|  |
| --- |
| I believe I can fly V2.0 – Detektion und Klassifikation von NUI-Flugbefehlen anhand einzelner Farbkamera-Bilder  Studienarbeit  des Studiengangs Informatik Studienrichtung Informationstechnik/Angewandte Informatik  an der  Dualen Hochschule Baden-Württemberg Karlsruhe  von  Henri Kohlberg  20.05.2019  Matrikelnummer 6214814  Kurs TINF16B1  Ausbildungsfirma SAP SE, Walldorf  Betreuer Prof. Dr. Marcus Strand  Bearbeitungszeitraum 01.10.2018 – 20.05.2019 |

Erklärung

(gemäß §5(3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 29. 9. 2015)

Ich versichere hiermit, dass ich meine Projektarbeit mit dem Thema: „Java-Anwendungsentwicklung für SAP Sailing Analytics“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Ort, Datum Unterschrift

Vorwort

Fliegen war schon immer ein Traum der Menschheit…

Abstract

Englisches Vorwort

Inhaltsverzeichnis

[Abkürzungsverzeichnis VI](#_Toc536795706)

[Abbildungsverzeichnis VII](#_Toc536795707)

[Typographische Hinweise VIII](#_Toc536795708)

[1. Einleitung 1](#_Toc536795709)

[1.1. I believe I can fly – V1.0 1](#_Toc536795710)

[1.2. Problemstellung 1](#_Toc536795711)

[1.3. Technische Voraussetzungen 2](#_Toc536795712)

[1.3.1. Hardware 2](#_Toc536795713)

[1.3.2. Robot Operating System 3](#_Toc536795714)

[1.3.3. Simulation einer Drohne 4](#_Toc536795715)

[2. Software Komponenten 6](#_Toc536795716)

[2.1. Architektur 6](#_Toc536795717)

[2.2. Simulator 6](#_Toc536795718)

[3. Personenerkennung 7](#_Toc536795719)

[3.1. Problemstellung 7](#_Toc536795720)

[3.2. Computer Vision 7](#_Toc536795721)

[3.3. Neuronale Netze 7](#_Toc536795722)

[3.4. Implementierung 7](#_Toc536795723)

[4. Gestenerkennung 8](#_Toc536795724)

[4.1. Problemstellung 8](#_Toc536795725)

[4.2. Analyse von Körperbewegungen 8](#_Toc536795726)

[4.3. Umwandlung in Steuerungsbefehle 8](#_Toc536795727)

[4.4. Implementierung 8](#_Toc536795728)

[5. Benutzerausgabe 9](#_Toc536795729)

[5.1. Steuerung der Drohne 9](#_Toc536795730)

[5.2. Steuerung des Simulators 9](#_Toc536795731)

[6. Zusammenfassung und Ausblick 10](#_Toc536795732)

[Literaturverzeichnis VI](#_Toc536795733)

[Glossar VII](#_Toc536795734)

Abkürzungsverzeichnis

API Application Programming Interface, dt. Programmierschnittstelle

FPS Frames per Second, dt. Bilder pro Sekunde

GPU Graphics processing unit, dt. Grafikprozessor

SDK Software Development Kit

Abbildungsverzeichnis

References -> Insert Table Of Figures

Typographische Hinweise

* Eigenenamen und Abkürzungen sind bei ihrer erstmaligen Erscheinung *kursiv* dargestellt. Nähere Informationen sind dem Fließtext oder Glossar zu entnehmen.
* Abkürzungen sind dem Abkürzungsverzeichnis zu entnehmen, werden aber üblicherweise ebenfalls im Fließtext erläutert.
* Fachbegriffe, die im Fließtext nicht weiter erläutert werden, aber trotzdem für das Verständnis des Inhalts von Vorteil sind, werden im Glossar näher beschrieben.
* Quellcode und technische Namen sind durch rote Schrift mit grauem Hintergrund oder eine markiert.

umgebende Textbox

# Einleitung

To be done

## I believe I can fly – V1.0

Diese Studienarbeit ist Teil einer Reihe von Studienarbeiten, die sich mit dem Projekt "I believe I can fly" auseinandersetzen. Bei diesem Projekt geht es im Allgemeinen um die Steuerung einer Drohne mithilfe von Körperbewegungen des Piloten. Ziel ist es, dem Piloten ein möglichst realistisches Fluggefühl zu vermitteln und eine hohe Immersion zu erreichen. Dazu trägt der Pilot während den Flügen eine Videobrille, welche wiederrum einen Livestream einer an der Drohne befestigten Kamera anzeigt. Die Kamera ist dabei so ausgerichtet, dass sie aus Sicht der Drohne stets nach vorne zeigt und dem Piloten die Welt so zeigt, als würde er in einem fliegenden Fahrzeug sitzen.

Eine solche Technik kann für ganz verschiedene Einsatzzwecke verwendet werden. Dabei spielt es keine Rolle, ob diese Steuerung für die visuelle Inspektion eines schwer zu erreichenden Ortes wie z.B. einer Hochspannungsleitung oder zu Unterhaltungszwecken von Heimanwendern verwendet wird. Ein wichtiger Aspekt ist in allen Anwendungsfällen eine intuitive Steuerung und ein realistisches Gefühl für das Fliegen, sodass die Drohne präzise gesteuert werden kann.

Die erste Version des Projektes entstand im Rahmen einer Studienarbeit 2015 und setzt die Grundanforderungen bereits funktionierend um. Zum Einsatz kommt hierbei ein Sensor der Xbox Kinect Reihe, welcher mithilfe eines 3D-fähigen Infrarot-Sensors den Benutzer erkennen und die Körperbewegungen errechnen kann. Diese Körperbewegungen werden anschließend in Befehle für die Drohne umgerechnet und ermöglichen eine zuverlässige Steuerung. (Benz, et al., 2015)

Eine weitere Studienarbeit befasst sich mit einem intelligenten Assistenzsystem, welches den Piloten bei der Navigation durch Engstellen unterstützen soll. Aufgrund von technischen Schwierigkeiten kommt diese Arbeit jedoch zu keinem positiven Ergebnis, sodass ein solche Assistenzsystem noch in den frühen Kinderschuhen steckt. (Meise, et al., 2017)

## Problemstellung

Die erste Version von "I believe I can fly" funktioniert zwar recht gut und ermöglicht dem Benutzer ein echtes Gefühl des Fliegens, jedoch basiert diese Lösung auf Hardware, welche möglicherweise nicht jedem Benutzer zur Verfügung steht. Die Produktion des bereits erwähnten Kinect-Sensors wurde von Microsoft offiziell beendet und ist somit nicht mehr für Neukunden verfügbar. (Sarkar, 2018) Selbst wenn ein Nutzer Zugriff auf einen solchen Sensor hätte, müsste er diesen bei jeder Verwendung der Drohne benutzen. Auch die Verwendung eines 3D-Sensors eines anderen Herstellers kann dieses Problem nicht umgehen.

Eine mögliche Lösung liegt in der Verwendung einer handelsüblichen 2D-Webcam, welche heute in fast allen mobilen Endgeräten wie Laptops oder Smartphones verbaut ist. Diese Kameras sind günstig in der Herstellung und liefern schon seit einigen Jahren die benötigten Voraussetzung was die Qualität des aufgenommen Bildes angeht, siehe Kapitel 1.3.1 Hardware. Das Problem bei der Verwendung einer 2D-Informationsquelle liegt an der fehlenden dritten Dimension, welche für eine vollständige 3D-Steuerung eigentlich benötigt werden würde. Dieses Problem wiederum kann durch verschiedene Verfahren gelöst werden, welche im 4. Kapitel Gestenerkennung verglichen werden.

Neben dem bisher benötigten Sensor wird für die erste Version des Projektes zum Fliegen auch immer eine reale Drohne benötigt. Neben der Drohne muss aber auch eine für den Flugbetrieb geeignete Umgebung vorhanden sein. Im Freien spielen oftmals das Wetter sowie gesetzliche Bestimmungen für den Aufstieg einer Drohne eine große Rolle, in geschlossenen Räumen ist meistens der vorhandene Platz der limitierende Faktor. Vor Allem für Trainingszwecke ist dies aber eher hinderlich. Ziel dieser Studienarbeit ist also auch die Integration eines Simulators, welcher die Drohne vollständig ersetzt. Der Benutzer sieht dann an Stelle der Realität aus Sicht der Drohne ein gerendertes Bild des Simulators, welches möglichst realistisch aussehen soll um weiterhin das Gefühl des Fliegens vermitteln zu können. Mögliche Lösungen werden in Kapitel 5.2 Steuerung des Simulator behandelt.

## Technische Voraussetzungen

Von Beginn des Projektes an sind klare Anforderungen an sowohl die verwendete Software als auch die verwendete Hardware gegeben. Dieses Voraussetzungen ermöglichen eine flexible Laufzeitumgebung und heben Limitierungen wie ein bestimmtes Betriebssystem von Anfang an auf.

### Hardware

Die Aufnahme der Bilder, Erkennung der Gesten als auch die Simulation bzw. Steuerung der Drohne sollen nach Möglichkeit auf einem einzigen Gerät funktionieren. Um eine gewisse Mobilität zu erreichen, bietet sich für diese Zwecke ein leistungsstarker Laptop an, welcher eine handelsübliche Webcam eingebaut hat.

Die Webcam dient der Aufnahme einzelner Bilder des steuernden Benutzers und sollte daher von guter Qualität sein. Die Qualität wird unter Anderem durch die Bilder pro Sekunde (*FPS*) als auch die zur Verfügung stehende Auflösung der Bilder bestimmt. Ist einer dieser beiden Parameter zu gering, leidet die Qualität Steuerung darunter, denn zu wenige FPS führen zu einer sehr trägen Steuerung und eine zu geringe Auflösung könnte unter Umständen zu einer fehlschlagenden Gestenerkennung führen. Beide Folgen beinträchtigen unmittelbar die Immersion des Benutzers und sollten daher vermieden werden. (Abrash, 2014)

Sowohl die Erkennung der Gesten als auch die Simulation der Drohne benötigen eine hohe verfügbare Rechenleistung. Da die Berechnungen gleichzeitig ausgeführt und die entsprechenden Algorithmen parallelisiert werden können, ist vor Allem eine leistungsstarke Grafikkarte (*GPU*) mit einer großen Anzahl an Rechenkernen notwendig. Die unterschiedlichen Hersteller für Grafikkarten haben unterschiedliche Architekturen mit verschiedenen Eigenschaften, welche sich gut und manchmal weniger gut für solche Berechnungen nutzen lassen. Dies sollte bei der Auswahl des Rechners unbedingt beachtet werden. Daher empfiehlt es sich, auf eine moderne Plattform zu setzten, welche die neusten Features unterstützt, da sich in den letzten Jahren einiges in diesem Bereich entwickelt hat. (NVIDIA Corporation, 2019) Der zur Verfügung stehende Laptop hat einen Prozessor mit vier Kernen der Firma Intel und eine NVIDIA Grafikkarte der Reihe GTX 1060 verbaut, welche gut für solche Einsatzzwecke geeignet ist und ausreichend Leistung liefern sollte.

Um eine Drohne über einen Rechner ansteuern zu können, muss die Drohne diese Art der Kommunikation logischerweise ebenfalls unterstützen. Auf dem Markt sind einige dieser Drohnen zu finden, jedoch ist die eigene Software-Entwicklung bei einem speziellen Modell sehr gut möglich, nämlich der *AR-Drone 2.0* des Herstellers Parrot. Der Hersteller stellt Entwicklern ein Software Development Kit (*SDK*) zur Verfügung, welches die Steuerung der Drohne für Programme über ein Application Programming Interface (*API*) ermöglicht. (Parrot Drones SAS, 2016) Dieser Quadrocopter wurde ebenfalls in den beiden vorangegangen Studienarbeiten verwendet und wird deshalb auch im Rahmen dieser Studienarbeit verwendet werden.

### Robot Operating System

Software sollte nach Möglichkeit immer modular aufgebaut werden und optimalerweise plattformunabhängig funktionieren. Dies hat den Vorteil, dass Quellcode nur einmal geschrieben werden muss und anschließend wiederverwendet werden kann, ohne eine Anpassung für die entsprechende Plattform vornehmen zu müssen. In der Robotik gibt es für diesen Zweck das *Robot Operating System* (*ROS*), welches typischerweise auf Linux aufsetzt und Entwicklern eine Sammlung verschiedener Libraries zur Verfügung stellt. Die notwendige Infrastruktur für modulare Software wird ebenfalls bereitgestellt und ermöglicht durch die Kompatibilität zu Programmiersprachen wie C, C++ und Python ein breites Spektrum an Anwendungen. (Open Robotics Foundation, 2019)

ROS wird als Open Source Projekt gepflegt und kann somit theoretisch von jeder Person erweitert werden. Verschiedene Linux-Distributionen werden hierbei offiziell als Betriebssystem unterstützt, jedoch gibt es auch die Möglichkeit, ROS auf Windows und MacOS zu betreiben. Diese Varianten werden offiziell zwar nicht unterstützt oder getestet, werden jedoch durch die Community vieler Entwickler unterstützt.

Die Modularität wird in ROS durch Pakete umgesetzt. Ein Paket hat im Idealfall eine übergeordnete Aufgabe und besteht intern aus weiteren Nodes. Ein Node übernimmt genau eine Aufgabe und kommuniziert mit anderen Nodes über Topics, um z.B. komplexere Algorithmen umsetzten zu können und die inhaltliche Zuordnung möglichst gering zu halten. Topics sind spezielle Kommunikationssysteme, welche Nachrichten von Nodes entgegennehmen und an alle anderen Nodes weiterreichen. Dementsprechend können alle Nodes auf bestimmte Nachrichten in einem Topic warten und beim Empfangen der Nachricht ihr Verhalten anpassen. Dieses System ist in der Informatik weitverbreitet und wird in der ereignisgesteuerten Programmierung auch als Bussystem bezeichnet. (Bruns, et al., 2010)

Dieses Konzept soll auch in der Studienarbeit verwendet werden. So kann ein Node die Erkennung der Person übernehmen, ein anderer Node wiederrum übernimmt die Erkennung der Gesten usw. Somit ist die Software modular und so können, unter der Voraussetzung, dass alle Schnittstellenspezifikationen eingehalten werden, einzelne Module problemlos ausgetauscht werden. Siehe allgemeines Modulkonzept

### Simulation einer Drohne

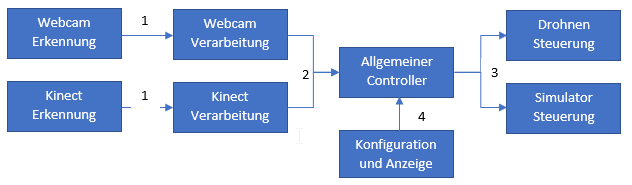
Unter gewissen Umständen ist es nicht immer möglich, eine reale Drohne fliegen lassen zu können. Deshalb kann es sinnvoll sein, die Drohne lediglich in einem Simulator fliegen zu lassen. Das Bild der echten Kamera, welches der Benutzer normalerweise in seiner Videobrille sehen würde, wird bei diesem Setup durch ein virtuelles Bild ersetzt und erlaubt dem Nutzer eine realistische Immersion. Wichtig an dieser Stelle ist die möglichst realistische Grafik der Simulation, sodass nicht das Gefühl einer simulierten Umgebung entsteht. Diese Möglichkeit ist vor Allem während der Entwicklung oder als Training für neue Nutzer sinnvoll und senkt dabei gleichzeitig das Risiko, dass durch unkontrolliertes Fliegen oder einen Fehler ein tatsächlicher Schaden entsteht. Für die Simulation stehen softwaretechnisch mehrere Möglichkeiten zur Auswahl, welche im Kapitel XXX genauer analysiert und verglichen werden.

# Software Komponenten

* Umsetzung in Python

## Architektur

* ROS Modul Übersicht
* Kommunikationsformate (JSON?)



## Simulator

* Welcher Simulator? Warum?
* Verfügbarkeit auf Plattformen
* Performance Probleme

# Personenerkennung

## Problemstellung

## Computer Vision

* Allgemeine Verfahren
* Gescheiterte Versuche (Background Subtraction)
* Verfügbare Libraries
* Performance (zu langsam, da nur CPU)

## Neuronale Netze

* Performance (gut, aber hohe GPU Auslastung)
* Verfügbare neuronale Netze
* Jährliche Challenges (COCO, …)

## Implementierung

* Wahl von OpenPose Netz

# Gestenerkennung

## Problemstellung

* Fehlende Z-Koordinate

## Analyse von Körperbewegungen

* Tracking vs. Erkennung
* Verschiebung der Koordinaten

## Umwandlung in Steuerungsbefehle

* Ablesen von bestimmten Eigenschaften (Schulterwinkel, Armlänge, …)
* Normalisieren der Warte
* "Pseudo"-Koordinate durch Längenunterschiede bei gleichem Winkel

## Implementierung

* Kalibrierung
* Fuzzy-Controller mit verschiedenen Parametern

# Benutzerausgabe

* Angle vs. Acro Mode

## Steuerung der Drohne

* Verwendete API / Protokolle
* Umwandlung der allgemeinen Befehle
* ROS Integration

## Steuerung des Simulators

* Umwandlung der allgemeinen Befehle
* ROS Integration

# Zusammenfassung und Ausblick

* Wie weit ist Umsetzung?
* Welche Probleme gibt es weiterhin?
* Mögliche Fortführungen

Literaturverzeichnis

**Abrash, Michael. 2014.** What VR could, should, and almost certainly will be within two years. [Online] 15. 01 2014. [Zitat vom: 01. 02 2019.] http://media.steampowered.com/apps/abrashblog/Abrash%20Dev%20Days%202014.pdf.

**Benz, Nicolai, Bergen, Marcus von und Vonscheidt, Denis. 2015.** *Gestensteuerung eines Flugroboters im AR-Kontext - I believe I can fly.* Karlsruhe : s.n., 2015.

**Bruns, Ralf und Dunkel, Jürgen. 2010.** *Event-Driven Architecture. Softwarearchitektur für ereignisgesteuerte Geschäftsprozesse.* Berlin : Springer, 2010. ISBN 978-3-642-02438-2.

**Meise, Christoph und Lenk, Max. 2017.** *Entwicklung eines Indoor-Assistenzsystems für Multicopter mit Hilfe von Monokularer Tiefenbildrekonstruktion.* Karlsruhe : s.n., 2017.

**NVIDIA Corporation. 2019.** Was ist CUDA? [Online] 2019. [Zitat vom: 01. 02 2019.] https://www.nvidia.de/object/cuda-parallel-computing-de.html.

**Open Robotics Foundation. 2019.** ROS. [Online] 2019. [Zitat vom: 01. 02 2019.] http://www.ros.org.

**Parrot Drones SAS. 2016.** Parrot for Developers. [Online] 2016. [Zitat vom: 01. 02 2019.] https://developer.parrot.com/products.html.

**Sarkar, Samit. 2018.** Microsoft discontinues Xbox One Kinect adapter. *Polygon.* [Online] Vox Media, 03. 01 2018. [Zitat vom: 01. 02 2019.] https://www.polygon.com/2018/1/2/16842072/xbox-one-kinect-adapter-out-of-stock-production-ended.

Glossar

**Library** to be done