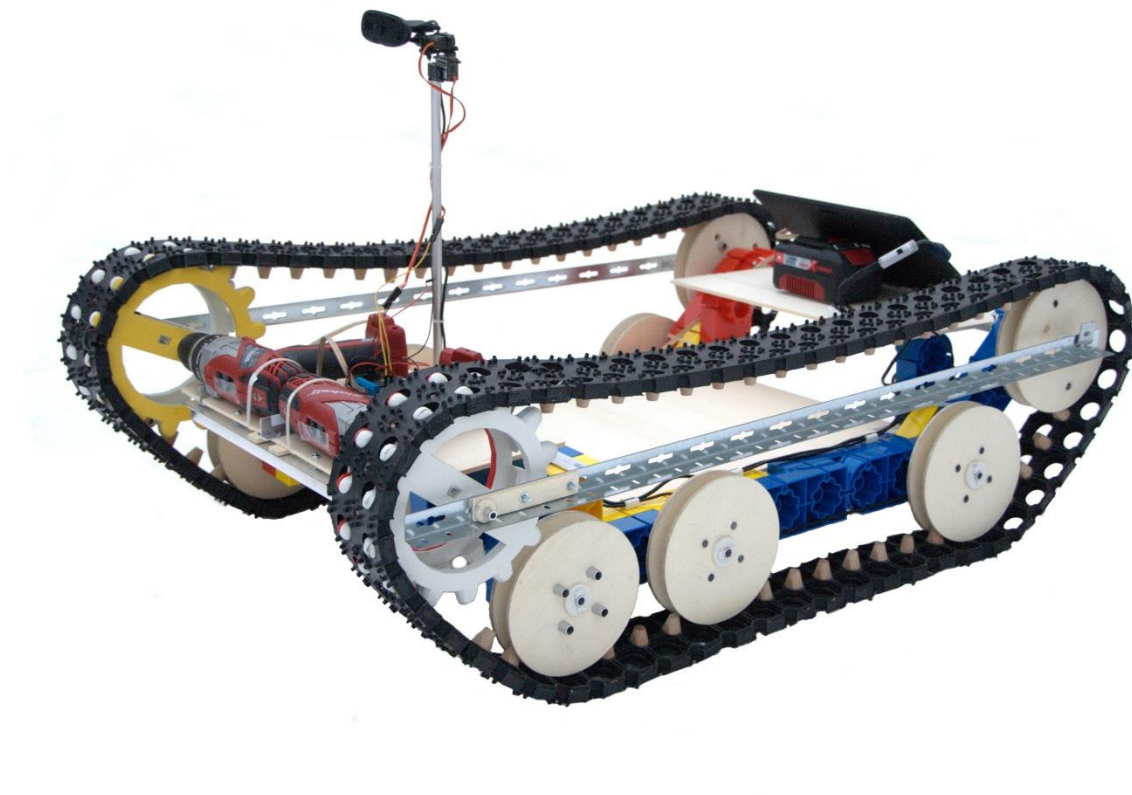


Maturitätsarbeit HS 2016/17

Nicola Gysler, 4fMN

WALL-LEE – Entwicklung eines Erkundungsroboters



Kantonsschule Im Lee Winterthur, 9. Januar 2017

Betreuer: Wolfgang Pils

Inhalt

1	Einleitung.....	1
2	Elektronik.....	2
2.1	Arduino.....	2
2.2	Arduino Motor Shield.....	3
2.3	Motoren und Akku.....	4
2.4	Steuerung und Kamera.....	5
3	Struktur und Mechanik.....	6
3.1	Grundstruktur.....	6
3.2	Raupensystem.....	8
4	Software.....	12
4.1	Programm für Windows.....	12
4.1.1	Drittkomponenten und Abhängigkeiten.....	12
4.1.2	WLAN Kommunikation.....	12
4.2	Programm für Arduino.....	13
5	Bildverzeichnis.....	15
6	Schlusswort.....	17
7	Danksagung.....	17
8	Anhang.....	18

1 Einleitung

Das Ziel dieser Maturaarbeit war es, ein Roboterfahrzeug zu bauen, das möglichst mobil ist und sogar Treppen überwinden kann.

Der Roboter wird via WLAN gesteuert. Er ist mit einer Kamera ausgestattet, auf welche live zugegriffen werden kann. Somit ist es möglich, ein Areal oder Gebäude zu überwachen und zu erkunden.

Die Fähigkeit Treppen zu bewältigen ist deshalb wichtig, weil in einem grossen Gebäude der Einsatzbereich stark vergrössert wird, wenn der Zutritt zu verschiedenen Stockwerken möglich ist. Der Roboter bewegt sich mithilfe von Raupen fort, ähnlich einem Raupenbagger, welcher ja ebenfalls für das Gelände geeignet ist.

Die meisten Teile für das Fahrzeug wurden mit relativ einfachen Mitteln selber gebaut. Für manche Bauelemente wurde ein 3D-Drucker verwendet, was den Bau schwierig herzustellender Teile erleichtert oder sogar erst ermöglicht hat.

Beim Bau von WALL-LEE wurde ausserdem versucht, die Kosten möglichst tief zu halten.

2 Elektronik

Gesteuert wird der Roboter mit einem Computer, der über WLAN die Steuerbefehle an einen weiteren kleinen Computer auf dem Roboter sendet. Diese Befehle werden dann per Kabel an einen Mikrocontroller weitergegeben, welcher schliesslich die Bewegungen des Roboters steuert.



Abb. 1: Schema der Kommunikation. Links ein Laptop, rechts ein Tablet mit Kamera und unten ein Mikrocontroller

2.1 Arduino

Arduino ist eine Open-Source-Plattform mit einfach aufgebauten Soft- und Hardware-Bauteilen.

Die Firma Arduino produziert Mikrocontroller-Kits, sogenannte *Arduino Boards*, mit welchen sowohl Sensordaten eingelesen als auch Aktoren angesteuert werden können.

Programmiert werden diese Boards mit der Programmiersprache C++.

Für diese Mikrocontroller gibt es eine Vielzahl an Erweiterungsmöglichkeiten, sogenannte *Shields*, wie zum Beispiel Bluetooth- und WLAN-Erweiterungen, oder wie in dieser Arbeit benötigt ein *Motor Shield*, welches in der Lage ist, grössere Ströme für Motoren zu regeln.

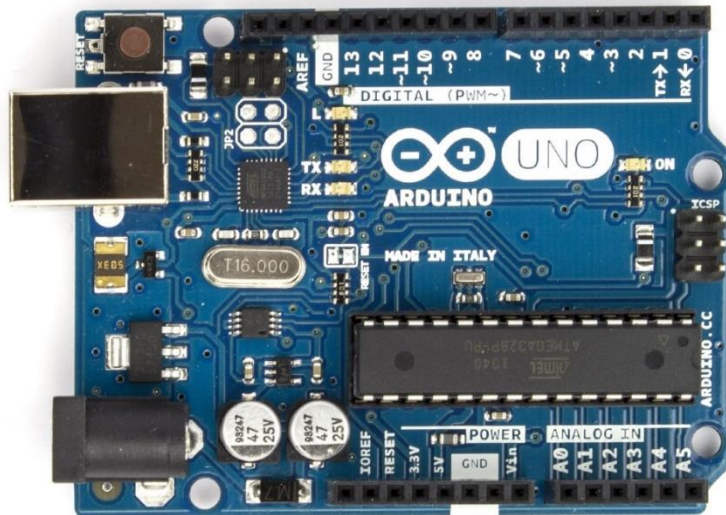


Abb. 2: Arduino Board Arduino UNO

2.2 Arduino Motor Shield

Der Roboter wird mithilfe des *Dual VNH5019 Motor Driver Shields* der Firma Pololu betrieben.

Aufgrund der Größe von WALL-LEE war bei der Auswahl des Shields wichtig, dass eine hohe Stromstärke kontrolliert werden kann. Dieses Motor Shield steuert zwei Motoren und kann pro Ausgang einen Dauerstrom von 12 A und einen Peak von 30 A verkraften.

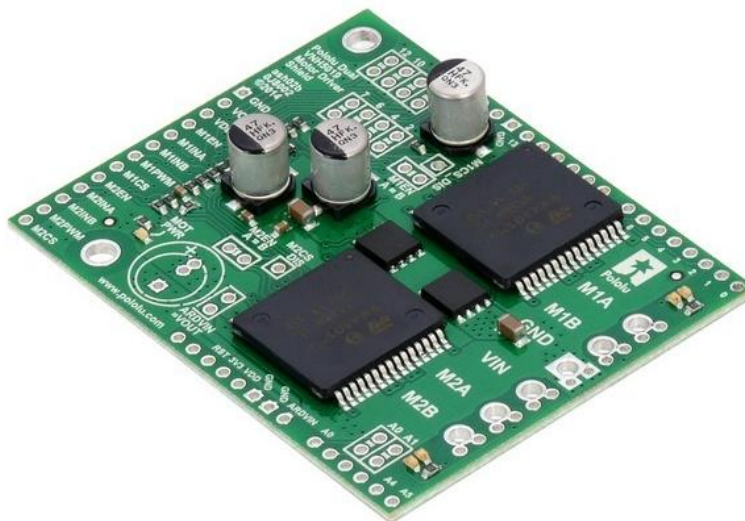


Abb. 3: Arduino Motor Shield *Dual VNH5019 Motor Driver Shield*

2.3 Motoren und Akku

Folgende Kriterien waren bei der Auswahl der Antriebseinheit wichtig: Preis, Drehmoment, Drehzahl. Da ein Elektromotor üblicherweise eine sehr hohe Drehzahl aufweist (Beispiel kleiner Elektromotor, ca 10'000 Umdrehungen pro Minute), wird für den Antrieb des Roboters ein Übersetzungsgetriebe benötigt. Da ein ausreichend starker Motor mit Getriebe nicht gerade günstig ist, entstand die Idee, einen älteren, nicht mehr benötigten Akkubohrer als Antrieb zu verwenden. Über die Tauschbörse Ricardo konnte kostengünstig ein weiteres, baugleiches Modell ersteigert werden.

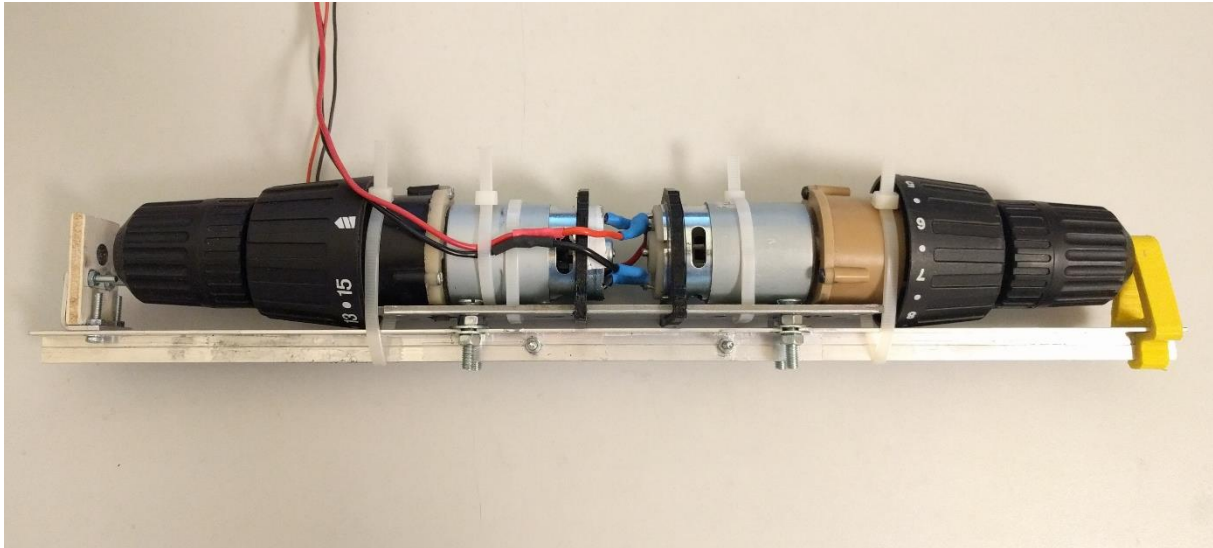


Abb. 4: Erstes Antriebsmodul

Diese Motoreinheit hatte damit eine Drehzahl von maximal 550 Umdrehungen pro Minute und eine Maximalspannung von 14,4 Volt.

Im weiteren Verlauf der Entwicklung des Roboters zeigte sich, dass die Motoren nicht genug Kraft hatten, um grössere Hindernisse wie beispielsweise eine Treppe zu überwinden. Im Nachhinein wurde das Drehmoment experimentell gemessen. Dies ergab einen Wert von nur 4.3 Nm. Berechnungen ergaben, dass man für eine 45 Grad-Steigung mindestens das doppelte Drehmoment benötigen würde.

Da die alten Motoren aus Akkubohrern der Firma Einhell stammen und der Kauf von komplett neuen das Budget gesprengt hätte, lag es nahe, Einhell kurzfristig um Unterstützung in Form eines Sponsorings anzufragen. Einhell Schweiz stellte für dieses Projekt grosszügig zwei neue, stärkere Akkubohrer inklusive Akkus und Ladegerät zur Verfügung.



Abb. 5: Logo der Firma Einhell Schweiz AG

Die neuen Akkubohrer verfügen über ein maximales Drehmoment von 48 Nm, also über 10-mal mehr als die vorherigen. Erreicht wird dieser Wert durch ein 2 Gang-Getriebe und einen leistungsfähigeren Motor mit einer höheren Maximalspannung von 18 V. Mit diesem neuen Antrieb hat der Roboter mehr als genug Kraft, um Hindernisse zu überwinden.



Abb. 6: Akku-Bohrschauber TE-CD 18 Li inkl. Akku

2.4 Steuerung und Kamera

Da heute viele Gebäude bereits über ein WLAN-Netzwerk verfügen, kann dieses direkt zur Steuerung des Roboters verwendet werden. Mit geringem Aufwand könnte die Lösung auch so ausgebaut werden, dass der Roboter über mobiles Internet gesteuert werden kann.

Für den WLAN-Empfang wird ein Windows-Tablet verwendet. Dadurch wird der Einsatz der kostenlosen Software *Skype* für die Bildübertragung möglich. Da *Skype* eine Videoübertragung in beide Richtungen unterstützt, wird der Erkundungsroboter gleichzeitig zu einem Telepräsenzroboter, indem der Erkunder selber auch auf dem Tablet am Roboter erscheint.

Für eine gute Bildqualität wird eine externe USB-Webcam verwendet.



Gesteuert wird der Roboter schliesslich über eine USB-Fernsteuerung.

Abb. 7: Verwendete USB-Flugsimulator-Fernsteuerung

3 Struktur und Mechanik

WALL-LEE fährt nach dem Prinzip eines Raupenfahrzeugs, vom Aufbau her ähnlich einem Bagger oder Panzer.

3.1 Grundstruktur

Die Grundstruktur des Roboters besteht aus einem Bausteinsystem der Firma Kiditec, ähnlich wie Lego, jedoch viel grösser.

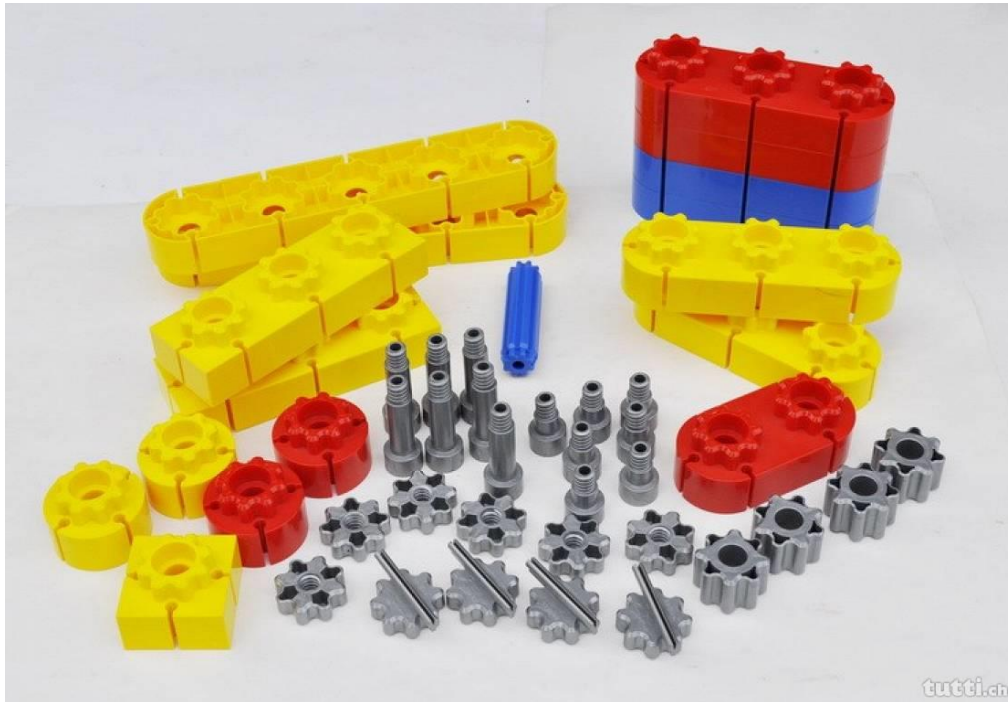


Abb. 8: Kiditec-Bausteine (ein 1x1-Stein hat die Länge 6x6 cm)

Diese Wahl wurde getroffen, da sich ein solches System sehr gut eignet, um Ansätze zu testen und ohne grossen Aufwand immer wieder umgebaut werden kann. Ursprünglich waren die Kiditec-Bausteine nur für die Prototyp-Phase vorgesehen, da die Verbindungen zwischen den einzelnen Bausteinen zu wenig stabil waren. Durch Zusammenbinden der Verbindungsstellen konnte die erforderliche Stabilität jedoch gewährleistet werden, wodurch das Kiditec-System nun doch für das Endprodukt eingesetzt werden konnte.

Der Roboter ist knapp einen Meter lang und 67 cm breit.

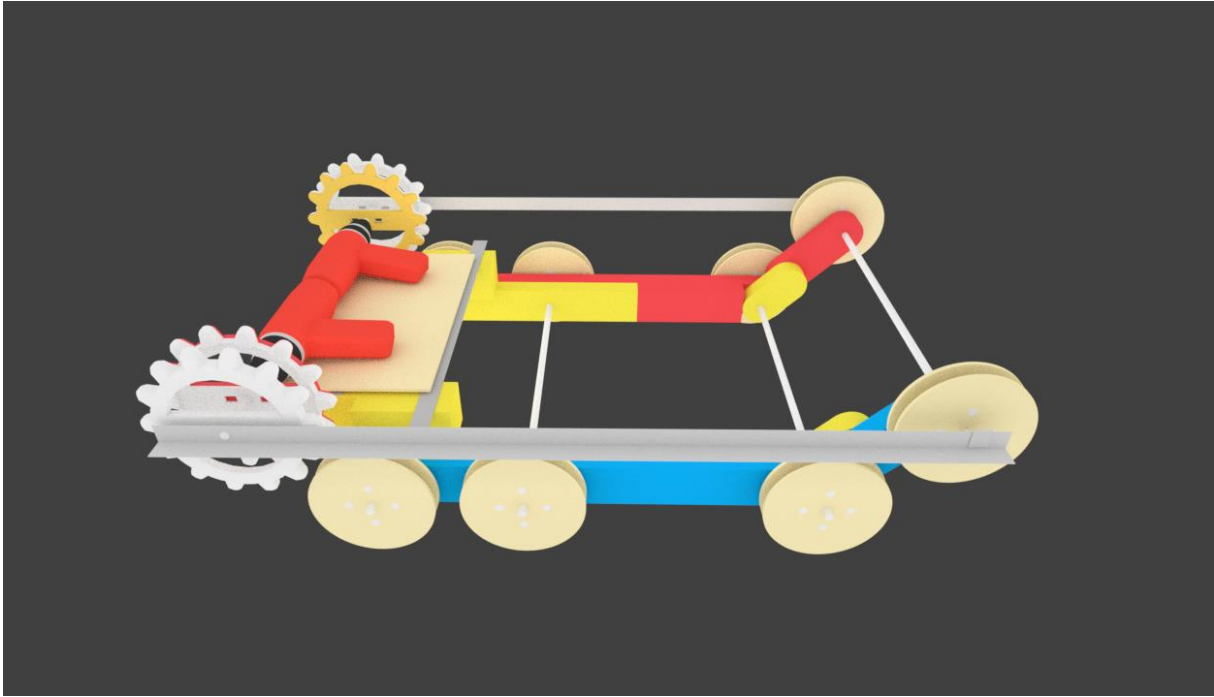
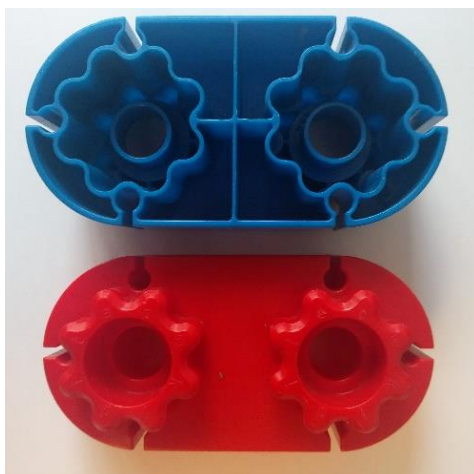


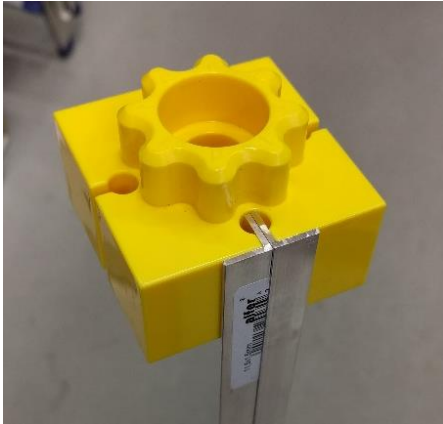
Abb. 9: computermodellerte Veranschaulichung der mechanisch wichtigen Teile, ohne Raupe



Die Bauelemente haben auf der einen Seite Noppen und auf der anderen dazu passende Andock-Vertiefungen. Weiter befinden sich Rillen an den Seiten.

Abb. 10: Andock-System der Kiditec-Elemente

Nun wurden weitere mechanische Teile wie Motoren und Achsen an diesem Grundgerüst angebracht. Dafür wurden zwei Ansätze verfolgt:



Ansatz 1: In die Rillen der Kiditec-Elemente wurden genau passende Metallwinkel gesteckt. Auf diesen wurden die Motoren befestigt.

Abb. 11: Metallwinkel passen genau in die Rillen der Kiditec-Elemente



Ansatz 2: Da das bereits vorhandene Loch in den Bausteinen grösser ist als der Achsendurchmesser, wurde ein blumenförmiger Andock-Noppen mit dem 3D-Drucker hergestellt. Dadurch kann die Achse mit dem Kiditec-Gerüst verbunden werden.

Abb. 12: 3D-gedruckter Andock-Noppen

3.2 Raupensystem

Die Fortbewegung erfolgt mittels Raupen. Der Roboter sollte möglichst geländetauglich sein und sogar Treppen hinauffahren können. Mit einem Raupensystem kann all das erreicht werden.

Das Steuern mit einem Raupenantrieb funktioniert, indem die eine Raupe schneller fährt als die andere. Wenn sich beide Raupen gleichschnell drehen, bewegt sich das Fahrzeug geradeaus. Es sind auch Drehungen auf der Stelle möglich, indem sich die Raupen in entgegengesetzte Richtungen drehen.

Eine Raupe läuft jeweils über vier in einem umgekehrten Trapez angeordnete Hauptpunkte. Drei davon sind einfache Räder. Das Vierte ist das Antriebsrad und wird von einem Akku-Bohrschrauber angetrieben. Die Trapezform ist von Vorteil um über Hindernisse zu fahren, weil sie vorne und hinten eine Art Rampe schafft.

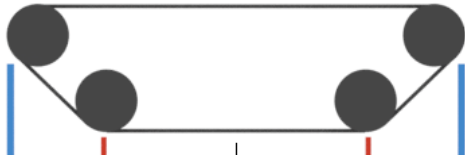
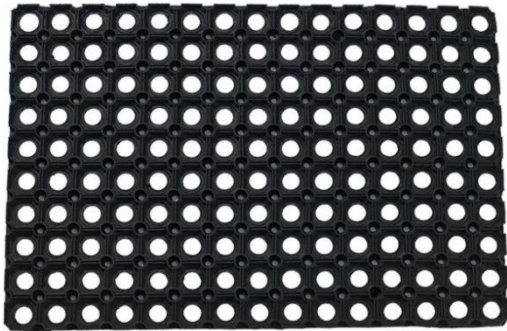


Abb. 13: Trapezform der Raupe (Seitenansicht)



Die Raupe besteht aus Gummi. Sie wurde aus einer Gummiringmatte hergestellt, welche in Streifen geschnitten wurde. Die Dicke der Matte beträgt 1.5 cm. Die Länge einer Raupe beträgt etwas mehr als 2 m.

Abb. 14: Gummiringmatte, 40x60 cm

Angetrieben wird die Raupe durch ein Antriebsrad mit Zacken, welche genau in die Löcher der Gummimatte greifen. Die Matte wurde vermessen und das Rad am Computer modelliert. Angefertigt wurde es mit einem 3D-Drucker.



Abb. 15: 3D-gedrucktes Antriebsrad



Abb. 16: Antriebsrad greift in Raupe

Die Zacken im Antriebsrad stellen gleichzeitig auch sicher, dass die Raupe in der Spur bleibt. Bei den übrigen Rädern muss die Führung auch gewährleistet sein. Dies wird realisiert, indem in der Mitte der Raupe kegelartige Noppen angebracht sind.

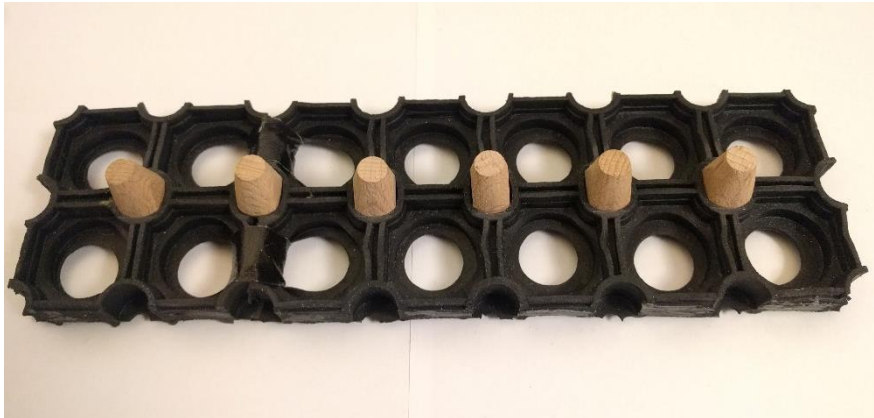


Abb. 17: Stück der Raupe mit Noppen



Abb. 18: Holznoppen

Diese Noppen werden durch den Spalt in den Rädern geführt. Zur Herstellung wurde zuerst ein Holzstab in Stücke gesägt. Diese wurden anschliessend mit einer Drehbank in eine kegelähnliche Form gebracht. Für beide Raupen wurden total 108 Stück benötigt

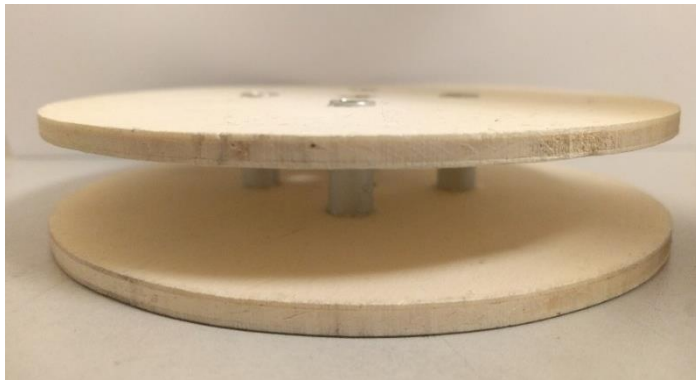
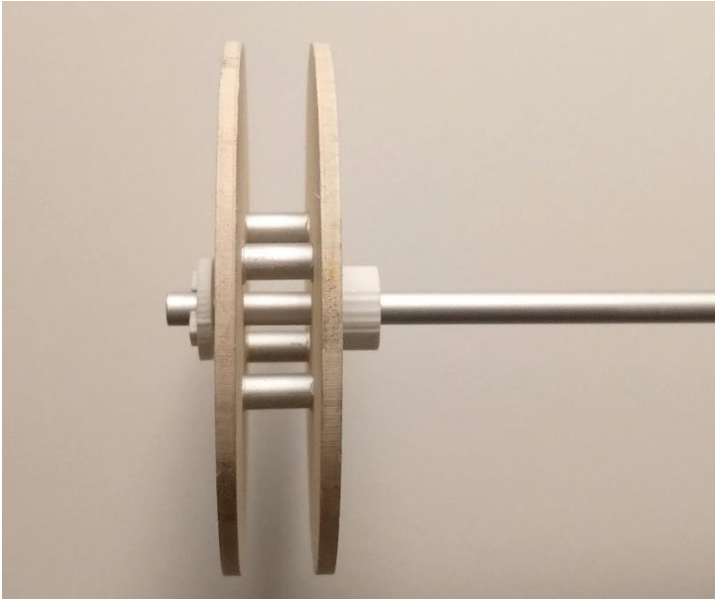


Abb. 19: Rad, gebaut aus Holz und Aluminiumrohr



Abb. 20 und Abb. 21: Raupe wird in der Spur gehalten



Damit das Rad an Ort und Stelle bleibt, wurden weitere 3D-gedruckte Teile verwendet.

Abb. 22: Rad auf Achse, beidseitig mit 3D-gedruckten Teilen befestigt

Um das stärkere Drehmoment auf das Fahrwerk zu übertragen, war es sinnvoll, den neuen Antrieb mitsamt des Akkubohrergehäuses zu befestigen.



Abb. 23: Motormodul mit neuen Akkubohrern von Einhell

4 Software

Da für den Roboter zwei verschiedene Hardware-Plattformen verwendet werden, nämlich Windows und Arduino, kommen auch zwei verschiedene Programmiersprachen zum Einsatz. Für die Windowsrechner wird C# verwendet und für Arduino C++.

Die Windows-Software läuft sowohl auf dem Roboter als auch auf dem Steuercomputer und zwar einmal im Sende- und einmal im Empfangsmodus.

Da ich mit C# noch wenig Erfahrung habe, hat mein Vater Martin Gysler die Windows-Software geschrieben und ich diejenige für das Arduino Board.

4.1 Programm für Windows

Diese Software umfasst drei Hauptaufgaben: Senden der Joystickbefehle über WLAN an den Fahrzeugcomputer, Empfangen der Steuerbefehle und Weiterleiten der Befehle an das Arduino-Board.

4.1.1 Drittkomponenten und Abhängigkeiten

Für den Betrieb der Steuersoftware werden Windows7 und das DotNet-Framework 4.6 vorausgesetzt.

Weiter werden folgende Fremdkomponenten verwendet:

MvvMlightLibs 5.2.0 (Galasoft)	Hilfsklassen für die Benutzeroberfläche
CommonServiceLocator (Microsoft)	Hilfsklassen für Desktopanwendungen
SlimDX (slimdx.org)	Hilfsklassen für die Joystick-Ansteuerung
SfGauge.WPF (Syncfusion)	Controls für die Benutzeroberfläche
WPFSpark (wpfsprk.codeplex.com)	Controls für die Benutzeroberfläche

4.1.2 WLAN Kommunikation

Die Kommunikation zwischen den beiden Windows-Geräten erfolgt über WLAN. Dabei wird die „Windows Communication Foundation“ (WCF) des .Net Frameworks verwendet. WCF unterstützt verschiedenste Kommunikationsprotokolle. Für die Steuerung des Roboters werden das Protokoll Net.tcp und der Port 808 verwendet. Der Roboter hat in dieser Kommunikation die Funktion eines Servers. Damit dieser im Netzwerk erreichbar ist, muss die Windows-Firewall die eingehende Anfrage zulassen (siehe Abb. 24).

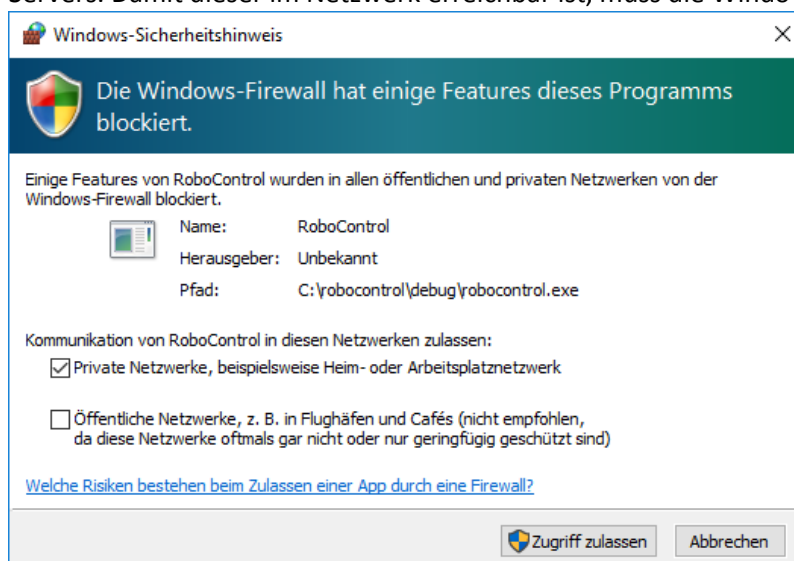


Abb. 24: Windows-Firewall Zugriffsfreigabe

Weiter müssen die Netzwerkeinstellungen so konfiguriert werden, dass der Rechner im Netzwerk sichtbar ist (siehe Abb. 25).

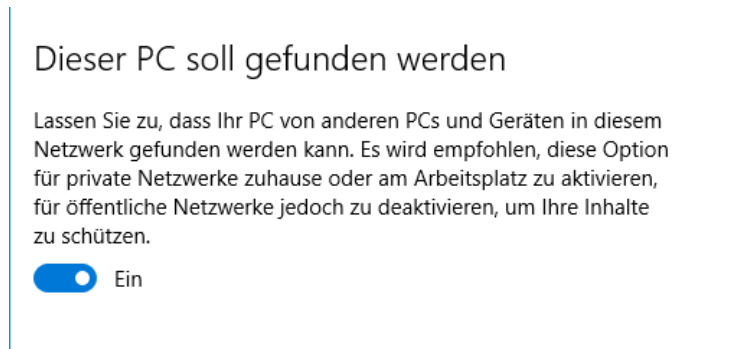


Abb. 25: Konfiguration Windows-Netzwerk

Der Steuercomputer übernimmt die Rolle eines Clients und sendet die Steuerinformationen als Anfragen an den Server. Für den Verbindungsaufbau muss der Client die IP-Adresse des Servers kennen.

4.2 Programm für Arduino

Für dieses Programm wurden folgende Quellen verwendet:

[http://www.c-sharpcorner.com/UploadFile/e46423/arduino-control-using-a-windows-presentation-foundation-\(wpf/](http://www.c-sharpcorner.com/UploadFile/e46423/arduino-control-using-a-windows-presentation-foundation-(wpf/)

<http://arduino.stackexchange.com/questions/1013/how-do-i-split-an-incoming-string>

Das Arduino Board liest die Steuerbefehle, welche vom Windows Tablet empfangen wurden, über die serielle Schnittstelle ein. Dabei werden Datenpakete in Form eines Strings empfangen:

Ein Steuerbefehl für die Motoren sieht zum Beispiel wie folgt aus:

[Motornummer]:[Motorwert]&[Motornummer]:[Motorwert]

Motornummer → Nummer des anzusteuernenden Motors

Motorwert → Geschwindigkeit, vierstellige Zahl zwischen -400 und 400, bei Bedarf mit führenden Nullen

Bsp.: "1:0000&2:0200"

Motor 1 Stillstand, Motor 2 halbe Kraft voraus

```

13
14 }
15
16 void loop()
17 {
18     if (Serial.available() > 0)
19     {
20
21         // Get next command from Serial (add 1 for final 0)
22         char input[INPUT_SIZE + 1];
23         byte size = Serial.readBytes(input, INPUT_SIZE);
24         // Add the final 0 to end the C string
25         input[size] = 0;
26
27         // Read each command pair
28         char* command = strtok(input, "&");
29         while (command != 0)
30         {
31             // Split the command in two values
32             char* separator = strchr(command, ':');
33             if (separator != 0)
34             {
35                 // Actually split the string in 2: replace ':' with 0
36                 *separator = 0;
37                 int ValueID = atoi(command);
38                 ++separator;
39                 int Value = atoi(separator);
40
41
42                 if (ValueID == 1){ // Do something with ID and Value
43                     md.setM1Speed(Value);
44
45                 }
46                 if (ValueID == 2){
47                     md.setM2Speed(Value);
48
49                 }
50
51             }
52             // Find the next command in input string
53             command = strtok(0, "&");
54         }
55     }
56 }

```

Einlesen ab der seriellen Schnittstelle

Parsen des Inputstrings

Weiterleitung der Werte an das Motorshield

Parsen des Inputstrings

Abb. 26: Arduino Code im void loop()

5 Bildverzeichnis

Abb. 1: Schema der Kommunikation. Links ein Laptop, in der Mitte der Mikrocontroller und rechts ein Tablet

Zeichnung von Nicola Gysler

Abb. 2: Arduino Board, *Arduino UNO*

<http://www.play-zone.ch/de/original-arduino-uno-r3-atmega328.html>

Abb. 3: Arduino Motor Shield *Dual VNH5019 Motor Driver Shield*

<https://www.pololu.com/product/2507>

Abb. 4: Erstes Motorpack mit schwächeren Motoren

Eigenaufnahme von Nicola Gysler

Abb. 5: Logo der Firma Einhell Schweiz AG

https://de.wikipedia.org/wiki/Einhell_Germany

Abb. 6: Akku-Bohrschrauber TE-CD 18 Li inkl. Akku

<http://www.akkuschrauber.com/einhell/te-cd-18-li/>

Abb. 7: Verwendete USB Flugsimulator-Fernsteuerung

<https://www.conrad.de/de/usb-flugsimulator-mit-gamecommander-205164.html>

Abb. 8: Kiditec-Bausteine (ein 1x1-Stein entspricht 6x6 cm)

http://www.tutti.ch/luzern/spielzeuge-basteln/spielzeuge/angebote/kiditec-schrauben-muttern-bausteine_12933947.htm

Abb. 9: Computermodellerte Veranschaulichung der mechanisch wichtigen Teile ohne die Raupe
modelliert von Nicola Gysler

Abb. 10: Andock-System der Kiditec-Elemente

Eigenaufnahme von Nicola Gysler

Abb. 11: Metallwinkel passen genau in die Rillen der Kiditec-Elemente

Eigenaufnahme von Nicola Gysler

Abb. 12: 3D-gedruckter Andock-Noppen

Eigenaufnahme von Nicola Gysler

Abb. 13: Trapezform der Raupe (Seitenansicht)

Bild gezeichnet von Nicola Gysler

Abb. 14: Gummiringmatte, 40x60 cm

<https://www.bauhaus.info/tuermatten-stufenmatten/ringmatte-/p/15853843>

Abb. 15: 3D-gedrucktes Antriebsrad

Eigenaufnahme von Nicola Gysler

Abb. 16: Antriebsrad greift in Raupe

Eigenaufnahme von Nicola Gysler

Abb. 17:Stück der Raupe mit Noppen

Eigenaufnahme von Nicola Gysler

Abb. 18. Holznoppen

Eigenaufnahme von Nicola Gysler

Abb. 19: Rad, gebaut aus Holz und Aluminiumrohr

Eigenaufnahme von Nicola Gysler

Abb. 20 und Abb. 21: Raupe wird in der Spur gehalten

Eigenaufnahmen von Nicola Gysler

Abb. 22: Rad auf Achse, beidseitig mit 3D-gedruckten Teilen befestigt.

Eigenaufnahme von Nicola Gysler

Abb. 23: Motormodul mit neuen Akkubohrern von Einhell

Eigenaufnahme von Nicola Gysler

Abb. 24: Windows-Firewall Zugriffsfreigabe

Screenshot

Abb. 25: Konfiguration Windows-Netzwerk

Screenshot

Abb. 26: Arduino Code im void loop()

Screenshot

6 Schlusswort

Der Arbeitsaufwand war deutlich höher als erwartet.

Obwohl im Verlauf meiner Arbeit immer wieder Überraschungen aufgetreten sind, hatte ich viel Spass daran und habe auch sehr viel dabei gelernt.

Auch wenn der Roboter durchaus noch Entwicklungspotential hat, erfüllt er doch die gesteckten Ziele.

Ein besonderer Erfolg war, dass es durch das Sponsoring von Einhell doch noch gelungen ist, WALL-LEE für das Treppensteigen fit zu bekommen.

7 Danksagung

Ich möchte mich ganz herzlich bei folgenden Personen für die grosse Unterstützung während meiner Maturaarbeit bedanken:

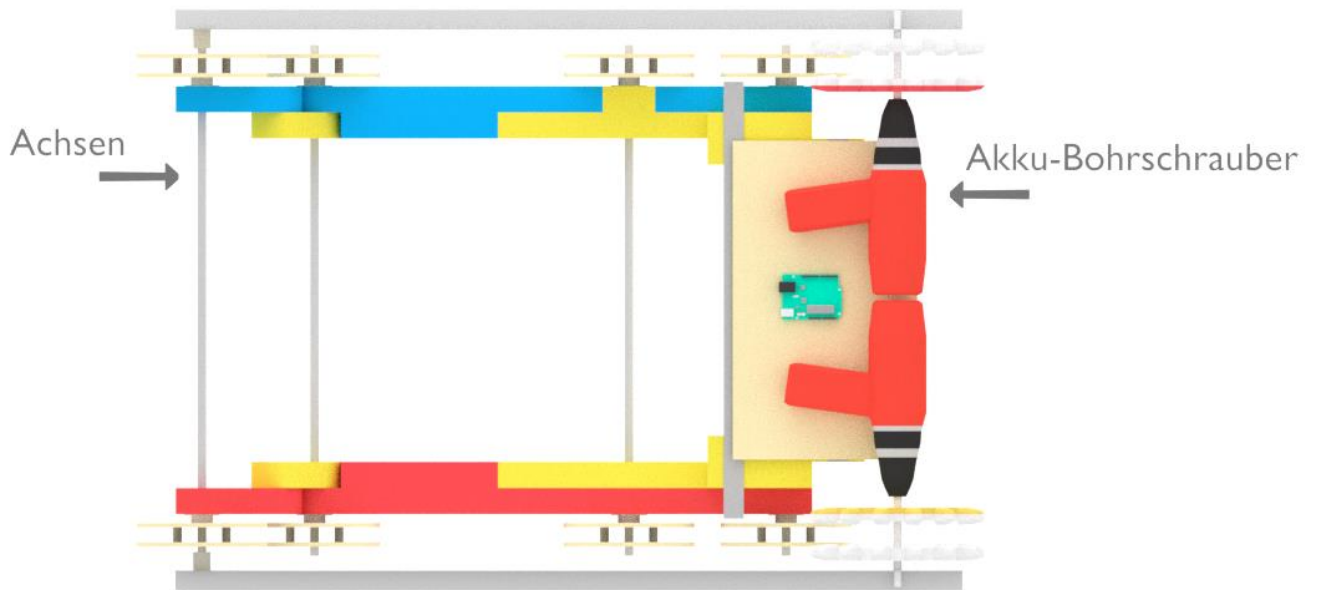
Meiner Betreuungsperson Wolfgang Pils für die kompetente Beratung und die wertvollen Tipps

Unserem Schulmechaniker Herrn Andreas Bertschi für seine fachliche Unterstützung und für die Möglichkeit, in seiner Werkstatt arbeiten zu können

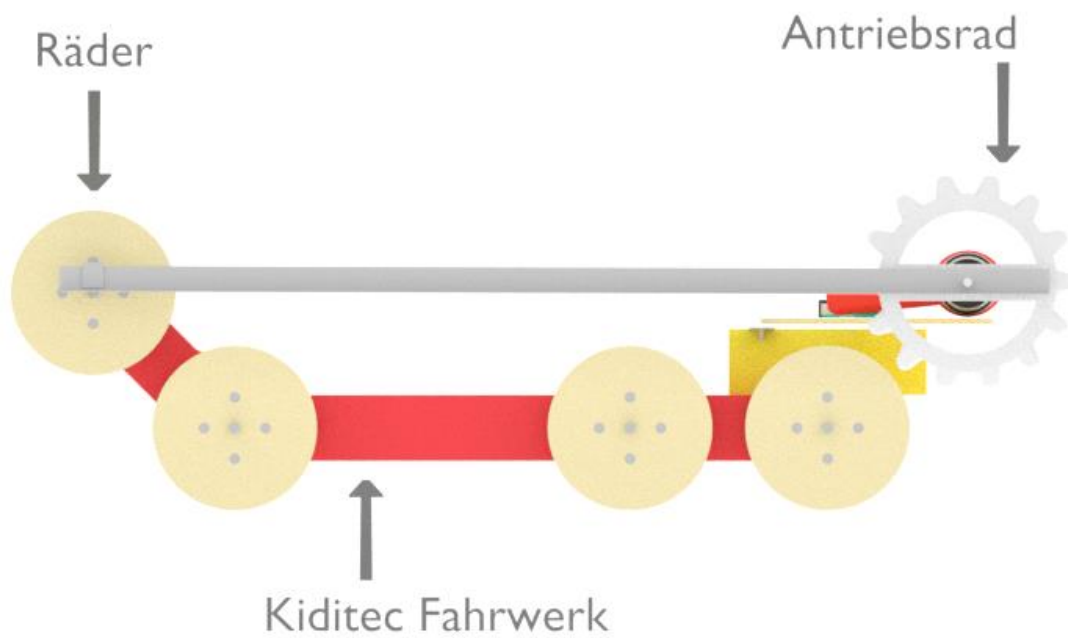
Meinem Vater Martin Gysler, der mir mit seinem Fachwissen mit Rat und Tat zur Seite stand und die Programmierung der C# Software übernommen hat

Meiner Mutter, Nadine Gysler für das ausdauernde Korrekturlesen und die aufmunternden Worte

Der Firma Einhell Schweiz für das grosszügige Sponsoring, insbesondere Frau Livia Baur für die freundliche Zusammenarbeit



Computermodell von WALL-LEE, Ansicht von oben



Computermodell von WALL-LEE, Ansicht von der Seite

Kapitel 8 : Anhang

Vollständiger Code für das Arduino Board

```
#include "DualVNH5019MotorShield.h"

DualVNH5019MotorShield md;

#include <ServoTimer2.h> // the servo library

ServoTimer2 servoPitch; // declare variables for up to eight servos
ServoTimer2 servoYaw;
ServoTimer2 servoYaw2;

int Yaw2 = 700;

void setup()
{
    Serial.begin(4800);

    #define INPUT_SIZE 34

    md.init();

    servoPitch.attach(3); // attach a pin to the servos and they will start pulsing
    servoYaw.attach(5);
    servoYaw2.attach(11);

    servoYaw2.write(700);

    pinMode(13, OUTPUT);
}

void loop()
{
    if (Serial.available() > 0)
    {
        // Get next command from Serial (add 1 for final 0)
        char input[INPUT_SIZE + 1];
        byte size = Serial.readBytes(input, INPUT_SIZE);
        // Add the final 0 to end the C string
        input[size] = 0;

        // Read each command pair
        char* command = strtok(input, "&");
        while (command != 0)
        {
            // Split the command in two values
            char* separator = strchr(command, ':');
            if (separator != 0)
            {
                // Actually split the string in 2: replace ':' with 0
                *separator = 0;
                int ValueID = atoi(command);
                ++separator;
            }
        }
    }
}
```

Kapitel 8 : Anhang

```
int Value = atoi(separator);

if (ValueID == 1){           // Do something with ID and Value
md.setM1Speed(Value);
}
if (ValueID == 2){
md.setM2Speed(Value);
}

if (ValueID == 3){
servoYaw.write(Value*-2+1500);

    if (Value >= 350 ){

        if (Yaw2 >= 700)
Yaw2 = Yaw2 - 11;

        servoYaw2.write(Yaw2);

    }

    if (Value <= -350){

        if (Yaw2 <= 2200)
Yaw2 = Yaw2 + 11;

        servoYaw2.write(Yaw2);

    }

}
if (ValueID == 4){
servoPitch.write(Value*-2+1500);
}

if (ValueID == 5){
int Switch1 = Value % 10;
int Switch2 = (Value / 10) % 10;
int Switch3 = (Value / 100) % 10;
int Switch4 = (Value / 1000) % 10;

    if (Switch1 == 1){
digitalWrite(13, HIGH);
}
    else{
digitalWrite(13, LOW);
}

}

}
// Find the next command in input string
command = strtok(0, "&");
}
}
}
```