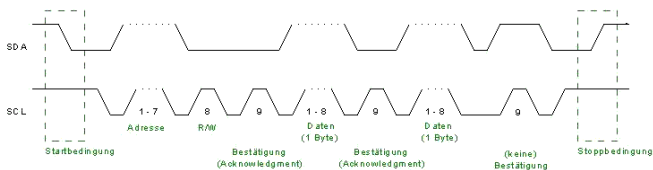
SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ Wert ergibt sich aus dem Spannungsabfall über der Messschaltung
- ▶ Die Werte liegen im Intervall [0;1023]
- ▶  $D_{RCX} = U * 204,6 \frac{1}{V}$
- ▶  $U = 5V - 10k\Omega * I = \frac{5V}{\frac{10k\Omega}{R} + 1}$
- ▶ 2 Betriebsmodi
  - ▶ passiv  
Messschaltung verhält sich wie ein Widerstand
  - ▶ aktiv  
3 ms liegt Batteriespannung am Sensoreingang an  
0,1 ms wird dann im passiven Modus gemessen



- ▶ Inter-Integrated Circuit (Interner Integrierter-Schaltungs-Bus)
- ▶ Von Philips in den 80igern entwickelt
- ▶ serieller, binärer Bus mit Master-Slave-Prinzip
- ▶ kommt mit 2 bidirektionalen Leitungen aus
  1. Versorgungsspannung von 9 Volt
  2. Ground
  3. Ground
  4. Versorgungsspannung (4.3 Volt, 270mA/7=Ø38mA)
  5. I2C-Kommunikation SCL (Serial Clock)
  6. I2C-Kommunikation SDA (Serial Data)



- ▶ Inter-Integrated Circuit (Interner Integrierter-Schaltungs-Bus)
- ▶ Von Philips in den 80igern entwickelt
- ▶ serieller, binärer Bus mit Master-Slave-Prinzip
- ▶ kommt mit 2 bidirektionalen Leitungen aus
  1. Versorgungsspannung von 9 Volt
  2. Ground
  3. Ground
  4. Versorgungsspannung (4.3 Volt,  $270\text{mA}/7 = \varnothing 38\text{mA}$ )
  5. I2C-Kommunikation SCL (Serial Clock)
  6. I2C-Kommunikation SDA (Serial Data)

# Sensor-Messungen

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ Akustiksensoren
- ▶ Lichtsensoren
- ▶ Ultraschallsensoren
- ▶ Winkelabhängigkeit des Ultraschallsensors
- ▶ Langzeittest
- ▶ Motortest

# Versuchsaufbau

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

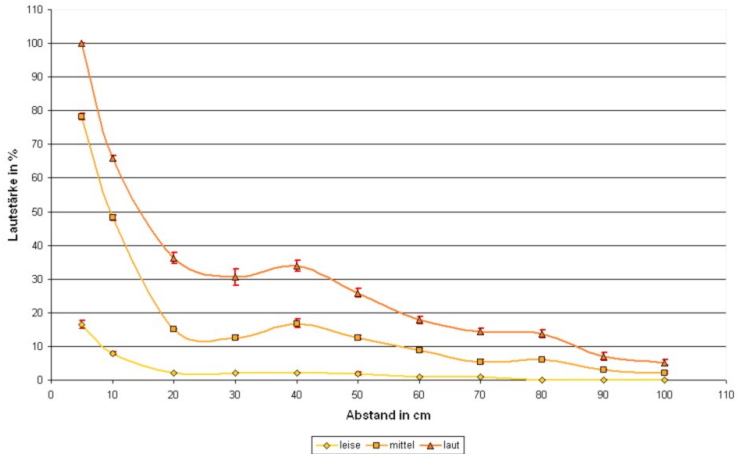


- ▶ Tonquelle (Handy) in diskreten Abständen (von 5-100cm) frontal zum Sensor bewegt
- ▶ gleicher Handyklingelton in 3 Lautstärkestufen
- ▶ Umgebungslautstärke sehr gering ( $\leq 3\%$ )
- ▶ Messungen in %

# Ergebnisse

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Graphen der Mittelwerte von 5 Reihen
- ▶ Standardabweichung als Fehler (rot) dargestellt



# Schlussfolgerungen

SP2007 Teamrobotik

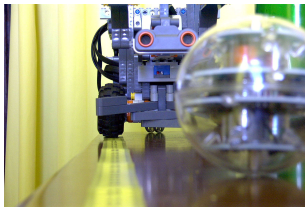
Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ nur bei mittlerer und hoher Lautstärkestufe traten geringe Abweichungen ( $\leq 2\%$ ) zwischen den Messreihen auf
- ▶ ebenfalls bei mittlerer und hoher Stufe waren die Messwerte bei 40cm Abstand insgesamt höher als bei 30cm
- ▶ mit dieser Ausnahme sind die Kurven konvex fallend

# Versuchsaufbau

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

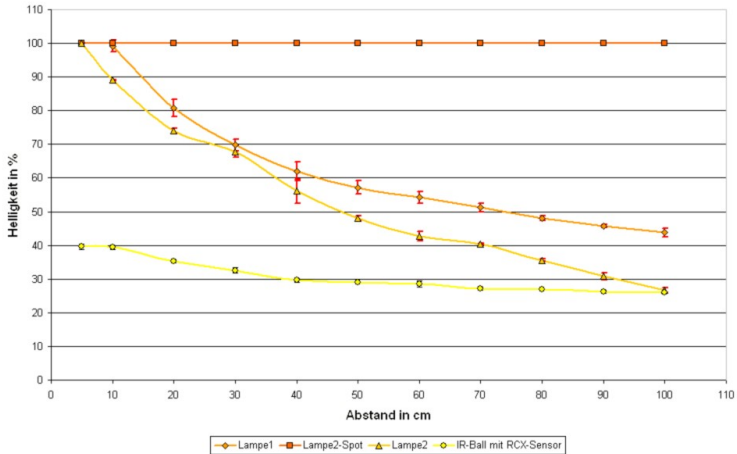


- ▶ Lichtquelle in diskreten Abständen (von 5-100cm) frontal auf den Sensor gerichtet
  1. schwache ausgeteilte Taschenlampe (Umgebungslicht: 30-37%)
  2. eigene starke Taschenlampe, gebündelt (Umgebungslicht: 1,5%)
  3. eigene starke Taschenlampe, ungebündelt (Umgebungslicht: 1,5%)
  4. Infrarot-Ball mit RCX-Sensor (Umgebungslicht: 20-23%)
- ▶ Messungen beim 1. & 4. Test in %, bei Test 2 & 3 raw-Daten (0-1023)

# Ergebnisse

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Graphen der Mittelwerte von 5 Reihen
- ▶ Standardabweichung als Fehler (rot) dargestellt

- ▶ bei ausgeteilter Taschenlampe gab es Schwierigkeiten konstante Werte zu erhalten, Ursache: Wackelkontakt
- ▶ in der 5. Messreihe des 1. Tests schwächelten die Batterien
- ▶ im 3. Test waren die Messwerte bei 30cm Abstand tendenziell etwas zu hoch
- ▶ Messwerte für den Ball hängen stark von seiner Ausrichtung ab

# Schlussfolgerungen

SP2007 Teamrobotik

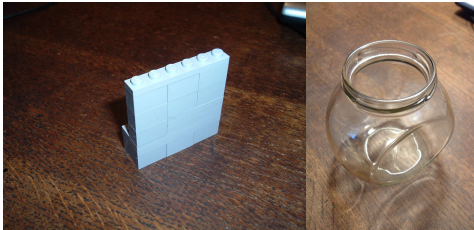
Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ Der NXT-Lichtsensordatensatz ist sensibler
- ▶ Der RCX-Sensordatensatz ist unter 0,5m Entfernung zur Ballsuche geeignet
- ▶ Die Lichtsensoren sind nur bedingt zur Entfernungsbestimmung geeignet

# Versuchsaufbau

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

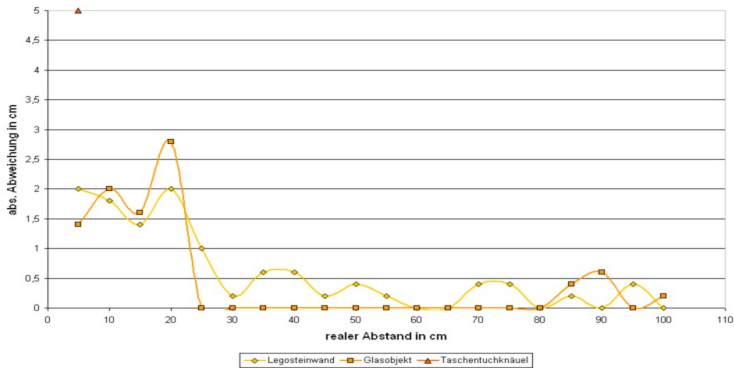


- ▶ Gegenstände (Legosteinwand, Glasobjekt und Taschentuchknäuel) in diskreten Abständen (von 5-100cm) in frontaler Sicht zum Sensor bewegt
- ▶ Messungen in cm

# Ergebnisse

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Graphen zeigen Abweichungen der Mittelwerte der 5 Messreihen zum realen Abstand
- ▶ Standardabweichung bei Legosteine max. 0,5cm, bei Glasobjekt  $< 0,5$ cm
- ▶ Taschentuchknäuel wurde nicht erkannt

# Schlussfolgerungen

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

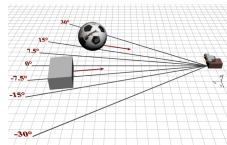
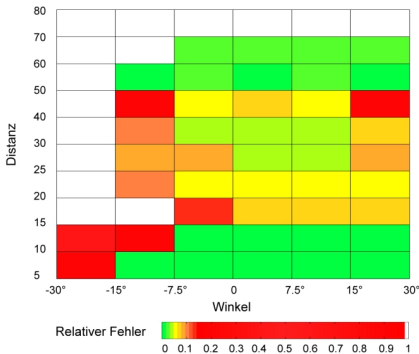
- ▶ wenn es zu abweichenden Werten kam, waren diese immer positiv
- ▶ größere Abweichungen (1,5-3,0cm) traten bei Abständen von 5-20cm auf
- ▶ ab 20cm Abstand waren die Werte zum Glasobjekt sehr präzise; bei der Steinwand gab es im Mittel nur Abweichungen bis 0,5cm
- ▶ das Taschentuchknäuel scheint die US-Wellen zu schlucken



# Winkelabhängigkeit des Ultraschallsensor

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



*"Die besten Resultate erhält man bei Distanzen kleiner als 20 cm und Winkeln kleiner als 5°."*<sup>1</sup>

<sup>1</sup>Quelle: [http://www.tik.ee.ethz.ch/~andreame/tik/public/sada/sa2006\\_claudia\\_thomas.pdf](http://www.tik.ee.ethz.ch/~andreame/tik/public/sada/sa2006_claudia_thomas.pdf)

# Schlussfolgerungen

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ bei schräger Sicht des Sensors zum Objekt kommt es z.T. zu sehr großen Abweichungen
- ▶ tendenziell links größere Abweichung, da rechts am Sensor der Lautsprecher sitzt und somit links befindliche Objekte im Wellenschatten liegen

# Versuchsaufbau

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

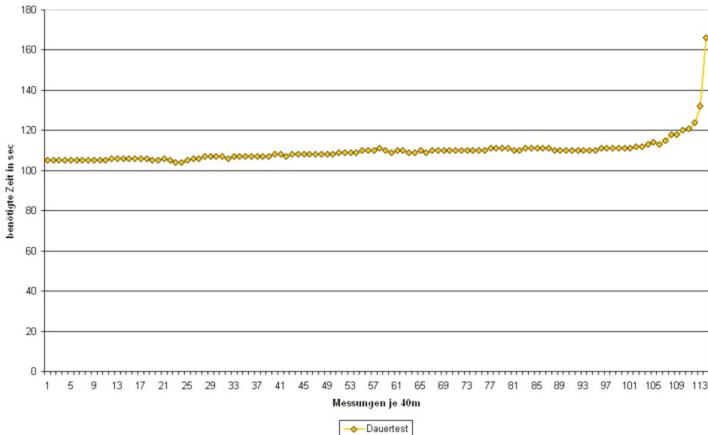


- ▶ Achse des Roboters versteift, sodass sich Motoren nur noch in gleiche Richtung mit gleicher Geschwindigkeit drehen
- ▶ Vorne und Hinten Touchsensoren montiert
- ▶ Bücherstapel in Abstand von  $1\text{m} + \text{Länge Robot}$
- ▶ Roboter fährt 20 mal die Strecke vor und zurück nach Touchsensorberührung

# Ausdauerertest

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

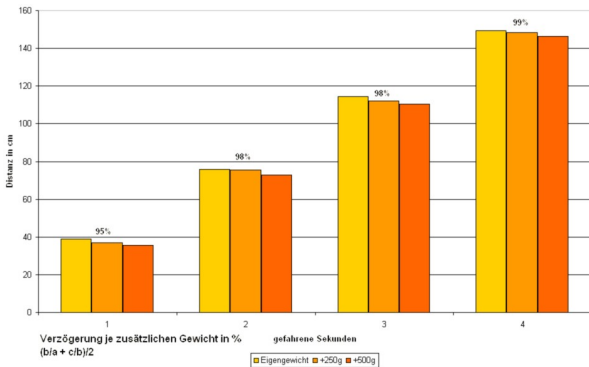


- ▶ Gefahrene Strecke: 4532m
- ▶ Benötigte Zeit: 3h 26min 29sec

# Motortest (100% Power)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

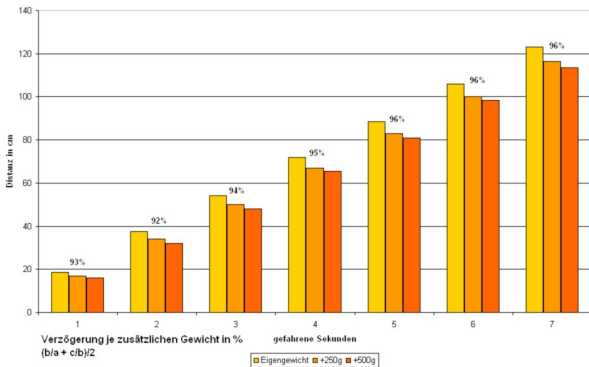


- ▶ Gemessen wurde die gefahrene Strecke
- ▶ Die Fahrzeiten betragen 1,2,3,4,5,6,7 Sekunden
- ▶ Belastung: 250g, 500g (Eigengewicht: 500g)

# Motortest (50% Power)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

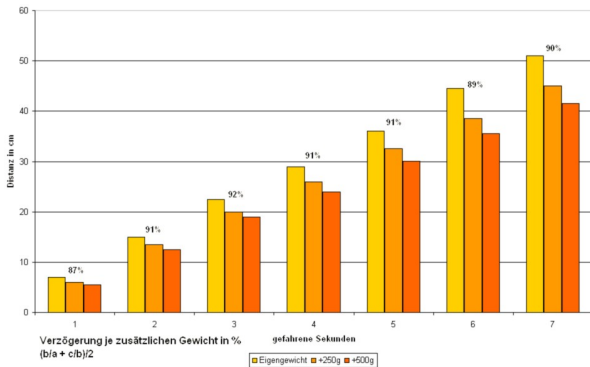


- ▶ Gemessen wurde die gefahrene Strecke
- ▶ Die Fahrzeiten betragen 1,2,3,4,5,6,7 Sekunden
- ▶ Belastung: 250g, 500g (Eigengewicht: 500g)

# Motortest (25% Power)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Gemessen wurde die gefahrene Strecke
- ▶ Die Fahrzeiten betragen 1,2,3,4,5,6,7 Sekunden
- ▶ Belastung: 250g, 500g (Eigengewicht: 500g)

- ▶ Der Ultraschallsensor sieht keine weichen Gegenstände
- ▶ Es bereitet dem Ultraschallsensor Probleme, wenn man schräg auf eine Wand zufährt
- ▶ Die Akkulaufzeit ist ordentlich
- ▶ Die Geradeausfahrt auf langen Strecken könnte Probleme machen
- ▶ Den Geräuschsensor immer weit genug weg von den Motoren anbringen



# Anforderungen

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

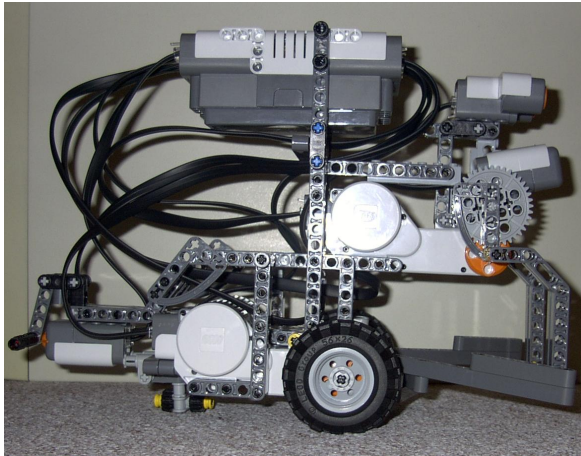
- ▶ Verwendung der gegebenen Ressourcen
- ▶ Sehen des Infrarot-Balls
- ▶ Greifen des Balls
- ▶ Sehen der Taschenlampe
- ▶ Ausfindigmachen von Hindernissen

- ▶ Anzahl der verfügbaren Bausteine ist sehr gering
- ▶ Taschenlampe und Ball müssen gesehen werden
- ▶ Hindernisse (Rand) nur minimal größer als Ball

# Lösung (Version 1)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Lichtsensor richtet sich beim Greifen neu aus
- ▶ Roboter ist zu hoch
- ▶ Ultraschallsensor ist zu hoch

# Lösung (Version 2)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

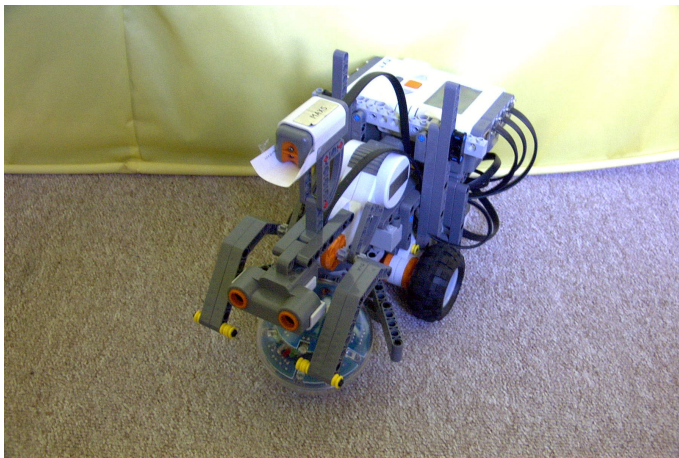


- ▶ Schwerpunkt liegt besser
- ▶ Ultraschallsensor ist noch zu hoch

# Lösung (Version 3)

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin



- ▶ Ultraschallsensor nur noch minimal über dem Ball
- ▶ getrennte Lichtsensoren für Ball und Taschenlampe
- ▶ etwas kürzeres Heck als Version 2

# Ball zur Taschenlampe

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ Ausfindigmachen des Balls durch Rotation
- ▶ Ball aufnehmen (mit Kurskorrekturen)
- ▶ Ausfindigmachen der Taschenlampe
- ▶ auf Lichtquelle zusteuern
- ▶ wenn bestimmter Wert überschritten ist, Ball schießen

- ▶ Klasse: `lejos.navigation.Pilot`
  - ▶ Steuerung von Robotern mit 2 Antriebsmotoren
- ▶ Konstruktor:
  - ▶ `Pilot(float Raddurchmesser, float Spurbreite, Motor linkerMotor, Motor rechterMotor)`
- ▶ Methoden:
  - ▶ `void resetTachoCount()`
    - ▶ Zurücksetzen der internen Tachos der beiden Motoren
  - ▶ `void rotate(phi)`
    - ▶ rotiert Roboter um phi Grad im Uhrzeigersinn
  - ▶ `int getAngle()`
    - ▶ Ausrichtung des Roboters in Grad (bezogen auf letzten Tacho-Reset)
  - ▶ `boolean isMoving()`
    - ▶ bewegt sich der Roboter?

# Code-Auszug: Lichtsuche

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

```
1  Pilot driver = new Pilot(5.6f,10.7f,Motor.A, Motor.B,false);
2  while (!grabber.hasBall()) {
3      if(volleDrehung){
4          ausrichtung = 0;
5          lichtstaerke = 0;
6          ballWeitWeg = true;
7          driver.resetTachoCount();
8          driver.setSpeed(150);
9          driver.rotate(360, true);
10         while (driver.isMoving()) {
11             temp_lichtstaerke = balleye.readValue();
12             if (temp_lichtstaerke > lichtstaerke) {
13                 ausrichtung = driver.getAngle();
14                 lichtstaerke = temp_lichtstaerke;
15             }
16             try {Thread.sleep(50);}
17             catch (InterruptedException ie) {}
18         }
19
20         if (lichtstaerke > maxBall) {
21             ballWeitWeg = false;
22         }
23
24         while (ausrichtung > 180) {
25             ausrichtung -= 360;
26         }
27
28         driver.rotate((int) ausrichtung, false);
29         volleDrehung = false;
30     }
```

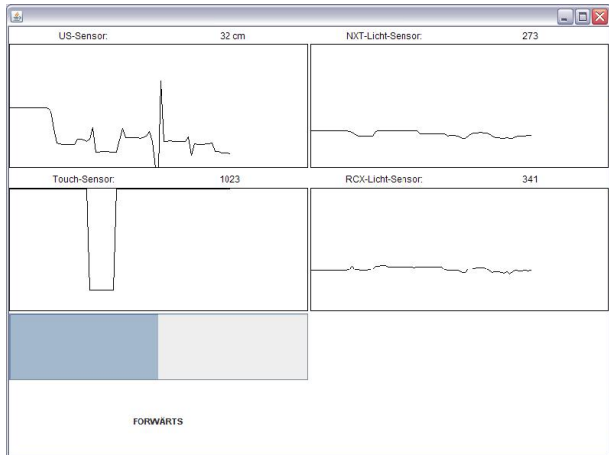


# Aufgabe

SP2007 Teamrobotik

Daniel Meyer,  
Nils Müller &  
Felix Penzlin

- ▶ Messwerte sollen während der Laufzeit mit Bluetooth übertragen werden
- ▶ Die Messwerte sind auf dem PC zu visualisieren
- ▶ Fernsteuerung via Bluetooth mit Laptop



- ▶ verwendete Bibliothek: iCommand
- ▶ Visualisierung erfolgte mit Java
  - ▶ Die Ausgabe von Sensordaten erfolgt als Graph
  - ▶ Der Roboter kann über die Tastatur gesteuert werden

- ▶ Semesterarbeit 2006.18, C. Frischknecht & T. Other

- ▶ LeJOS
- ▶ iCommand
- ▶ BricxCC
- ▶ NBC