**DIPLOMARBEIT**

Gesamtprojekt

**Camera Controlled Swarm Robots**

**Positionserkennung**

Mottl Mario 5BHEL Betreuer: Dipl.-Ing Josef Reisinger

**Swarm Controll**

Clemens Pruggmayer 5BHEL Betreuer: Dipl.-Ing Josef Reisinger

**Visualisierung und Simulation**

Michael reim 5BHEL Betreuer/in: Dipl.-Ing Josef Reisinger

Schuljahr 2020/21

Abgabevermerk:

Datum: TT.MM.JJJJ übernommen von:

****

**Höhere Technische Bundeslehranstalt Hollabrunn**

**Höhere Lehranstalt für Elektronik und Technische Informatik**

**EIDESSTATTLICHE ERKLÄRUNG**

**Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.**

**Mario Mottl**

**Clemens Pruggmayer**

**Michael Reim**

Hollabrunn, am 5. April 2019

**DIPLOMARBEIT**

**DOKUMENTATION**

|  |  |
| --- | --- |
| Namen der  Verfasser/innen | Michael Reim, Clemens Pruggmayer, Mario Mottl |
| Jahrgang  Schuljahr | 5BHEL |
| Thema der Diplomarbeit | Camera Controlled Swarm Robots |
| Kooperationspartner | - |

|  |  |
| --- | --- |
| Aufgabenstellung | Es sollen mehrere 1 bis n viele autonome Fahrzeuge (STM32F107RB + MDDS Board) über eine 1,5m erhöhten Kamera erfasst werden. Positionen der Fahrzeuge sollen in (x/y) Koordinaten verwandelt werden. Diese Daten sollen an die Visualisierung und Simulation geschickt. Diese erzeugt über selbst gezeichnete Bilder einen Weg für die Fahrzeuge. Der Weg soll an das Swarm Controll weitergeschickt werden welche die Positionsdaten mithilfe der Kamera in Fahrkommandos umwandelt. |

|  |  |
| --- | --- |
| Realisierung | Zur Erfassung der Autos wurde eine „DFK 33UX273“ von ImagingSource verwendet. Diese wird per USB an einen Laptop/Computer angeschlossen. Für die Visualisierung und Simulation wurde eine 3D Visualisierungssoftware in C++ geschrieben. Welche die Autos in Echtzeit am Bildschirm anzeigt.  Swarm Controll wurde mithilfe des SvVis Protokoll realisiert. Auf den Autos(Cortex M4) läuft eine selbstgeschriebene RTOS-Software, die die Kommandos des SVIS Protokoll in Bewegungen umwandelt. |

|  |  |  |
| --- | --- | --- |
| Ergebnisse | Eine alte Version des SvVis Protokolls wurde abgeändert und verbessert. Eine Teststrecke wurde aufgebaut und mit einer Halterung für die Kamera erweitert. Positionserkennung wurde mithilfe von Python realisiert. Die Autosteuerungssoftware wurde selbständig mit RTOS realisiert. Visualisierung wurde von Grund auf in C++ + OpenGL geschrieben für den speziellen Anwendungsfall. |  |

|  |  |
| --- | --- |
| Typische Grafik, Foto etc.  (mit Erläuterung) | Die obige Grafik stellt den Aufbau des Gesamtsystems dar. Darin sieht man die 3 Hauptkomponenten Kamera + Kamerasoftware, Visualisierung und Swarm Controll. Daten zwischen den einzelnen Blöcken wir über TCP bereitgestellt. Verbindung zu den Autos kann auf zwei Wege bereitgestellt werden: Bluetooth und Wlan. |

|  |  |
| --- | --- |
| Teilnahme an Wettbewerben,  Auszeichnungen |  |

|  |  |
| --- | --- |
| Möglichkeiten der Einsichtnahme in die Arbeit | HTL Hollabrunn  Anton Ehrenfriedstraße 10  2020 Hollabrunn |

|  |  |  |
| --- | --- | --- |
| Approbation  (Datum / Unterschrift) | Prüfer/Prüferin | Direktor/Direktorin  Abteilungsvorstand/Abteilungsvorständin |

**DIPLOMA THESIS**

**Documentation**

|  |  |
| --- | --- |
| Author(s) | Michael Reim, Clemens Pruggmayer, Mario Mottl |
| Form  Academic year | 5BHEL |
| Topic | Camera Controlled Swarm Robots |
| Co-operation partners | - |

|  |  |
| --- | --- |
| Assignment of tasks | One or more autonomous vehicle (powered by an STM32F107RB + MDDS Board) should get detected by a camera that is attached 1,5 metres above the table. The positions of the vehicles should be converted into an x-y coordinate grid. The produced data should then be sent to our visualisation and simulation for correction purposes. A picture of a path should be drawn. The generated path should then be sent off to the Swarm Control. Where the path will be transformed into vehicle commandos. |

|  |  |  |
| --- | --- | --- |
| Realisation | For proper detection, an “DFK 33UX273” from Imaging Source was used. It is connected via USB to a Laptop. The visualisation and simulation were programmed in “C++” + “OpenGl” which draws the cars onto the screen in Realtime. Swarm Control uses a technology called “SvVis”. The vehicles run on a self-implemented RTOS-Software (Real Time Operating System) which transforms the vehicle commandos into engine movement. |  |

|  |  |
| --- | --- |
| Results | An older version of the “SvVis” was used and altered to fit our purposes. A test track with a bracket for the camera was built. The tracking software was programmed in python. Vehicle Control software was written with RTOS. The visualisation and simulation were programmed in “C++” + “OpenGl” for out specific purpose. |

|  |  |
| --- | --- |
| Illustrative graph, photo  (incl. explanation) | The graph above pictures the whole System. There you can see the three main components. Tracking, Swarm Control and Visualisation and Simulation. The communication between the individual blocks is realised in TCP. The communication between laptop and cars can be established in two ways WLAN, Bluetooth. |

|  |  |
| --- | --- |
| Participation in competitions  Awards | - |

|  |  |
| --- | --- |
| Accessibility of  final project thesis | HTL Hollabrunn  Anton Ehrenfriedstraße 10  2020 Hollabrunn |

|  |  |  |
| --- | --- | --- |
| Approval  (Date / Signature) | Examiner/s | Head of Department / College |

DA Antrag und unterschriebene Erklärung aus der Diplomarbeitsdatenbank einfügen

Inhaltsverzeichnis

[1 Camera Controlled Swarm Robots 12](#_Toc64022130)

[1.1 Blockschaltbild 12](#_Toc64022131)

[1.2 Funktionsbeschreibung 12](#_Toc64022132)

[2 Positionserkennung 13](#_Toc64022133)

[2.1 Übersicht 13](#_Toc64022134)

[2.2 Aufbau 13](#_Toc64022135)

[2.3 Kamera Ansteuerung 13](#_Toc64022136)

[2.4 Erkennung Positions LEDs 13](#_Toc64022137)

[2.4.1 Algorithmus zur Erkennung 14](#_Toc64022138)

[2.4.2 Berechnung der Position 15](#_Toc64022139)

[2.5 Auswertung der Bilddaten 15](#_Toc64022140)

[2.6 Kommunikation mit Simulation / Visualisierung 15](#_Toc64022141)

[3 Visualisierung und Simulation 17](#_Toc64022142)

[3.1 Übersicht Softwarearchitektur 17](#_Toc64022143)

[3.2 Aufbau der Engine 17](#_Toc64022144)

[3.3 Erstellung von 3D-Modellen 17](#_Toc64022145)

[3.4 Bewegungen im Dreidimensionalen Raum 17](#_Toc64022146)

[3.5 Auswertung von erhaltenen Positionsdaten 17](#_Toc64022147)

[3.6 Übertragung zu Swarm Controll 17](#_Toc64022148)

[4 Swarm Controll 18](#_Toc64022149)

[4.1 Übersicht Softwarearchitektur 18](#_Toc64022150)

[4.2 Berechnung des Weges 18](#_Toc64022151)

[4.3 Konvertierung von Positionen auf Steuerbefehle (WIP) 18](#_Toc64022152)

[4.4 Steuerbefehle 18](#_Toc64022153)

[4.4.1 Aufbau von Steuerbefehlen (Protokollaufbau) 18](#_Toc64022154)

[4.4.2 Senden von Steuerbefehlen (API) 18](#_Toc64022155)

[5 Fahrzeug Software 19](#_Toc64022156)

[5.1 Übersicht der Architektur 19](#_Toc64022157)

[5.2 Kommunikation mit Swarm Controll 19](#_Toc64022158)

[5.2.1 Kommunikationsprotokoll 19](#_Toc64022159)

[5.3 Hardware Ansteuerung 19](#_Toc64022160)

[5.3.1 LED-Ansteuerung 19](#_Toc64022161)

[5.3.2 Kommunikationsmodule 19](#_Toc64022162)

[5.3.2.1 Bluetooth 19](#_Toc64022163)

[5.3.2.2 WLAN 19](#_Toc64022164)

[5.3.3 Motor Ansteuerung 19](#_Toc64022165)

[6 Ergebnisse 20](#_Toc64022166)

[6.1 Funktionalität Positionserkennung 20](#_Toc64022167)

[6.2 Steuerung der Fahrzeuge 20](#_Toc64022168)

[6.3 Simulationstest mit Pseudodaten 20](#_Toc64022169)

[6.4 Steuersoftwarte Funktionalitätstest 20](#_Toc64022170)

[6.5 Schwarmbewegung 20](#_Toc64022171)

[6.5.1 Kreis 20](#_Toc64022172)

[6.5.2 Zick-Zack 20](#_Toc64022173)

[7 Anhang 21](#_Toc64022174)

[7.1 Inbetriebnahme (f. 4Klasse TdoT) 21](#_Toc64022175)

[7.2 Projektmanagement 21](#_Toc64022176)

[7.2.1 Projektplan 21](#_Toc64022177)

[7.2.2 Projekttagebuch 21](#_Toc64022178)

[7.2.3 Projektkosten 21](#_Toc64022179)

[8 Quellenverzeichnis 22](#_Toc64022180)

[8.1 Bücher 22](#_Toc64022181)

[8.2 Onlinemedien 22](#_Toc64022182)

[8.3 Zeitschriften 22](#_Toc64022183)

Wenn Sie in Ihrer Dokumentation ausschließlich mit den „Formatvorlagen“ dieses Dokuments gearbeitet haben können Sie hier sehr einfach das Inhaltsverzeichnis automatisch aktualisieren lassen.

Klicken Sie dazu mit der rechten Maustaste in das vorhandene Inhaltsverzeichnis und wählen Sie im Kontextmenü „Felder aktualisieren“ und dann „gesamtes Verzeichnis aktualisieren“ auswählen – fertig! … diesen Text löschen!

# Camera Controlled Swarm Robots

## Blockschaltbild

## Funktionsbeschreibung

# Positionserkennung

## Übersicht

## Aufbau

## Kamera Ansteuerung

## Erkennung Positions LEDs

Für die Erkennung der Positions LEDs wird ein simpler Algorithmus verwendet. Durch die spezielle Anordnung der LEDs in einem gleichschenkeligen Dreieck kann man die ungefähre Position der anderen LEDs mit dem Satz des Pythagoras berechnen.



Um sicherzustellen, dass die Positions LEDs zu einem Auto und nicht zu einem anderen gehören. Werden immer nur zwei LEDs mit der Kamera erkannt und die Position der letzten LED wird berechnet.

### Algorithmus zur Erkennung

Für den Algorithmus wird der in OpenCV eingebaute Filter + Detektor verwendet. Die Parameter werden wie folgt gesetzt.

params = cv2.SimpleBlobDetector\_Params()

# Change thresholds

params.minThreshold = 0

params.maxThreshold = 256

# Filter by Area.

params.filterByArea = True

params.minArea = 30

# Filter by Circularity

params.filterByCircularity = True

params.minCircularity = 0.1

# Filter by Convexity

params.filterByConvexity = True

params.minConvexity = 0.5

# Filter by Inertia

params.filterByInertia = True

params.minInertiaRatio = 0.5

detector = cv2.SimpleBlobDetector\_create(params)

Diese Klasse ist besonders gut geeignet um blobs (zusammenhängende Bildpunkte mit derselben Farbe und Intensität). Diese Klasse implementiert einfache Algorithmen, um diese zu extrahieren.

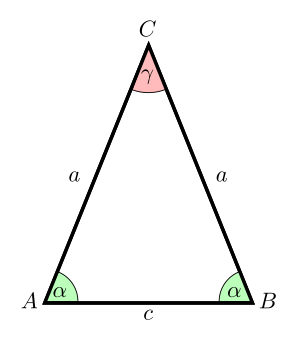
Konvertierung des Quellbildes in Binärbild, indem man einen Schwellenwert mit mehreren  
Schwellenwerten von minThreshold (einschließlich) bis maxThreshold (exklusiv) mit einem Abstand Threshholdstep zwischen benachbarten Schwellenwerten anwendet.

Extrahieren von verbundenen Komponenten aus jenem Binärbild mit findContours und berechnen von deren Zentren.

Gruppierung von Zentren aus mehreren Binärbildern anhand ihrer Koordinaten. Nahe Zentren bilden eine Gruppe, die einem Blob entspricht, der vom Parameter minDistBetweenBlobs gesteuert wird.

Schätzen aus den Gruppen die endgültigen Zentren der Blobs und ihre Radien und geben diese als Positionen und Größen der Schlüsselpunkte zurück.

### Berechnung der Position



Um die Seitenlängen zu bestimmen wurde einmal als Referenz a & c in Pixel gemessen.

Es wird weiteres als Kontrolle der Winkel α & γ berechnet.

Darüber hinaus wird in regelmäßigen Abständen die Seitenlänge neu berechnet. Um sicherzustellen, dass es sich noch immer um dasselbe Auto handelt.

## Auswertung der Bilddaten

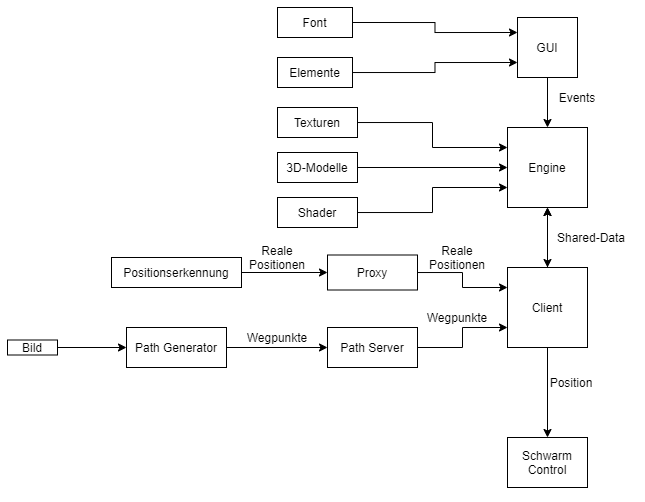
Um aus den vielen Bilddaten eine vernünftige Position zu generieren wird der Mittelpunkt des Autos berechnet. Jede Koordinate einer Positions LED wird in einem Tupel (x, y) (posLed) abgespeichert. Jedes Auto (cararray) besteht aus einem Array aus drei Tupel (posLed1, posLed2, posLed3) + Fahrtrichtung. Über diese drei Tupel wird gemittelt, um den Mittelpunkt des Fahrzeugs zu erhalten.

## Kommunikation mit Simulation / Visualisierung

# Visualisierung und Simulation

## Übersicht Softwarearchitektur

### Blockschaltbild



### GUI

Das GUI ist die graphische Oberfläche der Visualisierung. Es enthält Text und verschiedene Elemente wie zum Beispiel eine Eingabezeile. Die Schriftart, die verwendet wird, wird über ein File in das Programm eingelesen. Mehr zu diesem Thema gibt es im Kapitel 3.2 (Graphical User Interface).

### Engine

Die Engine ist das Rückardt der Software. Sie beinhaltet das graphische Rendering, steuert Bewegungen und den Ablauf der Simulation und reagiert auf Events. Dazu wird im Kapitel 3.3 (Aufbau der Engine) näher eingegangen.

### Client

Der Client ist für das Empfangen und Senden von Daten über das Netzwerk zuständig. Da die Engine mit dem Client kommunizieren muss, wird ein Shared-memory verwendet. Auf diesen Speicher haben nur die Engine und der Client Zugriff.

### Modelle, Texturen, Shader

Modelle beschreiben die Form der Objekte in einer Rendering-Szene. Sie beinhalten Raumpunkte, genannt Vertices, Normalvektoren und Texturkoordinaten.  
Texturen werden verwendet, um Modellen Farben zu geben. Diese sind Bild-Files, welche in den Speicher geladen werden, und anschließend am Modell angezeigt werden. Welcher Bereich einer Textur auf einem Modell angezeigt wird, wird über die Texturkoordinaten gesteuert.

Shader sind Programme, die auf der Graphikkarte ausgeführt werden. Sie berechnen mithilfe der Vertices, Texturkoordinaten und Normalvektoren die endgültige Farbe eines Pixels auf dem Bildschirm. Dieser Prozess wird „Shading“ genannt. Genauer erläutert wird das im Kapitel 3.3 (Aufbau der Engine) und 3.4 (Erstellung von 3D-Modellen).

### Path Generierung

Die Generierung eines Weges erfolgt aus einem Bild-File. Der Anwender kann mit einem Programm, womit man Bild-Files editieren kann, einen Beliebigen weg auf das Bild zeichnen. Dieses Bild wir anschließend vom „Path Generator“ eingelesen und in Wegpunkte untereilt, welche die Fahrzeuge abfahren. Diese Wegpunkte und dessen Nummer erhält der „Path Server“. Er ist für die Aufbereitung und für das Senden der Wegpunkte zur Visualisierung zuständig.  
Wenn die realen Fahrzeuge angesteuert werden, werden neben den generierten Wegpunkten, noch zusätzliche Positionen von der Positionserkennung empfangen. Diese Positionen sind die Koordinaten der abgefilmten Fahrzeuge und werden verwendet, um die Bahnen der virtuellen Fahrzeuge in der Visualisierung zu korrigieren.

Da die Positionserkennung Daten in der falschen Byteorder sendet, ist ein Proxy zwischen der Positionserkennung und der Visualisierung, der die Byteorder umdreht und weitersendet.

Genauer wird im Kapitel 3.6 (Auswertung von erhaltenen Positionsdaten) auf dieses Thema eingegangen.

## Graphical User Interface

## Aufbau der Engine

## Erstellung von 3D-Modellen

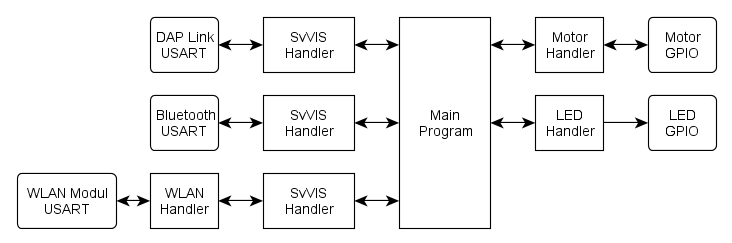
## Bewegungen im Dreidimensionalen Raum

## Auswertung von erhaltenen Positionsdaten

## Übertragung zu Swarm Controll

# Fahrzeug Software

## Übersicht der Architektur



## Kommunikation mit Swarm Controll

Ein Fahrzeug kann über 3 verschiedene Wege mit der Außenwelt kommunizieren:

1. Serielle Schnittstelle über DAP-Link Adapter
2. Serielle Schnittstelle über Bluetooth mittels HC06 Modul
3. TCP/IP Kommunikation mittels ESP8266 Modul

Da die Kommunikation über den DAP-Link Adapter nur für Entwicklungs- und Testzwecke verwendet werden kann, wird die Kommunikation im Endaufbau über Bluetooth oder WLAN stattfinden.

### Kommunikationsprotokoll

Die Software des Fahrzeugs kommuniziert über das SvVis Protokoll mit der Swarm Control. Dieses Protokoll besteht aus einem Header mit der Größe von 1 Byte und darauffolgenden Nutzdaten. Die länge der Nutzdaten ist vom Header und den Daten selbst abhängig.

|  |  |  |
| --- | --- | --- |
| Header Byte(s) | Bedeutung | Länge der Daten |
| 0x0A  10 | String | Daten werden mit '\0' beendet. Implementierung ist auf eine String-Länge von 31 begrenzt |
| 0x0B-0x13  11-19 | 16-bit integer (int16\_t) | 2 Bytes  16 Bit |
| 0x15-0x1D  21-29 | 32-bit floating point number (float) | 4 Bytes  32 Bit |

## Hardware Ansteuerung

### LED-Ansteuerung

Die Ansteuerung der LEDs ist die simpelste Hardware-Ansteuerung des Fahrzeuges. Es gibt eine Led, die periodisch Blinkt, um eine visuelle Bestätigung zu geben, dass die Software des Autos aktiv ist. Außerdem kann die Software die 3 Positionserkennungsleds einzel ein- und ausschalten.

### Kommunikationsmodule

#### DAP Link

Die serielle Schnittstelle über den DAP\_Link ist hauptsächlich für die Softwareentwickling verwendbar, da ein USB-Kabel für die Kommunikation benötigt wird.

#### Bluetooth

Das verwendete HC06 Modul erlaubt kabellose Datenübertragung mit der Fahrzeugsoftware. Allerdings ist die Reichweite dieser Verbindung nicht sehr groß

#### WLAN

Das ESP8266 Modul ermöglicht die Kommunikation über eine TCP-Verbindung, die die TCP-Pakete über WLAN sendet und empfängt.

Allerdings müssen dazu einige Details über die zu verwendende Verbindung in die Konfigurationsdatei des Programms Eingetragen werden:

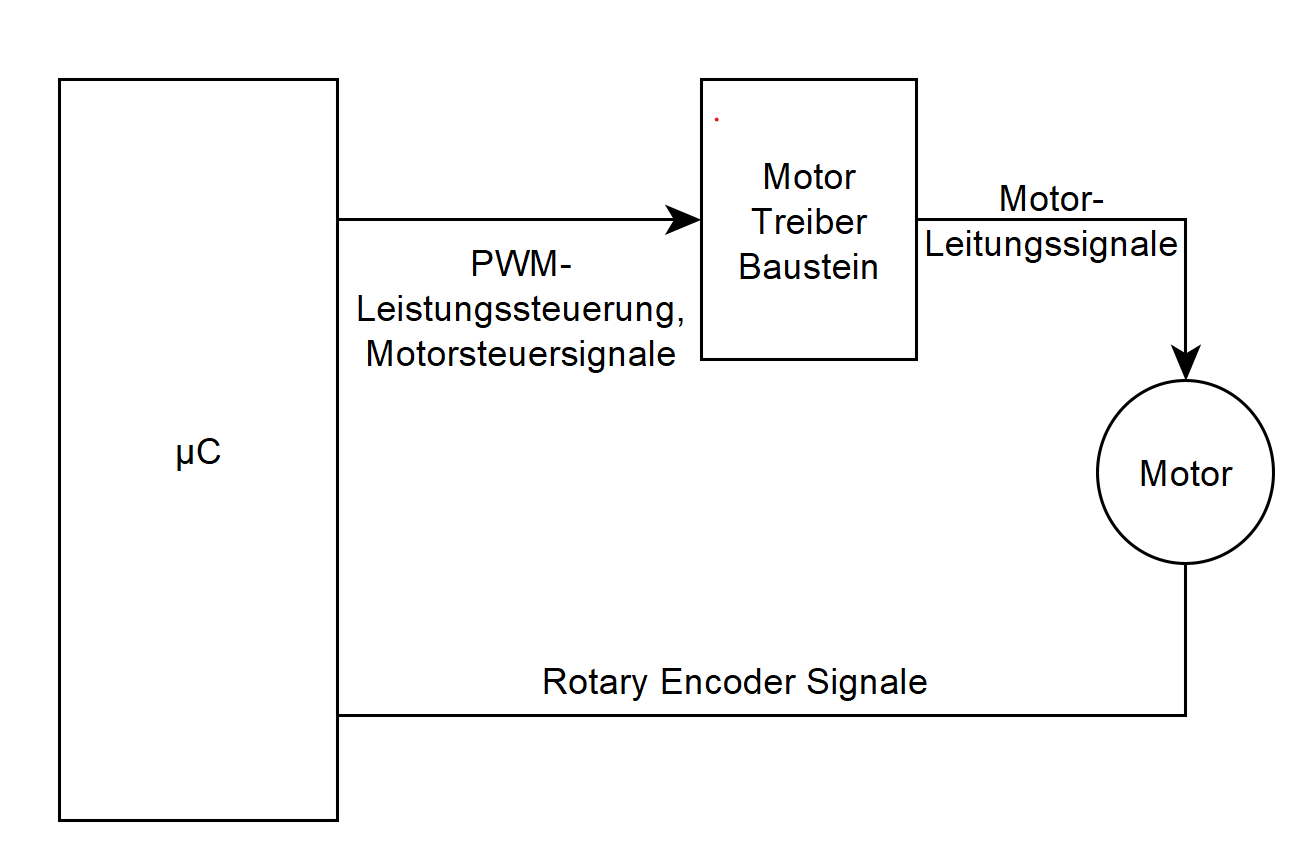
* SSID des WLAN Access Points
* Password für das WLAN-Netzwerk
* IP Adresse der Swarm Control
* TCP Port der Swarm Control

Die Ansteuerung des WLAN Moduls erfolgt über AT-Kommandos über eine Serielle Schnittstelle.

### Motor Ansteuerung

Die Ansteuerung der Motoren ist einer der wichtigsten Komponenten der Auto-Software, da dieser Teil dafür verantwortlich ist, dass sich aus Auto bewegt.

Die Motoren werden mithilfe eines Motortreiberbausteins angetrieben. Dieser Motortreiberbaustein bietet die Möglichkeit, die Motorleistung mit einem PWM Signal zu regeln. Der Verwendete Mikrocontroller (STM32F107RB) bietet eine Möglichkeit, ein PWM-Signal mit einstellbaren duty-cycle über die Integrierten Timer-Peripherie Komponenten generieren zu lassen. Die Motoren haben zusätzlich Inkrementalgeber, um die derzeitige Drehzahl des Motors auslesen zu können.



Eine Verbesserung dieses Designs wäre ein Hardwaremäßiger Rotary Decoder Baustein, der die Decodersignale auswertet und dadurch Arbeit des Microcontrollers abnimmt.

Eine weiter Möglichkeit wäre Schrittmotoren zu verwenden, da Schrittmotoren keine Drehzahlregelung mit Rückmeldung benötigen. Außerdem Kann mit Schrittmotoren eine wesentlich genauere Positionierung des Fahrzeugs erreicht werden.

# Fahrzeug Software

## Übersicht der Architektur

## Kommunikation mit Swarm Controll

### Kommunikationsprotokoll

## Hardware Ansteuerung

### LED-Ansteuerung

### Kommunikationsmodule

### Bluetooth

### WLAN

### Motor Ansteuerung

# Ergebnisse

## Funktionalität Positionserkennung

## Steuerung der Fahrzeuge

## Simulationstest mit Pseudodaten

## Steuersoftwarte Funktionalitätstest

## Schwarmbewegung

### Kreis

### Zick-Zack

# Anhang

## Inbetriebnahme (f. 4Klasse TdoT)

## Projektmanagement

### Projektplan

### Projekttagebuch

### Projektkosten

# Quellenverzeichnis

## Bücher

## Onlinemedien

## Zeitschriften