Gymnasium Wolbeck

Raumanalyse

mit einer Quadrokopter

Alexander Neuwirth, Jonathan Sigrist

Jugend Forscht 2016

Inhaltsverzeichnis

[1 Wer sind wir? 3](#__RefHeading__369_487429023)

[2 Einleitung 3](#__RefHeading__3317_560598563)

[2.1 Verlauf – Kurzbeschreibung 3](#__RefHeading__505_487429023)

[3 Vorüberlegungen 4](#__RefHeading__518_487429023)

[3.1 Erste Planungen und Ideen 4](#__RefHeading__653_1259517275)

[3.2 Beschaffen der Drohne und Installation der Software 4](#__RefHeading__524_487429023)

[4 Das Projekt organisieren 5](#__RefHeading__655_1259517275)

[4.1 Testen der Drohne 5](#__RefHeading__603_1341788920)

[4.2 Erstes Programmieren der Drohne 5](#__RefHeading__605_1341788920)

[4.3 Die Programmierelemente von node.js 5](#__RefHeading__657_1259517275)

[5 Die Drohne programmieren 6](#__RefHeading__661_1259517275)

[5.1 Steuerung über die Konsole 6](#__RefHeading__663_1259517275)

[5.2 Eine Website für Interaktion mit dem Anwender 6](#__RefHeading__665_1259517275)

[5.3 Entfernungsmessung mit Ultraschall 6](#__RefHeading__667_1259517275)

[5.4 Bewegungsbefehle 7](#__RefHeading__669_1259517275)

[5.5 Bildanalyse 7](#__RefHeading__633_1341788920)

[6 Das Darstellungsprogramm 8](#__RefHeading__671_1259517275)

[7 Probleme und Lösungen 8](#__RefHeading__607_1341788920)

[8 Ausblick 8](#__RefHeading__609_1341788920)

[8.1 Automatische Raumanalyse 8](#__RefHeading__635_1341788920)

[8.2 Erster Rundumtest 8](#__RefHeading__611_1341788920)

[8.3 Grundidee zu den Hindernissen 8](#__RefHeading__675_1259517275)

[9 Zusammenfassung 8](#__RefHeading__615_1341788920)

[10 Vorschriften für die Nutzung von Quadrocoptern 9](#__RefHeading__637_1341788920)

[11 Literaturverzeichniss/Quellen 9](#__RefHeading__617_1341788920)

# Wer sind wir?

Wir sind zwei Schüler der Jahrgangsstufe Q2 am Gymnasium Wolbeck. Beide haben wir Interesse an der Informatik, der Physik und der Technik und arbeiten auch mal außerhalb der Schulzeit gerne an Projekten. Wir sind 17 Jahre alt und haben beide schon mal unabhängig von einander an einem Jugend Forscht Wettbewerb teilgenommen.

Alexander Neuwirth hat im Jahr 2015 mit einer LEGO®-Mindstorms Roboter Programmierung bei Jugend Forscht teilgenohmen.

Jonathan Sigrist hat bereits in einem der letzten Jahre bei Jugend Forscht und anderen Wettbewerben mit einem Projekt zur effizienten Nutzung von Solarzellen in der Physik beigetragen und einen Sonderpreis für Umweltbewusstsein erlangt. Seine Facharbeit hat er im Fach Physik geschrieben. Bereits mehrere Male hat er sich an außerschulischen Camps beteiligt.

# Einleitung

Ziel des Projektes ist die Vermessung und Darstellung eines geschlossenen Raumes mit Hilfe einer Parrot® AR.Drone 2.0.

Die Steuerung der Drohne soll mittels des *ar-drone* Moduls in Node.js® realisiert werden.

Die Drohne erhält Steuerkommandos von einem Computer über eine WLAN-Verbindung und übermittelt ihrerseits Daten an diesen, beispielsweise einen Videostream von der integrierten HD-Kamera. Empfangene Daten sollen dann vom Computer ausgewertet und zu einer Darstellung des Raums aufbereitet werden.

Des Weiteren soll der Quadrokopter im Auto-Modus, d.h. möglichst ohne durch Menschen gesteuert zu werden, den Raum analysieren und Hindernissen ausweichen können.

Am Ende soll eine vollständige, möglichst fehlerfreie und automatische Überprüfung eines Raumes möglich sein.

## Verlauf – Kurzbeschreibung

Am Anfang haben wir strukturiert die Module und Sensoren der Drohne getestet. Des Weiteren haben wir uns in Javascript fortgebildet um node.js nutzen zukönnen.

Die Drohne erstellt beim Anschalten ein WLAN-Netz, in welches sich ein Computer mit WLAN-Schnittstelle einloggen kann. Der Computer kann dann Befehele an die Drohne schicken und umgekehrt auch Navigations- und Sensordata erhalten.

Welche Befehle geschickt werden sollen wird über einen Browser kontrolliert, d.h. ein Programm auf dem Computer ist ein HTTP-Server, der die gewünschten Kommandos des Anwenders an die Drohne weiterleitet und die Sensordaten aufbereitet und für den Anwender darstellt.

Hauptbestandteil der Website ist ein Videostream der Frontkamera oder Bodenkamera der Drohne, welcher gleichzeitig analysiert wird.

Auch wird die Position und die zurückgelegte Strecke der Drohne auf der Website dargestellt.

# Vorüberlegungen

## Erste Planungen und Ideen

Unsere Idee ist die vollautomatische Analyse eines Raum mittels einer Drohne, jedoch sollte die Drohne vorerst auch noch von einem Anwender kontrolliert werden, um Unfälle zu vermeiden

Dazu war das von Alexander durchgeführte Projekt mit dem LEGO®-NXT Roboter natürlich eine gute Vorlage. Allerdings ging es dort hauptsächlich um die Verbindung und die allgemeine Steuerbarkeit und nicht um die Automatisierung und Analyse erhaltener Daten.

Nach längerer Recherche im Internet haben wir und für die AR.Drone 2.0 von Parrot® entschieden, da sie mittels Node.js einfach ansteuerbar ist und gute Flugeigenschaften besitzt. Außerdem hat sie einen USB-Anschluss zum Datenaustausch. Auch ist die Reichweite der Drohne lediglich an die eines WLAN-Netzes gebunden, somit lässt sich diese durch einen WLAN-Router schnell vergrößern.

Den Utraschallsensor der AR.Drone wollten wir nach vorne richten, um damit einen Raum zu vermessen. Mit diesen Daten kann man auf einem Computer den Grundschnitt des gesammten Raumes grafisch anzeigen lassen. Ebenso könnte der Anwender die Drohne an einen beliebigen Punkt im Raum schicken. Die Drohne fliegt dann eine geeignete Flugroute zum Zielort ohne mit Hindernissen zusammen zu stoßen.

## Beschaffen der Drohne und Installation der Software

Wir haben uns für eine AR.Drone von Parrot® entschieden. Um die Kosten möglichst klein zu halten, haben wir uns eine gebrauchte, generalüberholte Drohne gekauft. Dazu noch einen weiteren Akku mit 2000mAh, um längere Testflüge zu absolvieren. Der mitgelieferte Akku hatte nur eine Laufzeit von zwölf Minuten und so kann man mit einem Akku fliegen, während man den anderen aufladen kann.

Mit der kostenlosen Handy/Tablet-App "AR.FreeFlight" von Parrot® kann die Drohne sehr leicht geflogen werden. Um sie mit dem Computer zu steuern, ist Node.js und das frei verfügbares Modul *ar-drone* von Felix Geisendörfer benötigt. Als Editor haben wir notepad++ und vim genutzt. Zum synchronieren des Projektes ist git zum Einsatz gekommen.

# Das Projekt organisieren

## Testen der Drohne

Als die Drohne angekommen ist, haben wir die Flugeigenschaften mit Hilfe der Handy-App getestet und ein Gefühl dafür entwickelt, wie man die Drohne ansteuern muss, um ein genaues Fliegen zu gewährleisten. Die App wurde auch später oft verwendet, um Probleme aufzudecken und zu beheben.

## Erstes Programmieren der Drohne

Ein wichtiger erster Schritt war der, die Drohne über den Computer zu kontrollieren. Dafür reichte schon die direkte Implementierung des Moduls in eine JavaScript Datei, welche dann mit Node.js ausgeführt worden ist. Über die Konsole konnten direkte Befehle an die Drohne geschickt werden (zum Beispiel: "takeoff()", "forward(0.1)").

## Die Programmierelemente von node.js

Um mit node.js arbeiten zu können, müssen sowohl die Programmiersprache, als auch die Verwendungsöglichkeiten klar sein. Es handelt sich bei Node.js um JavaScript, was auf dem Server und nicht auf dem Client läuft. Daraus ergeben sich Verwaltungsvorteile, welche mit normalem JavaScript nicht leicht zu erreichen sind. Insbesondere die asynchrone Callback-Funktion und die einfache Möglichkeit eine Website mit in das Programm einzubinden sind hier von Vorteil.

Um auf die Drohne zuzugreifen, muss das Modul *ar-drone* in das Projekt eingebunden werden. Danach können alle Funktionen frei genutzt werden. Das Modul der Drohne gibt desweiteren die Möglichkeit auf einen Videostream von der Drohne in Echtzeit zuzugreifen. Um diesen im Browser in Echtzeit darzustellen wird ein das Modul *dronestream*, das auf *ar-drone* aufbaut, verwendet.

Auch sollten Befehle vom Webinterface möglichst in Echtzeit von der Drohne ausgeführt werden. Das Modul *socket.io* bietet die Möglichkeit einer intuitiven Implementation von WebSockets und somit einer direkten Datenübertragung vom Browser zum Server. Des Weiteren wird so auf Client- sowie Serverseite in Javascript programmiert.

# Die Drohne programmieren

In dem folgenden Kapitel soll erklärt werden, wie die Drohnensteuerung programmiert ist und wie sich durch ein Programm oder durch einen Benutzer komplexe Abläufe realisieren lassen. Dabei kann auch später auf Grundfunktionen zurückgegriffen werden.

## Steuerung über die Konsole

Wie vorher bereits erwähnt lässt sich die Drohne über die Konsole steuern. Damit konnten sich schon einfache Programme schreiben lassen, welche die Drohne starten, einmal hochfliegen und wieder landen lassen.

## Eine Website für Interaktion mit dem Anwender

Für die weitere Arbeit war eine andere Benutzeroberfläche nötig. Eine Website bietet zwei weitere große Vorteile. Zum einen kann der Videostream abgebildet werden und die spätere grafische Darstellung der Daten ist so auch deutlich einfacher. Zum anderen kann man sich aber auch mit mehreren Clients mit dem Server verbinden.

Wir haben also einen Webserver programmiert, welcher sich nach Möglichkeit mit der Drohne verbindet und von welchem man Befehle an diese senden kann. Mit diesem Server kann man sich dann mittels einem normalen Internetbrowser verbinden. Damit erhält man die Kontrolle über die Drohne. Es ist so auch möglich, sich mit mehreren Clients auf einem Server zu verbinden und so die Drohne parallel anzusteuern, jedoch kann es zu Konflikten beim Steuern kommen, da beide Clients gleich berechtigt sind.

## Entfernungsmessung mit Ultraschall

Die AR.Drone hat an ihrer Unterseite ein Ultraschallsensor, welcher die Entfernung durch Schallrückwurf zum nächsten Objekt recht genau angibt. Als erstes haben wir geguckt, in welchem Bereich die Entfernung gemessen wird und wie stark sich die Streuung des Schalls auf das Ergebnis auswirkt. Dabei kam ein erstaunlich genauer Wert heraus, solange man nicht versucht das System zu verstellen. Um die Entfernung nach vorne anstatt nach unten zu messen, haben wir erst einen Schallspiegel in einem 45° Winkel angebracht. Bei dem Versuch ist allerdings die Flugfähigkeit der Drohne zerstört worden, da diese alle vorhandenen Sensoren selbst verwendet um beispielweise die Höhe halten zukönnen. Die Werte waren ungenau oder häufig über lange Perioden komplett ungültig. Und ohne oder mit falschen Werten hat die Drohne versucht, diese Distanz als falsch interpretierte Höhe auszugleichen und ist entweder gegen die Decke oder gegen den Boden geflogen. Es stellten sich nun drei Lösungsmöglichkeiten für das Problem dar:

1. Eine Untersuchung des **Videostreams**.
   1. Mit Hilfe der Videoübertragung und geeigneten Analysealgorithmen erhält man Daten über die Entfernung und Positionierung von Objekten vor der Drohne. Hier liegt das Problem darin, dass eine komplette dreidimensionale Untersuchung mit nur einem Kamerabild deutlich zu lange braucht und die Entfernungen nach vorne nicht genau sind.
   2. Man könnte zwei Bilder des Videostreams, die sich durch die Ursprüngliche Position der Drohne unterscheiden verwenden. Somit könnte sich ein ziemlich realistisches dreidimensionales Modell erstellen lassen, jedoch brauch auch diese Bildauswertung zu lange. Außerdem variiert die Distanz der Position von denen die Bilder aufgenommen wurden zu stark, sodass der Algorithmus verschiedene Ergebnisse ausgeben würde.
2. Ein **Laser** für exakte punktgenaue Messungen. Um wirklich genaue Werte zu analysieren, kann ein Lasermessgerät genutzt werden. Es ist allerdings recht teuer und muss extern an die Drohne angebaut werden. Des Weiteren braucht es für eine Raumanalyse aber auch gar keine so genauen Werte und die Streuung ist sogar erwünscht.
3. Ein weiteres **Ultraschallmessgerät**. Ein seperates Messgerät ermöglicht in Kombination mit einem Raspberry-Pi eine völlig ausreichende Genauigkeit. Zusätzlich werden so auch Objekte kurz unter oder über der Flughöhe mit erkannt. Und es werden Flächen analysiert und nicht nur einzelne Punkte. Das hat den Vorteil, dass Fehlermessungen mit herausragenden Objekten vermieden werden können.

Ein weiterer separater Ultraschallsensor hat also durchaus Vorteile gegenüber der Lasermessung. Durch den Raspberry-Pi sind auch noch weitere Möglichkeiten offen.

## Bewegungsbefehle

Um die Drohne einfach nutzbar zu machen, muss sie eigenständig zu Zielpositionen fliegen können. Das Programm steuert dabei mit dem Kompass der Drohne, dem Neigungswinkel und der Verschiebung des Bildes der Bodenkamera die Drohne so, dass sie von der Startposition aus in eine bestimmte Richtung eine gewisse Entfernung weit fliegt.

Weitere Befehle sind zum Beispiel dafür da, die Drohne einmal um 360° zu drehen und die Entfernungen zu messen. Dabei ist es wichtig, dass die Drohne auf einer Position bleibt und nicht langsam weggleitet. Das ist duch die Verfolgung des Bildes der Bodenkammera gewährleistet. Das Bild wird dabei um die eigene Achse gedreht, passend mit der Drohne selbst.

Auch nach vorne kann die Position stabilisiert werden. Durch Gleiten nach rechts oder links, sowie nach oben oder unten verschiebt sich das Bild dem entsprechend. Gleitet die Drohne nach vorne oder nach hinten, wird das Bild kleiner oder größer und die verfolgten Punkte wandern näher zu einander oder von einander weg. Allerdings ist die Nutzung der Bodenkamera deutlich vorteilhafter.

Wir entschieden uns das Modul *ardrone-autonomy* zu verwenden, da dieses den Extended Kalman Filter implementiert, um die Position der Drohne zu stabilisieren. Sie verwendet auch nicht einen Videostream der Kameras sonder Navigationsdaten der Drohen, die bereits vorberechnet sind. Somit bleibt die wahl des Videostreams beim Anwender.

Außerdem bietet dieses Modul die Möglichkeit Strecken in Metern zurückzulegen.

## Bildanalyse

Jsfeat face, canny edge, scharr

Die Bildanalyse sollte auf der Client-Seite stattfinden um den die Drohne steuernden Server zu entlasten falls es viele Clients gibt. *JSFeat*  ist speziell für Computer Vision auf der Client-Seite unter Verwendung von JS/HTML5 ausgelegte Bibliothek. Wir haben in unser Projekt 3 Algorithmen implementiert:

1. Canny edge detector
2. scharr\_derivatives
3. Brightness Binary Feature Face objekt detector(bbf face)

(-!IMG)

# Das Darstellungsprogramm

Die von der Drohne gesammelten Daten sollen werden für den Anwender sichtbar gemacht durch Canvas-Oberfläche auf der Website zum grafischen Darstellen genutzt. Dort können dann einzelne Hindernisse zusätzlich zu bereits zurückgelegten Strecke eingezeichnet werden. Aus den gegebenen Messpunkten wird auch die Flugroute der Drohne ausgerechnet und an die Drohne weitergegeben.

(-!IMG)

# Probleme und Lösungen

# Ausblick

## Automatische Raumanalyse

Die Realisierung einer vollautomatischen Raumanalyse ist uns bisher nicht gelungen, jedoch dürfte diese gut realisierbar sein, wenn man ein Raspberry-Pi sicher auf der Drohne befestigen kann.

## Erster Rundumtest

Bei diesem Versuch dreht sich die Drohne einmal um die eigene Achse und nimmt Messpunkte auf. Dabei war zuerst die Position der Drohne als Mittelpunkt unbeweglich angenommen. Später wurde die Position aus dem Bewegungsabbild der Bodenkamera realistischer berechnet. Man konnte einen Überblick über die Analyse kriegen.

## Grundidee zu den Hindernissen

Misst die Drohne eine Entfernung, so kann der Messpunkt lokalisiert werden, in dem man die Position der Drohne im Raum mit der Richting und der Entfernung der Messung erweitert. Dieser Punkt wird dann in Zusammenhang mit den vorher und nachher gemessenen Punkten in Verbindung gebracht und bildet einen Teil eines Hindernisses. Liegen nacheinander gemessene Punkte zu weit auseinander, so soll der Teil zwischen den Punkten weiter analysiert werden. Ist ein Punkt ungültig, wird er nicht in Zusammenhang gebracht. Liegen Punkte nah beieinander, werden sie in einen Zusammenhang gebracht oder ganz zusammengefasst. Die Genauigkeit der Messung kann durch verschiedene Parameter beeinflusst werden:

1. Wählt man einen kleineren Drehwinkel, so wird die Abtastrate höher und die Genauigkeit wird erhöht. Die Drohne dreht sich langsamer.

Ist der maximal zulässige Abstand zwischen zwei Messpunkten kleiner, so wird eine höhere Genauigkeit erzwungen. Außerdem werden Störobjekte(Fäden, Stangen, Kabel) besser erkannt.

(-!IMG)

# Zusammenfassung

(-!EDIT)

# Vorschriften für die Nutzung von Quadrokoptern

Diese vier Bedingungen sind für unser Projekt relevant (Auszug):

1. Das deutsche Luftsicherheitsgesetz schließt eine Verwendung von Drohnen außerhalb des Sichtfeldes eines menschlichen Bedieners kategorisch aus.
2. Prinzipiell ist das fliegen von UAVs (Unmanned Areal Vehicle) versicherungspflichtig.
3. Innerhalb eines Abstandes von 1,5 km vom Flughafenzaun ist die Nutzung von Flugmodellen und unbemannten Flugsystemen grundsätzlich ganz verboten.
4. Über Menschenmengen, militärischen Objekten, Kraftwerken und Krankenhäusern darf grundsätzlich nicht geflogen werden.

# Literaturverzeichniss/Quellen

* ARDrone Parrot von Alternate(-!genau)
* Akku Amazon(-!genau)
* Extraflügel von Amazon(-!genau)
* github
* node.js
* module für node
* videobearbeitungsalgorithmen, jsfeat, canny,schaar
* jquery ninja
* ar-drone
* dronestream
* ardron-autonomy
* EKF
* Gesetzesgelümps
* <http://www.stuttgarter-zeitung.de/inhalt.unbemannte-fluggeraete-gesetze-bremsen-den-milliardenmarkt-fuer-drohnen.4ce7ef60-3279-4434-94b0-9078fa207dca.html>
* http://www.drohnen.de/vorschriften-genehigungen-fuer-die-nutzung-von-drohnen-und-multicoptern/