

Fachhochschule Wels

FUP

Titel

LINEFOLLOWER CHALLENGER & NAUTILUS

Protokoll

Betreuer: Ing. Michael Zauner BSc

Übungsteilnehmer:

Name	Matr. Nr
Thumfart Stefan	S0910438054
Haas Dominik	S0910438034
Affenzeller Christoph	S0910438022
Parzer Patrick	S0910438079

abgegeben am: 28.06.2011

Inhaltsverzeichnis

1	Aufgabenstellung und Anforderungen an den Roboter.....	3
1.1	Slalom.....	3
1.2	Slalom-Enhanced.....	5
2	Mechanischer Aufbau.....	7
2.1	Antrieb.....	7
2.2	Platine als Chassis.....	8
2.3	Räder.....	8
2.4	Lenkung.....	9
2.5	Sensoren.....	9
2.6	Display.....	11
2.7	Akku.....	11
3	Elektronischer Aufbau.....	12
3.1	Schaltplan.....	12
3.1.1	Prozessoreinheit.....	13
3.1.2	Bodensensoren.....	13
3.1.3	Entfernungsmesser.....	13
3.1.4	Stromversorgung.....	14
3.1.5	Programmierschnittstelle.....	15
3.1.6	Display.....	16
3.2	Layout.....	17
3.2.1	Bestückung.....	17
3.2.2	Leiterbahnen.....	17
4	Software.....	19
4.1	Display.....	19
4.1.1	Displaybeschreibung.....	19
4.2	Slalom.....	20
4.2.1	Steuerung und Lenkung.....	20
4.2.2	Lösung der Steuerung und Lenkung.....	21
4.3	Enhanced Mode.....	23
4.3.1	Steuerung und Lenkung.....	23
4.3.2	Lösung der Steuerung und Lenkung.....	24
5	Verbesserungsmöglichkeiten.....	25
5.1	Bodensensoren.....	25
5.2	Robotergröße.....	25
5.3	Modulbauweise der Abstandssensoren.....	26
5.4	Fixer Akkuplatz.....	26
6	Gegnerisches Wettkampfkonzzept.....	26

1 Aufgabenstellung und Anforderungen an den Roboter

1.1 Slalom

- **Aufgabenstellung:**

Ziel ist es einen, durch eine schwarze Linie gekennzeichneten Kurs so schnell wie möglich zu bewältigen. Dabei müssen bestimmte Regeln eingehalten werden und der Roboter muss dem Reglement entsprechen.

- **Reglement:**

- **Größen und Gewichtsbeschränkungen:**

Die Größe des Roboters ist auf „30cm in der Breite“ beschränkt. „Für Höhe und Länge des Roboters sind keine Regeln vorhanden. Das Gewicht des Roboters kann von den Konstrukteuren ebenfalls beliebig gewählt werden.“

(robotchallenge.org, 2011)

- **Zeitmessung und Zeitlimits:**

„Die Zeitmessung erfolgt ab dem Startsignal bis zum Überqueren der Ziellinie mittels elektronischer Zeitmessung. Wobei die Zeitmessung beim Überqueren der Ziellinie mit dem vordersten Teil des Roboters auslöst.“

„Die Strecke muss vom Roboter innerhalb von 3 Minuten bewältigt werden. Schafft es ein Roboter nicht, den Kurs innerhalb von 3 Minuten zu bewältigen, wird dieser qualifiziert.“

(robotchallenge.org, 2011)

- **Autonome Kontrolle:**

„Sobald ein Roboter die Startlinie überquert hat muss dieser die restliche Strecke vollständig autonom bewältigen, sonst wird dieser disqualifiziert. Ein Roboter der die Arena verlässt wird ebenfalls disqualifiziert. Dies gilt auch, wenn sich Teile des Roboters lösen und die Arena verlassen.“

Wenn ein Roboter die Linie verlässt muss dieser an genau der Stelle, oder früher im Kurs den Lauf wieder fortsetzen.“

(robotchallenge.org, 2011)

○ **Kursanforderungen:**

„Der Kurs ist auf einem weißen Untergrund aufgeklebt (oder gedruckt). Die Linie ist schwarz und ca. 15mm breit. Es ist ein Start- und Endbereich vorgesehen, der durch eine quer verlaufende Linie extra gekennzeichnet wird (diese Linie hat eine Aussparung von ca. 10cm, durch die die eigentliche Strecke verläuft)“.

▪ Charakterisierung des Kurses:

- Der Kurs soll keine Überkreuzungen der Strecke beinhalten
- Haarnadeln oder Spitzkehren sind möglich, doch der Abstand der angrenzenden Linien ist nicht geringer als 15 cm (von der Mitte der Linien gemessen).
- Der Abstand von der Linie zum Rand der Arena ist nicht geringer als 15 cm (gemessen von der Mitte der Linie)
- Der kleinste Kurvenradius beträgt 15 cm

(robotchallenge.org, 2011)



Abbildung 1: Kursverlauf (2011)

- **Anforderungen:**

Der Linienverfolger muss eine schwarze Linie detektieren und diese verfolgen können. Realisiert wurde dies mittels sieben Licht-Sensoren an der Unterseite des Roboters, die Lichtreflexionen detektieren können. Je nach dem welcher Sensor anschlägt, werden die Motoren verschieden angesteuert, um die Abweichung der Linie zum mittleren Sensor ausgleichen zu können.

1.2 Slalom-Enhanced

- **Aufgabenstellung:**

Die grundsätzliche Aufgabenstellung und das Reglement sind identisch mit dem des Slaloms. Bei diesem Bewerb müssen zusätzlich 3 verschiedene Hindernisse bewältigt werden. Die in den Parcours eingebauten Hindernisse sind (siehe Abb. 5):

- eine Unterbrechung der Linie
- ein Tunnel
- ein Ziegelstein

- **Reglement:**

- **Tunnel:**

Die Linie führt durch einen „30cm hohen und 30cm breiten“ Tunnel (siehe Abb. 2). Der Roboter muss die Linie weiter verfolgen und mit den wechselnden Lichtverhältnissen zu Recht kommen.

(robotchallenge.org, 2011)



Abbildung 2: Tunnel (2011)

- **Unterbrechung:**

„An einer Stelle des Kurses ist die Linie für 10cm unterbrochen. Die Unterbrechung befindet sich nicht in einer Kurve.“(siehe Abb. 3)

(robotchallenge.org, 2011)



Abbildung 3: Linienunterbrechung

- **Ziegelstein:**

„An einer beliebigen Stelle des Kurses befindet sich ein Ziegelstein (ca. 25 x 12 x 6,5 cm LxBxH). Der Roboter muss diesen Stein umfahren, um den Kurs weiterverfolgen zu können.“(siehe Abb. 4)

(robotchallenge.org, 2011)

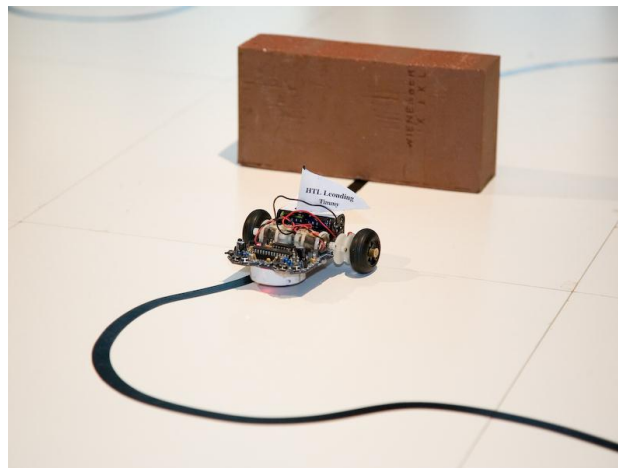


Abbildung 4: Ziegelstein



Abbildung 5: Streckenverlauf und Aufbau der Hindernisse (Slalom Enhanced)

- **Anforderungen:**

Für den Enhanced-Bewerb muss der Roboter zusätzlich die Hindernisse überwinden können, wobei für den Ziegelstein zusätzlich ein Abstandssensor nötig ist. Wenn dieser anschlägt, verfolgt der Roboter einen einprogrammierten Weg, um den Ziegelstein zu umfahren. Die Unterbrechung der Linie und der Tunnel können durch Programmierung, ohne weitere Sensoren bewältigt werden.

2 Mechanischer Aufbau

In diesem Kapitel werden der mechanische Aufbau, sowie eventuelle mechanische Verbesserungen behandelt. Insbesondere wird auf das Lenkungs- und Antriebskonzept, sowie auf die Leichtbauweise eingegangen.

2.1 Antrieb

Bei der Konstruktion und Auslegung des Antriebs waren folgende Dinge zu beachten:

- Reifen, für eine möglichst hohe Haftreibung
- Motoren genau in Achse ausrichten

Der Antrieb erfolgt über zwei auf einer Achse liegenden Faulhaber Motoren mit der Bezeichnung 1717006SR (siehe Abb.5). Als Untersetzungsgetriebe für den Motor kam das Getriebe 16/7, mit einer Übersetzung von 3,71:1 zum Einsatz.

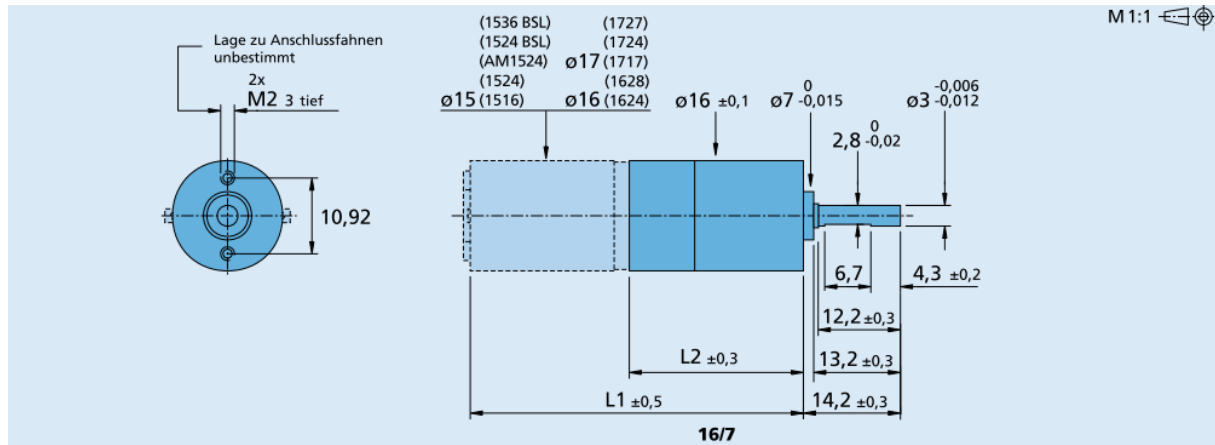


Abbildung 6: Motor: Faulhaber 1717006SR

2.2 Platine als Chassis

Um Gewicht einzusparen und den grundsätzlichen Aufbau des Roboters so einfach wie möglich zu halten wurden alle Komponenten direkt auf der Platine montiert. Somit konnte ein sehr leichter, Roboter mit einer großen Beschleunigung aufgebaut werden.

2.3 Räder

Bei den Rädern wurden zwei verschiedene Konzepte eingesetzt. Einerseits wurden professionelle Rennreifen für Glattbahnen verwendet, die sich durch eine sehr gute Haftung und dadurch möglichst hohe Kurvengeschwindigkeiten auszeichnen. Zusätzlich sind diese Reifen sehr leicht und es können keine Probleme mit z.B. unwuchten Reifen auftreten.



Abbildung 7: Räder (1:10)

Als zweites Konzept wurden sehr gut haftende Gekko-Pads verwendet. Diese wurden auf Felgen aufgeklebt, wodurch sich allerdings eine kleine Unwucht am Stoß der Enden ergibt. Gekko-Pads bestehen aus einem speziellen Silikon und sind eigentlich für Ablageflächen von Handys etc. in Autos konzipiert, um diese am wegrutschen zu hindern. Nachteil bei dieser Art von Reifen ist, dass sich der Staub von der Strecke sehr stark auf den Pads ablagert und diese somit mit der Zeit an Haftung verlieren.

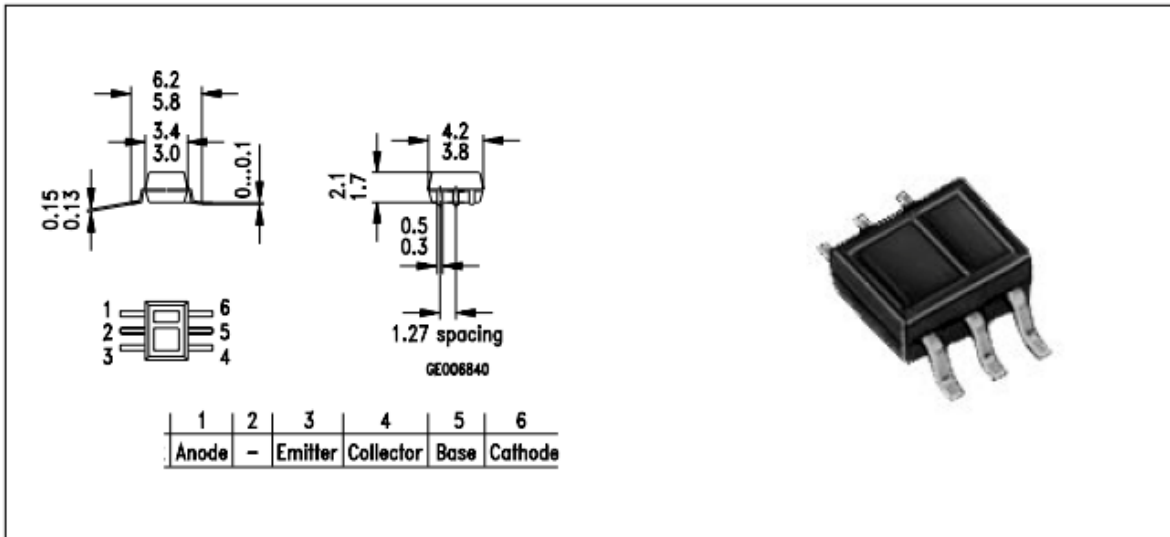
2.4 Lenkung

Die Lenkung des Roboters wurde durch die unterschiedliche Ansteuerung der Motoren realisiert. Über die Drehzahl der einzelnen Motoren konnte somit der Lenkeinschlag eingestellt werden.

2.5 Sensoren

- **Bodensensoren:**

Die Bodensensoren dienen zum Erkennen der Linie. Je nachdem, welcher Sensor anspricht, werden die Motoren dementsprechend angesteuert. Als Bodensensoren wurden sieben Infrarot-Reflexlichtschranken, vom Typ SPH9102 der Firma Sharp verwendet, die an der Unterseite der Platine angebracht wurden.



Maße in mm, wenn nicht anders angegeben/Dimensions in mm, unless otherwise specified.

Abbildung 8: Bodensensoren SPH9102

- **Abstandssensoren:**

Abstandssensoren werden nur für den Slalom-Enhanced Bewerb zum Erkennen des Ziegelsteins benötigt. Es wurden Infrarotsensoren vom Typ GP2Y0D345K mit einer Reichweite von ca. 45cm verwendet, um den Ziegelstein möglichst bald erkennen zu können und diesen, ohne die Geschwindigkeit stark zu vermindern, umfahren zu können.



Abbildung 9: Abstandssensoren

2.6 Display

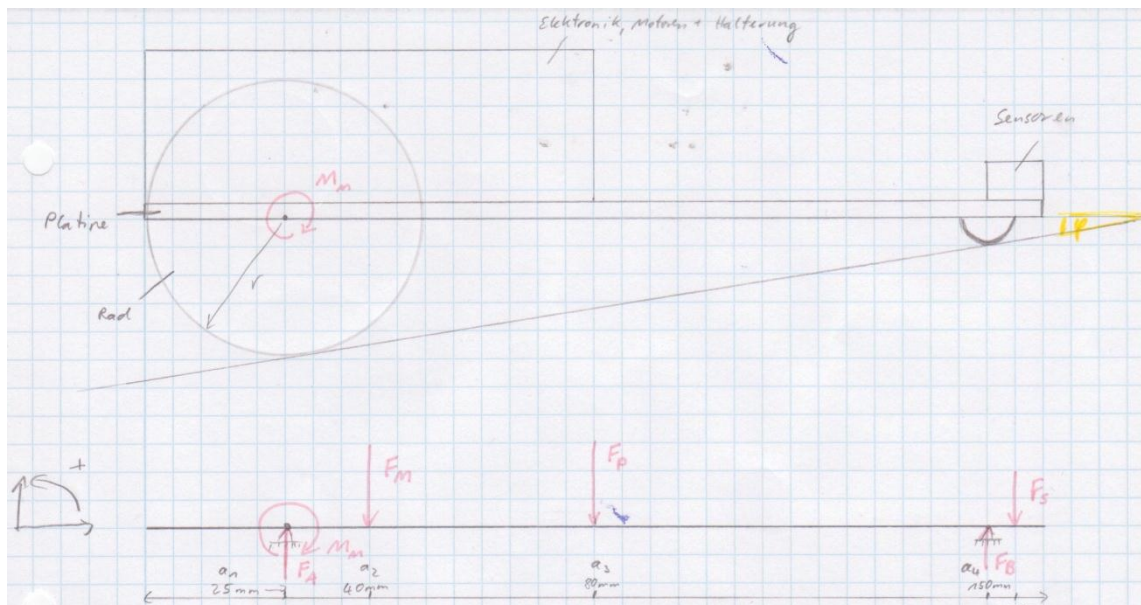
Das Display wurde mittels Steckverbindung auf der Platine zwischen den Motoren angebracht, und dient zur Einstellung des gewünschten Programmes bzw. zur direkten Veränderung von Parametern.

2.7 Akku

Als Akku wurde ein Lithium-Ionen Akku mit 7,4V und 800mA Stunden verwendet.

2.8 Berechnungen

Bauteile
Motor + Getriebe (3,71/1)
Motorhalterungen
Platine (inkl. Bauteile) (hinterer Teil)
Sensorboard (inkl. Bauteile und Sensoren)
Akku



Auflagerreaktionen:

verwendete Formelzeichen:

F_A ...Lagerkraft Reifen

F_B ...Lagerkraft Diode

F_M ...Gewichtskraft Motoren + Halterungen

F_P ...Gewichtskraft Platine

F_S ...Gewichtskraft Sensorboard

F_{Ak} ...Gewichtskraft Akku

a_1, a_2, a_3, a_4, a_5 ...Abstände zu den Bauteilen (siehe Abb.:)
 M_M ...Motordrehmoment (=0mNm lt. Datenblatt)

$$\sum F_{ix} = 0$$

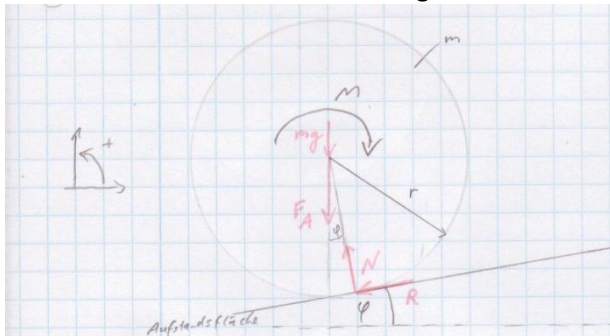
$$\sum F_{iy} = 0 : F_A - F_M - F_P + F_B - F_{Ak} - F_S = 0$$

$$\sum M_A = 0 : -2 * M_M - F_M * a_1 - F_P * a_2 + F_B * a_3 - F_{Ak} * a_4 - F_S * a_5$$

$$\rightarrow F_B = \frac{2 * M_M + F_M * a_1 + F_P * a_2 + F_{Ak} * a_4 + F_S * a_5}{a_3}$$

$$\rightarrow F_A = F_M + F_P - F_B + F_S$$

bei Annahme von reinem Rollen ergibt sich als maximale Beschleunigung ($F_B = 0$ N):



$$F_{ges} = m_{Reifen} * g + F_A$$

$$m * \ddot{y} = -F_{ges} + N \rightarrow N = F_{ges}$$

Annahme: $R = \mu_H * N = N$

$$m * \ddot{x} = -R$$

$$I_M * \ddot{\varphi} = -M + R * r$$

$$\frac{m * r^2}{2} * \ddot{\varphi} = -R * r + M$$

$$\ddot{\varphi} = \frac{2 * (M - R * r)}{m * r^2}$$

$$\ddot{x} = \frac{2 * (M - R * r)}{m * r}$$

3 Elektronischer Aufbau

3.1 Schaltplan

Zur Schaltplanentwicklung wurden die bereits an der FH-Wels vorhandenen Schaltpläne des modularen elektronischen Systems verwendet und zusammengeführt.

3.1.1 Prozessoreinheit

Als Prozessoreinheit wurde der Microcontroller ATmega 644 von Atmel verwendet.

3.1.2 Bodensensoren

Damit der Mikrocontroller die analogen Ausgangssignale bearbeiten kann muss sichergestellt werden, dass bei weißem Hintergrund die Spannung am Open-Kollektor-Ausgang des Sensors einen Wert von 0,4 V nicht übersteigt.

Befindet sich der Sensor über der Schwarzen Linie, darf derselbe Ausgang 2,4 V nicht unterschreiten, um die TTL-Logikpegel für den Microcontroller herzustellen. Weiters muss die Diode im Bauteil mit einem Vorwiderstand gegen Überstrom gesichert werden (siehe Abb.10).

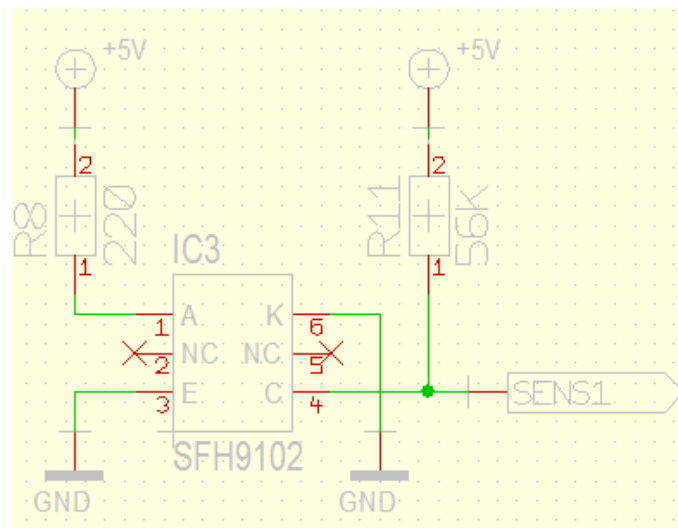


Abbildung 10: Beschaltung Bodensensoren

3.1.3 Entfernungsmesser

Der IR-Entfernungsmesser besteht, ähnlich wie die Bodensensoren, aus einer IR-Diode und einem Fototransistor. Jedoch hat dieser Baustein einige zusätzliche Funktionen z.B.: hat er eine Oszillatoreinheit als Spannungsstabilisierung welche einen Fremdlichtunabhängigen Betrieb ermöglicht. Zudem besitzt dieser Baustein eine Signalverarbeitungseinheit, welche digitale Signale ausgeben kann. Die Schaltschwellen, der Signale werden durch eine externe digitale Beschaltung festgelegt.

Somit ergibt sich die Beschaltung in Abb. 11. Der Kondensator C8 dient als Stützkondensator. Der Kondensator C7 und der Widerstand R31 sind laut Datenblatt zu beschalten. Der Widerstand R31 dient als Vorwiderstand für die Infrarotdiode. C7 dient als Stabilisierungskondensator.

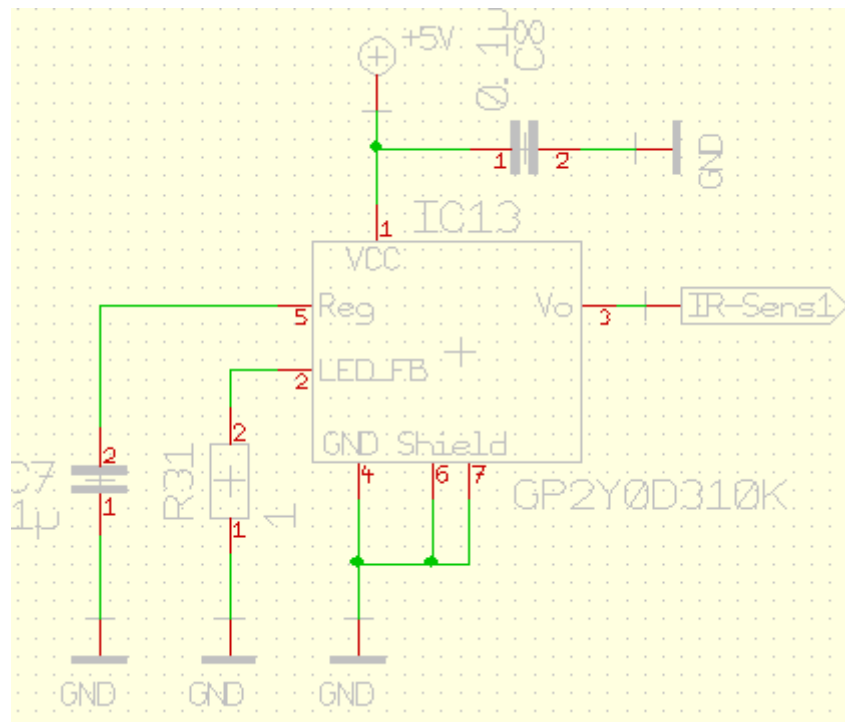


Abbildung 11: Beschaltung IR-Entfernungsmesser

3.1.4 Stromversorgung

Für die Stromversorgung wurde ein Lithium-Ionen-Akku mit 7,4V und 800mA Stunden verwendet. Da diese Spannung jedoch nur zur Ansteuerung der Motoren verwendet werden kann, wird für den Steuerteil eine Spannungsstabilisierung eingebaut. Dazu wird der Analogvergleicher LM2594 verwendet. Durch den Baustein wird ein Puls-Breiten-Moduliertes Rechtecksignal (Schaltnetzteil) erzeugt. L1 und C6 Glätten dieses Signal zu einem stabilen Gleichspannungssignal. Die Diode D1 dient als Freilaufdiode, um Überspannungen beim Schalten des Signals „Out“ zu verhindern. Über den Spannungsteiler R9 und R10 wird ein Referenzsignal für die Puls-Breiten-Modulation erzeugt.

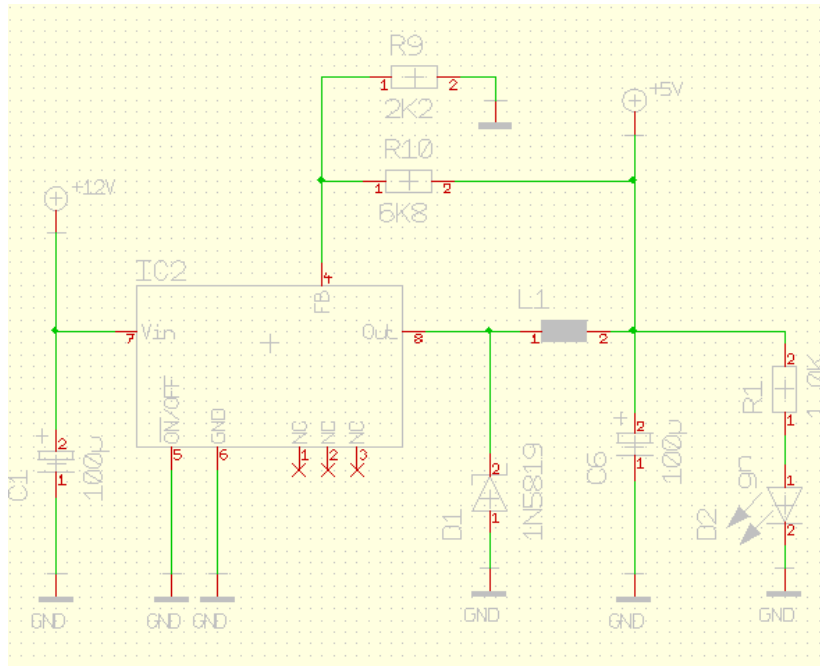


Abbildung 12: Stromversorgung LM2594

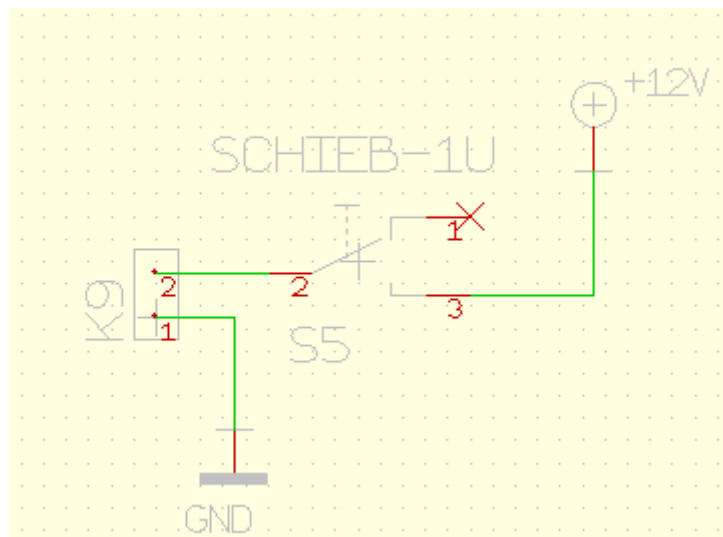


Abbildung 13: Hauptschalter

3.1.5 Programmierschnittstelle

Um den Microcontroller ATMEGA 644 programmieren zu können, wurde die AVRISP Schnittstelle verwendet. Um diese Schnittstelle einbinden zu können musste lediglich ein Datenblatt der Schnittstelle zugezogen werden und die Kontakte dementsprechend mit dem Controller verbunden werden.

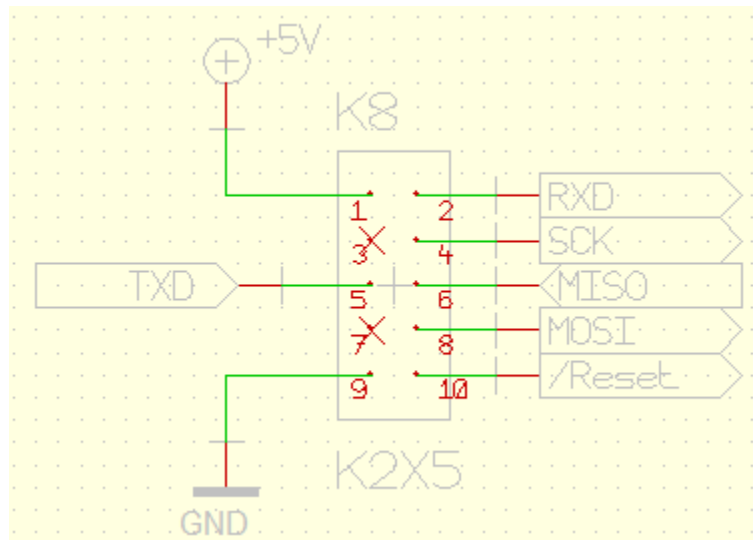


Abbildung 14: Programmierschnittstelle

3.1.6 Display

Auch in diesem Fall ist das Datenblatt des Displays unerlässlich. Das Trimm-Potentiometer R5 (siehe Abb. 15) dient zur Einstellung des Kontrasts des Displays. Die Widerstände R7 und R8 (siehe Abb. 15) dienen als Vorwiderstände für die Beleuchtung des Displays, um den Widerstand nicht zu überlasten oder einen Widerstand mit höherer Verlustleistung nutzen zu müssen.

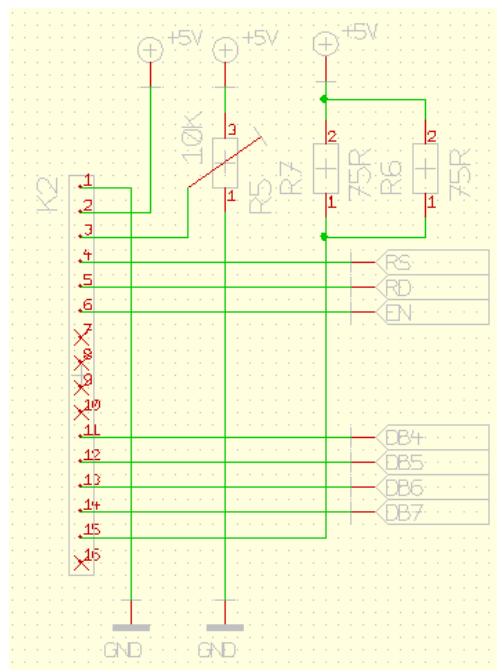


Abbildung 15: Display

3.2 Layout

3.2.1 Bestückung

Bei der Bestückung wurde auf eine günstige Positionierung der Bauteile geachtet. Um einen einwandfreien Betrieb zu ermöglichen ist bei der Positionierung der Bodensensoren mit besonderer Sorgfalt vorzugehen.

Der Prozessor als Zentrales Element wurde in die Mitte des Aufbaus gesetzt, um Leiterbahnen aus allen Bereichen der Platine zum Prozessor führen zu können. Bei der Ansteuerung der Motoren ist darauf zu achten das die Leiterbahnen möglichst kurz geführt werden sollen, um Elektromagnetische Strahlungen zu vermeiden.

3.2.2 Leiterbahnen

Um die Leiterbahnführung zu erleichtern und die Abstrahlung von elektromagnetischen Wellen zu vermindern wurden in die Innenliegenden Leiterbahnebenen eine Massefläche und eine Versorgungsfläche eingebracht.

Weiters ist darauf zu achten, dass die Strombelastbarkeit der Leiterbahnen vor allem bei der Stromversorgung und der Motoransteuerung nicht überschritten wird. Prinzipiell wurde versucht die Leiterbahnen an der Oberfläche der Leiterplatte in Längsrichtung und auf der Unterseite in Querrichtung auszurichten. Aus Gründen der Bauteilplatzierung und dem Wunsch eine möglichst geringe Anzahl an Durchkontaktierungen zu erreichen war dies nicht immer möglich. Jedoch ist die Vorzugsrichtung der Leiterbahnen gut zu erkennen (siehe Abb. 16). Abb. 16 ist eine schematische Darstellung des Layouts. Leiterbahnbreiten und Bauteile haben sind nur schematisch dargestellt. Auf die Darstellung der Masse und der Versorgungsfläche, in den Innenlagen wurde verzichtet.

4 Software

Die Gesamte Programmierung wurde in der Programmiersprache C geschrieben. Als Entwicklungsumgebung wurde CodeVisionAVR Version 2.04.4a Advanced verwendet. Diese Software ist eine Entwicklung von HP Info Tech. Alle Programme wurden in CodeVisionAVR geschrieben und dort kompiliert. Die erstellten .hex und .eep Files wurden mittels AVR Studio 4 auf den Mikrocontroller geladen.

Aus mehreren Vorgängerprojekten konnten große Programmteile übernommen werden, wie z.B. die Ansteuerungen sowie Regelung der Motoren als auch die Ansteuerung des Displays.

4.1 Display

Das Display dient zur Schnelleinstellung mehrerer Parameter, auf welche später noch genauer eingegangen wird. Das Display ist hinter den Mikrocontroller und mittig zwischen den Reifen platziert. Bei Bedarf kann das Display entfernt werden da es nur mittels Steckverbindung angebracht ist.

4.1.1 Displaybeschreibung

Im Hauptmenü konnten folgende Einstellungen getroffen werden:

- Start
In Start konnte das jeweilig eingestellte Programm (Slalom od. Enhanced Mode) gestartet werden
- Parameter
Unter Parameter konnten folgende Parameter eingestellt und ausgelesen werden:
 - V_max
Durch diese Variable konnte die Maximale Geschwindigkeit des Roboters eingestellt werden.

- Linie KP
„Linie KP“ wurde verwendet, um die Geschwindigkeit noch genauer einzustellen.
 - Time
Mit diesem Parameter kann die aktuelle Laufzeit des Roboters ausgelesen werden. Dieser Wert wurde beim Enhanced Mode zum Einschalten der vorderen Abstandssensoren verwendet.
 - Drehz. KP
Dient zur Grobeinstellung der Motorregelung.
 - Drehz. KI
Dient ebenfalls zur Grobeinstellung der Motorregelung.
- Info
Im Menü „Info“ wird der gewählte Modus, sowie die Akkuleistung am Display ausgegeben.
 - Mode
Dieser Menüpunkt dient zum Einstellen des gewünschten Modus. Unterschieden wird zwischen „Slalom“ und „Slalom-Enhanced“ Mode.
 - Test
Der „Test“-Modus dient zum Überprüfen der Sensoren.

4.2 Slalom

Für die Disziplin Slalom erkennt und verfolgt der Roboter mithilfe der vorne angebrachten Infrarotsensoren die schwarze Linie.

Hierbei wird grundsätzlich das Programm von Vorprojekten verwendet. Jedoch ist das Displayprogramm sowie die Ansteuerung der Motoren für den Roboter angepasst.

4.2.1 Steuerung und Lenkung

Die Geschwindigkeit sowie die Lenkung des Roboters werden durch das Ansprechen der Infrarotsensoren gesteuert. Gelenkt wird nur durch die jeweilig unterschiedlichen Drehzahlen

der Motoren. Eine Linksdrehung ist durch eine erhöhte Geschwindigkeit des rechten Rades möglich, eine Rechtsdrehung wird durch eine erhöhte Geschwindigkeit des linken Rades möglich.

Für alle verschiedenen im Slalom auftretenden Fälle von Sensorstellung ist jeweils ein spezifischer Lenkeinschlag eingestellt. Für jede spezifische Einstellung ist auch eine gewisse Geschwindigkeit eingestellt. So ist der Roboter beim Ansprechen der äußeren Sensoren langsamer und beim Ansprechen der inneren Sensoren schneller.

4.2.2 Lösung der Steuerung und Lenkung

Im Slalom und Slalom Enhanced Mode ist das exakte Verfolgen der Linie von großer Bedeutung, da andernfalls Zeit eingebüßt wird. Die übergeordnete if-Schleife dient zur Überprüfung, ob einer der Bodensensoren anschlägt. Um die Linie exakt verfolgen zu können werden verschiedene Kombinationen, wie die Sensoren ansprechen können abgearbeitet und die Geschwindigkeit der Motoren durch bestimmte Multiplikatoren eingestellt. Zusätzlich wird bei den äußeren Sensoren mittels Flags (links, rechts) gespeichert, auf welcher Seite der Roboter die Linie zuletzt detektiert hat.

```
if(RRR == 1 || RR == 1 || R == 1 || M == 1 || L == 1 || LL == 1 || LLL == 1)
{
    if(RRR == 1 && RR == 0)
    {
        speedMotor2 = 1.0 * linie * speed;
        speedMotor1 = -0.0 * linie * speed;
        links = 0;
        rechts = 1;
    }
    if(RRR == 1 && RR == 1)
    {
        speedMotor2 = 0.9 * linie * speed;
        speedMotor1 = -0.3 * linie * speed;
        links = 0;
        rechts = 1;
    }
    if(RRR == 0 && RR == 1 && R == 0)
    {
        speedMotor2 = 0.9 * linie * speed;
        speedMotor1 = -0.44 * linie * speed;
        links = 0;
        rechts = 0;
    }
    if(RR == 1 && R == 1)
    {
        speedMotor2 = 0.8 * linie * speed;
        speedMotor1 = -0.58 * linie * speed;
        links = 0;
        rechts = 0;
    }
    if(RR == 0 && R == 1 && M == 0)
    {
        speedMotor2 = 0.8 * linie * speed;
        speedMotor1 = -0.72 * linie * speed;
        links = 0;
        rechts = 0;
    }
}
```

```

if(R == 1 && M == 1)
{
    speedMotor2 = 0.86 * linie * speed;
    speedMotor1 = -0.86 * linie * speed;
    links = 0;
    rechts = 0;
}
if(R == 0 && M == 1 && L == 0)
{
    speedMotor2 = 1.0 * linie * speed;
    speedMotor1 = -1.0 * linie * speed;
    links = 0;
    rechts = 0;
}
if(M == 1 && L == 1)
{
    speedMotor2 = 0.86 * linie * speed;
    speedMotor1 = -0.86 * linie * speed;
    links = 0;
    rechts = 0;
}
if(M == 0 && L == 1 && LL == 0)
{
    speedMotor2 = 0.72 * linie * speed;
    speedMotor1 = -0.8 * linie * speed;
    links = 0;
    rechts = 0;
}
if(L == 1 && LL == 1)
{
    speedMotor2 = 0.58 * linie * speed;
    speedMotor1 = -0.8 * linie * speed;
    links = 0;
    rechts = 0;
}
if(L == 0 && LL == 1 && LLL == 0)
{
    speedMotor2 = 0.44 * linie * speed;
    speedMotor1 = -0.9 * linie * speed;
    links = 0;
    rechts = 0;
}
if(LL == 1 && LLL == 1)
{
    speedMotor2 = 0.3 * linie * speed;
    speedMotor1 = -0.9 * linie * speed;
    links = 1;
    rechts = 0;
}
if(LLL == 1 && LL == 0)
{
    speedMotor2 = 0.0 * linie * speed;
    speedMotor1 = -1.0 * linie * speed;
    links = 1;
    rechts = 0;
}
}

```

Falls der Roboter die Linie verlässt, bzw. diese nicht mehr erkennt wird der folgende Code durchlaufen:

```

if( RRR == 0 &&
    RR == 0 &&
    R == 0 &&
    M == 0 &&
    L == 0 &&
    LL == 0 &&
    LLL == 0 &&
    links == 1)
{
    speedMotor2 = 0.0 * linie * speed;
    speedMotor1 = -1.0 * linie * speed;
}
if( RRR == 0 &&
    RR == 0 &&

```

```

R == 0 &&
M == 0 &&
L == 0 &&
LL == 0 &&
LLL == 0 &&
rechts == 1)
{
    speedMotor2 = 1.0 * linie * speed;
    speedMotor1 = -0.0 * linie * speed;
}

if( RRR == 0 &&
RR == 0 &&
R == 0 &&
M == 0 &&
L == 0 &&
LL == 0 &&
LLL == 0 &&
links == 0 &&
rechts == 0)
{
    speedMotor2 = 1.0 * linie * speed;
    speedMotor1 = -1.0 * linie * speed;
}

```

Wie oben zu sehen ist, erkennt keiner der Sensoren (RRR bis LLL; RRR ist jener Sensor rechts außen; LLL ist jener Sensor links außen) die Linie. In den oberen beiden IF Abfragen wird der Fall behandelt das der Roboter nach links bzw. rechts Ausbricht, was mit den bereits erwähnten Flags „rechts“ und „links“ möglich ist.

Die letzte Abfrage behandelt den Fall einer Linienunterbrechung. Eine solche wird dadurch erkannt, wenn keiner der Sensoren die Linie erkennt und die Flags „rechts“ und „links“ nicht gesetzt wurden. In diesem Fall sollte der Roboter gerade aus fahren. (Dieser Fall ist nur für den Slalom-Enhanced-Bewerb erforderlich)

4.3 Enhanced Mode

Für die Disziplin Enhanced Mode wurde der Code vom Slalom verwendet und erweitert.

Beim Erkennen eines Hindernisses wird eine fest vorprogrammierte Ausweichroutine aufgerufen, um das Hindernis sicher zu umfahren. Im Tunnel wird die Linie ohne Änderungen verfolgt und die Linienunterbrechung wird gerade überbrückt.

4.3.1 Steuerung und Lenkung

Die Steuerung ist gleich wie beim Slalom Mode. Beschrieben bei 4.2.1.

4.3.2 Lösung der Steuerung und Lenkung

Da die Steuerung gleich wie beim Slalom Mode ist wurde nur noch die Ausweichroutine ergänzt.

Der nachfolgende Code stellt die Ausweichroutine dar. Sobald alle Bedingungen erfüllt sind beginnt der Roboter mit dem ausweichen.

Bedingungen sind:

- Das Hindernis wurde noch nicht passiert
- Der Abstandsensor erkennt bzw. erkannte das Hindernis
- Die Zeit stimmt

Mit der Zeit wurde vorgegeben ab wann der Sensor aktiviert ist. Dies dient dazu ein frühzeitiges Anschlagen des Sensors (z.B. durch den Tunnel) zu unterbinden.

```
if (ziegelPassiert == 0)
{
    if((AbstandR == 0 && countZiegel >= (time / ABTAST_ZEIT)) ||
        (ziegel == 1 && countZiegel >= (time / ABTAST_ZEIT)))
    {
        ziegel = 1;
        if(count <= (240 / ABTAST_ZEIT))
        {
            speedMotor2 = 1.0 * 1.2;
            speedMotor1 = -1.05 * 1.2;
            links = 0;
            rechts = 0;
        }

        if(count >= (240 / ABTAST_ZEIT) &&
            count <= (1000/ ABTAST_ZEIT))
        {
            speedMotor2 = 1.0 * 1.2;
            speedMotor1 = -0.5 * 1.2;
        }

        if(count >= (1500 / ABTAST_ZEIT))
        {
            speedMotor2 = 0.5* 1.2;
            speedMotor1 = 0.5* 1.2;
            ziegel = 0;
            links = 0;
            rechts = 0;
        }
        count++;
    }
}
```

Wie oben zu sehen, wird das Ausweichen immer gleich abgearbeitet.

Der nachfolgende Code beschreibt das Vorzeitige Abbrechen der Ausweichroutine falls der Roboter die Linie schon vorher wieder erkannt hat.

```

if(count >= ( 500 / ABTAST_ZEIT))
{
    if( (RRR == 1 && RR == 1) ||
        (RR == 1 && R == 1) ||
        (R == 1 && M == 1) ||
        (M == 1 && L == 1) ||
        (L == 1 && LL == 1) ||
        (LL == 1 && LLL == 1))
    {
        ziegel = 0;
        count = 0;
        ziegelPassiert = 1;
    }
}

```

Die Abfrage „countZiegel >= (time / ABTAST_ZEIT)“ ist nötig, da der Abstandssensor die Tunnelwand als Hindernis erkennen kann. CountZiegel musste für den jeweiligen Streckenverlauf eingestellt werden, um die Abstandssensoren beim Tunnel deaktivieren und beim Ziegelstein aktivieren zu können. Bei unserem Lauf in Wien wurden die Abstandssensoren erst nach dem Tunnel eingeschaltet, um einen Fehlanschlag der Sensoren im Tunnel zu vermeiden.

5 Verbesserungsmöglichkeiten

5.1 Bodensensoren

Wie in Punkt 2.5 beschrieben wurden Bodensensoren an der Unterseite der Platine befestigt. Durch diese Konstruktion sind diese allerdings sehr anfällig für Beschädigungen. Wir hatten das Problem, dass durch den Stoß zwischen den Parcourplatten die Sensoren immer wieder auf dem Boden aufgesessen sind und somit lösten sich die Lötunkte von der Platine. Eine mögliche Verbesserung die man vornehmen könnte, wären in die Platine versenkte Sensoren, die an der Oberseite der Platine verlötet werden.

5.2 Robotergröße

Durch die Verwendung einer Europlatine (100mm x 160mm) konnten die Bodensensoren nur auf den 10cm breiten Frontstück verteilt werden, wodurch schon bei geringen Ablenkungen von der Linie sehr stark gegengelenkt werden musste. Durch die starken Lenkbewegungen ist es zu starken Übersteuerungen gekommen.

Durch eine breitere Konstruktion des Roboters könnten die Lenkbewegungen feiner abgestimmt werden und somit ein Übersteuern vermieden werden.

Zusätzlich sollte ein größerer Abstand der Sensoren zu den angetriebenen Reifen bzw. zur Lenkkonstruktion eingehalten werden, um eine ruhigere Fahrweise zu ermöglichen.

5.3 Modulbauweise der Abstandssensoren

Um die Abstandssensoren in ihrer Position verändern zu können sollten diese nicht fix auf der Hauptplatine angelötet werden, sondern auf eine extra Platine angebracht werden, die über Flachbandkabel mit der Hauptplatine verbunden wird. Somit können die Abstandssensoren in ihrer Position sowie im Neigungswinkel verändert werden.

5.4 Fixer Akkuplatz

Bei unserem Aufbau des Roboters wurde der Akku nur mittels Klebeband am Steg zwischen Hauptplatine und Bodensensoren befestigt. Somit wurden bei jedem Akkuwechsel der Trägheitsschwerpunkt und die Fahreigenschaften verändert.

Um den Akku besser fixieren zu können und somit auch den Trägheitsschwerpunkt stabil zu halten sollte eine Akkuschale fix angebracht werden.

6 Gegnerisches Wettkampfkonzep

Einer der stärksten Linefollower auf der RobotChallenge in Wien war der polnische Roboter Hurricane (siehe Abb. 17 und Abb. 18). Durch eine erhöhte Bodenhaftung mittels Lüfter und der damit verbundenen exakten Verfolgung der Linie konnte sich dieser Roboter einen Vorsprung verschaffen. Durch den Lüfter wurde versucht den Roboter an den Untergrund anzusaugen und so einen Unterdruck zu erreichen. Dadurch wurde eine sehr gute Haftung des Roboters erreicht und dieser weichte auch bei höheren Geschwindigkeiten kaum von der Linie ab.

Ein weiterer Vorteil von Hurricane war das breite Sensorschild auf dem die Sensoren in größeren Abständen zueinander angebracht waren. Durch diese Abstände musste der Roboter nicht so aggressiv lenken und konnte somit ein Aufschwingen wie wir es bei unserem Roboter hatten verhindern.

Zusätzlich hatte der Hurricane noch LED's über jedem Sensor angebracht, die beim ansprechen des Sensors leuchteten. Somit konnte die Funktionalität der Bodensensoren sehr schnell überprüft werden.

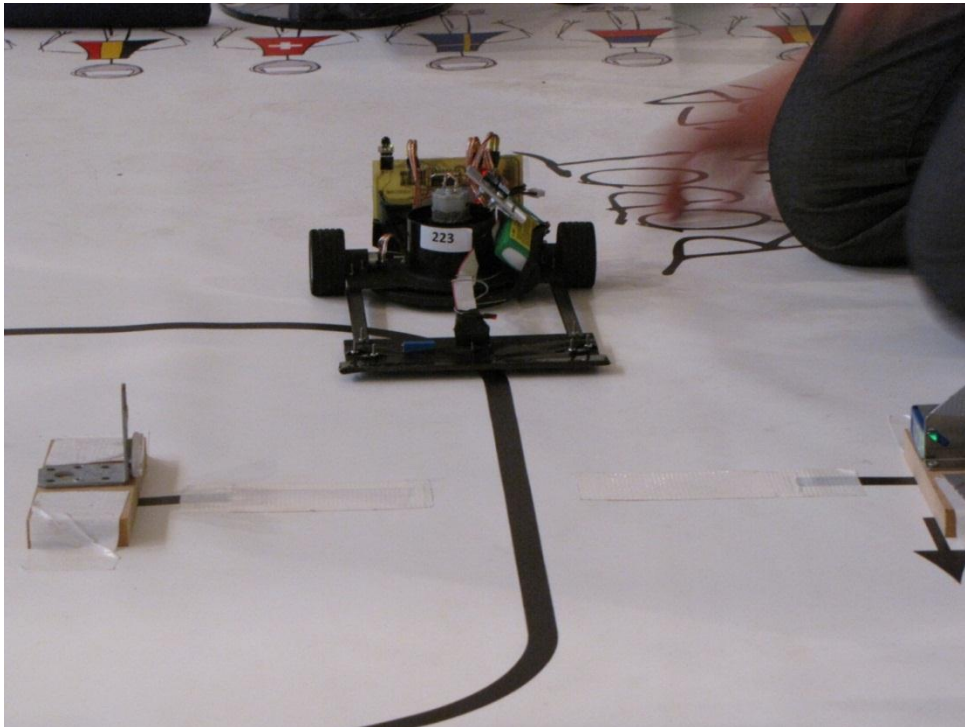


Abbildung 17: "Hurricane"



Abbildung 18: "Hurricane"

FIELD A
Line Follower Qualifying (ends at 16:00) starts at 10:30

Pl.	Robot	Time	Att.	Pl.	Robot	Time	Att.
1	Hurricane - 223 <small>ROB</small>	00:08.07	7	13	Hellcat - 51 <small>AUT</small>	00:13.91	7
2	Mefisto - 39 <small>ROB</small>	00:09.42	4	14	The Iron - 27 <small>SVK</small>	00:14.05	4
3	2fast4you - 186 <small>USU</small>	00:09.44	2	15	Sparks - 25 <small>ROB</small>	00:14.21	1
4	IRE [MBT] - 75 <small>ROB</small>	00:09.75	9	16	MATO - 21 <small>AUT</small>	00:15.99	7
5	GreenNight - 178 <small>ROB</small>	00:10.09	1	17	MONTY - 64 <small>AUT</small>	00:16.34	4
6	Challenger[FH-... - 150 <small>AUT</small>	00:10.24	6	18	Armadillo - 144 <small>ROB</small>	00:18.78	4
7	RoadRunner [FH-... - 46 <small>AUT</small>	00:11.03	5	19	RGT - 148 <small>SVK</small>	00:19.91	3
8	Jack - 160 <small>ROB</small>	00:11.38	11	20	Parkinson - 91 <small>ROB</small>	00:20.22	6
9	Nautilus[FH-Wels] - 149 <small>AUT</small>	00:11.66	4	21	R.Obot - 89 <small>IRL</small>	00:23.36	3
10	Serdel - 171 <small>ROB</small>	00:11.73	8	22	Leonis - 97 <small>SVK</small>	00:24.62	2
11	Lasher - 30 <small>ROB</small>	00:12.01	1	23	Nite 1 - 11 <small>SVK</small>	00:33.49	1
12	Minoris - 96 <small>SVK</small>	00:12.48	3	24	heinz werner - 233 <small>AUT</small>	00:37.90	3

- 75 ROB

Abbildung 19: Qualifikation Linienverfolger

Durch die leichte Bauweise und den Lüfter konnten dann auch Rundenzeiten von 8 Sekunden erreicht werden (siehe Abb. 19).

Mit dem oben beschriebenen Konzept von den Robotern Challenger und Nautilus konnten Zeiten von 10,24 Sekunden(Challenger) bzw. 11,66 Sekunden (Nautilus) erreicht werden..

7 Quellen

robotchallenge.org. (04. 01 2011). Abgerufen am 10. 06 2011 von [robotchallenge.org](http://www.robotchallenge.org):

http://www.robotchallenge.org/fileadmin/user_upload/_temp_/RobotChallenge/Reglement/RC-LineFollower.pdf