

## Bachelor-Thesis

# Human-Machine Interface for Operating a Blimb

Spring Term 2012

---

**Supervised by:**

Konrad Rudin  
Mora Javier Alonso  
X Paul Beardsley XXXXX

**Authors:**

Krebs Matthias  
Ledergerber Anton



# Declaration of Originality

I hereby declare that the written work I have submitted entitled

## **Human-Machine Interface for Operating a Blimb**

is original work which I alone have authored and which is written in my own words.<sup>1</sup>

### **Author(s)**

Anton  
Matthias

Ledergerber  
Krebs

### **Supervising lecturer**

Konrad  
Javier Alonso  
Paul XXX

Rudin  
Mora  
Beardsley XXX

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' ([http://www.ethz.ch/students/exams/plagiarism\\_s\\_en.pdf](http://www.ethz.ch/students/exams/plagiarism_s_en.pdf)). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

---

Place and date

---

Signature

---

<sup>1</sup>Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

# Inhaltsverzeichnis

<b>Abstract</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>v</b>
<b>Symbols</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Context . . . . .	1
1.2 Goals . . . . .	1
1.3 System Overview . . . . .	1
1.4 Similar Systems and their HMI . . . . .	1
1.5 Structure of the Report . . . . .	1
<b>2 Einige wichtige Hinweise zum Arbeiten mit L<sup>A</sup>T<sub>E</sub>X</b>	<b>3</b>
2.1 Gliederungen . . . . .	3
2.2 Referenzen und Verweise . . . . .	3
2.3 Aufzählungen . . . . .	3
2.4 Erstellen einer Tabelle . . . . .	4
2.5 Einbinden einer EPS-Graphik . . . . .	5
2.6 Mathematische Formeln . . . . .	5
2.7 Weitere nützliche Befehle . . . . .	6
<b>3 Finding a Hardware and Software Solution</b>	<b>7</b>
3.1 Requirements . . . . .	7
3.2 Existing Solutions . . . . .	7
3.2.1 Hardware . . . . .	7
3.2.2 Software . . . . .	7
3.3 Realization . . . . .	7
3.3.1 Compact and Convenient Solution . . . . .	7
3.3.2 QGroundControl . . . . .	7
3.3.3 Mavlink . . . . .	7
<b>4 The different Control Modes</b>	<b>9</b>
4.1 Elaboration . . . . .	9
4.2 Manual Control Modes . . . . .	9
4.3 Automatic Control Modes . . . . .	9
<b>5 Trajectory Planning</b>	<b>11</b>
5.1 Experimental Design . . . . .	11
5.2 Definition of Trajectories . . . . .	11
5.2.1 Paths and Trajectories . . . . .	11
5.2.2 Interpolation and Approximation . . . . .	11
5.3 Spline Theory . . . . .	12

5.3.1	Piecewise Polynomial Interpolating Splines . . . . .	12
5.3.2	B-Splines . . . . .	12
5.4	Trajectory Generation . . . . .	12
5.4.1	System Constraints . . . . .	12
5.4.2	Time Parametrization . . . . .	13
5.5	Controller Implementation . . . . .	13
5.5.1	Trajectory Following . . . . .	13
5.5.2	Pure Pursuit Controller . . . . .	13
5.5.3	Cross Track Error Controller . . . . .	13
5.6	Discussion . . . . .	13
<b>6</b>	<b>Conclusion</b>	<b>15</b>
	<b>Bibliography</b>	<b>16</b>



# Abstract

Hier kommt der Abstact hin ...





# Acknowledgements

Without the help of a few people this thesis would not have been possible. We received the necessary support from all sides throughout the project to realize the this HMI which we are proud of.

Prof. Dr. Roland Y. Siegwart

Dr. Paul Beardsley

PhD students Konrad Rudin and Javier Alonso Mora

Gerhard Röthlin

Lorenz Meier

Alexander Rudyk



# Symbols

## Symbols

$\phi, \theta, \psi$	roll, pitch and yaw angle
$b$	gyroscope bias
$\Omega_m$	3-axis gyroscope measurement

## Indices

$x$	x axis
$y$	y axis

## Acronyms and Abbreviations

ETH	Eidgenössische Technische Hochschule
EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter



# Kapitel 1

## Introduction

1.1 Context

1.2 Goals

1.3 System Overview

1.4 Similar Systems and their HMI

1.5 Structure of the Report



## Kapitel 2

# Einige wichtige Hinweise zum Arbeiten mit L<sup>A</sup>T<sub>E</sub>X

Nachfolgend wird die Codierung einiger oft verwendeten Elemente kurz beschrieben. Das Einbinden von Bildern ist in L<sup>A</sup>T<sub>E</sub>X nicht ganz unproblematisch und hängt auch stark vom verwendeten Compiler ab. Typisches Format für Bilder in L<sup>A</sup>T<sub>E</sub>X ist EPS<sup>1</sup>.

### 2.1 Gliederungen

Ein Text kann mit den Befehlen `\chapter{.}`, `\section{.}`, `\subsection{.}` und `\subsubsection{.}` gegliedert werden.

### 2.2 Referenzen und Verweise

Literaturreferenzen werden mit dem Befehl `\cite{.}` erzeugt. Ein Beispiel: [3]. Zur Erzeugung von Fussnoten wird der Befehl `\footnote{.}` verwendet. Auch hier ein Beispiel<sup>2</sup>.

Querverweise im Text werden mit `\label{.}` verankert und mit `\ref{.}` erzeugt. Beispiel einer Referenz auf das zweite Kapitel: Kapitel 2.

### 2.3 Aufzählungen

Folgendes Beispiel einer Aufzählung ohne Numerierung,

- Punkt 1
- Punkt 2

wurde erzeugt mit:

```
\begin{itemize}
  \item Punkt 1
  \item Punkt 2
\end{itemize}
```

Folgendes Beispiel einer Aufzählung mit Numerierung,

1. Punkt 1

---

<sup>1</sup>Encapsulated Postscript

<sup>2</sup>Bla bla.

## 2. Punkt 2

wurde erzeugt mit:

```
\begin{enumerate}
  \item Punkt 1
  \item Punkt 2
\end{enumerate}
```

Folgendes Beispiel einer Auflistung,

**P1** Punkt 1

**P2** Punkt 2

wurde erzeugt mit:

```
\begin{description}
  \item[P1] Punkt 1
  \item[P2] Punkt 2
\end{description}
```

## 2.4 Erstellen einer Tabelle

Ein Beispiel einer Tabelle:

Tabelle 2.1: Daten der Fahrzyklen ECE, EUDC, NEFZ.

Kennzahl	Einheit	ECE	EUDC	NEFZ
Dauer	s	780	400	1180
Distanz	km	4.052	6.955	11.007
Durchschnittsgeschwindigkeit	km/h	18.7	62.6	33.6
Leerlaufanteil	%	36	10	27

Die Tabelle wurde erzeugt mit:

```
\begin{table}[h]
\begin{center}
\caption{Daten der Fahrzyklen ECE, EUDC, NEFZ.}\vspace{1ex}
\label{tab:tabnefz}
\begin{tabular}{l|ccc}
\hline
Kennzahl & Einheit & ECE & EUDC & NEFZ \\ \hline
Dauer & s & 780 & 400 & 1180 \\
Distanz & km & 4.052 & 6.955 & 11.007 \\
Durchschnittsgeschwindigkeit & km/h & 18.7 & 62.6 & 33.6 \\
Leerlaufanteil & \% & 36 & 10 & 27 \\
\hline
\end{tabular}
\end{center}
\end{table}
```



## 2.5 Einbinden einer EPS-Graphik

Das Einbinden von Graphiken kann wie folgt bewerkstelligt werden:

```
\begin{figure}[h]
  \centering
  \includegraphics[width=0.75\textwidth]{pics/k_surf.eps}
  \caption{Ein Bild.}
  \label{pics:k_surf}
\end{figure}
```

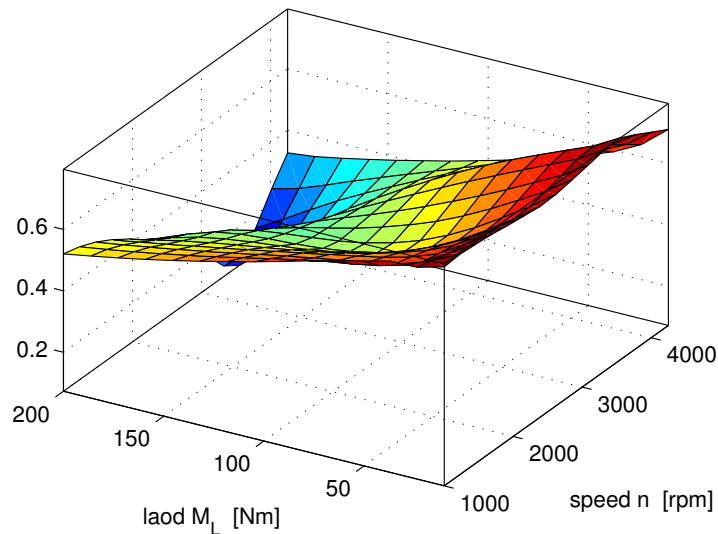


Abbildung 2.1: Ein Bild.

oder bei zwei Bildern nebeneinander mit:

```
\begin{figure}[h]
  \begin{minipage}[t]{0.48\textwidth}
    \includegraphics[width = \textwidth]{pics/cycle_we.eps}
  \end{minipage}
  \hfill
  \begin{minipage}[t]{0.48\textwidth}
    \includegraphics[width = \textwidth]{pics/cycle_ml.eps}
  \end{minipage}
  \caption{Zwei Bilder nebeneinander.}
  \label{pics:cycle}
\end{figure}
```

Bemerkung: Ersetzt man den Positionierungsparameter `h` durch `H`, so wird das Gleiten der Abbildung verhindert.

## 2.6 Mathematische Formeln

Einfache mathematische Formeln werden mit der `equation`-Umgebung erzeugt:

$$p_{me0f}(T_e, \omega_e) = k_1(T_e) \cdot (k_2 + k_3 S^2 \omega_e^2) \cdot \Pi_{max} \cdot \sqrt{\frac{k_4}{B}}. \quad (2.1)$$

Der Code dazu lautet:

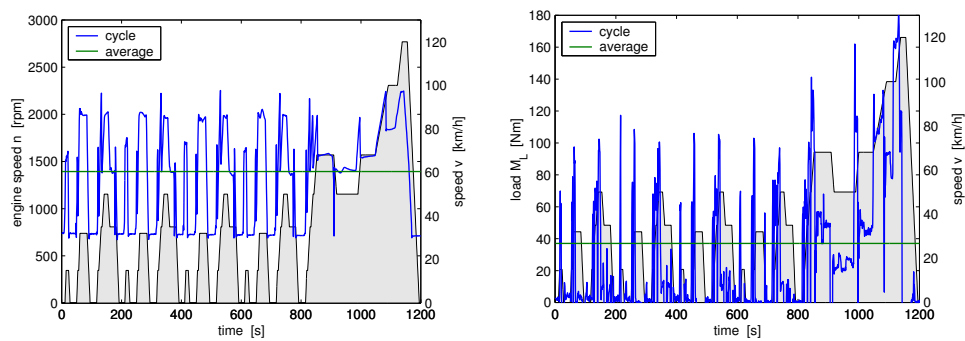


Abbildung 2.2: Zwei Bilder nebeneinander.

```
\begin{equation}
p_{me0f}(T_e,\omega_e) \ = \ k_1(T_e) \cdot (k_2+k_3 \ S^2
\omega_e^2) \cdot \Pi_{max} \cdot \sqrt{\frac{k_4}{B}} \ , \ .
\end{equation}
```

Mathematische Ausdrücke im Text werden mit `$formel$` erzeugt (zB:  $a^2 + b^2 = c^2$ ).

## 2.7 Weitere nützliche Befehle

Hervorhebungen im Text sehen so aus: *hervorgehoben*. Erzeugt werden sie mit dem `\epmh{.}` Befehl.

## Kapitel 3

# Finding a Hardware and Software Solution

References to [6]

### 3.1 Requirements

*Remote Control, Intuitive Control for 6DoF, Livestream, Waypoints*

### 3.2 Existing Solutions

#### 3.2.1 Hardware

*RC, Joystick, QGoSphere, 3dMouse, Wii Controller, Smartphones, Tablets, TabletPC*

#### 3.2.2 Software

*QGroundControl, OpenPilot, Qt-Libraries*

### 3.3 Realization

#### 3.3.1 Compact and Convenient Solution

*About advantages of TabletPC, 3dMouse, RC*

#### 3.3.2 QGroundControl

*Adaptions in QGroundControl, 3dMouse, Touchscreen, Splines and Trajectory Controller*

*Only how it looks like and how to use. 3dMouse and Touchscreen are not described further, splines, trajectories and trajectory controller are described in chapter 5*

#### 3.3.3 Mavlink

*Summary of Protocol, adaptions and use for SKYE*



## Kapitel 4

# The different Control Modes

### 4.1 Elaboration

*About the need of different modes, the requirements of image capturing and overview of the realized modes*

### 4.2 Manual Control Modes

*Direct Control and Assisted Control*

### 4.3 Automatic Control Modes

*Half Automatic Control and Full Automatic Control*



# Kapitel 5

## Trajectory Planning

For the two most advanced modes, i. e. the Half-Automatic and the Full-Automatic Mode, trajectories had to be generated. In this chapter the best trajectories for SKYE are elaborated and tested with suitable trajectory controllers. Performance results based on a MATLAB simulation are shown.

### 5.1 Experimental Design

The main application fields of the system SKYE are image capturing and agile performance demonstrations. The waypoints used to test the trajectory algorithms had therefore to be alike these situations. All the results below belong to the three sample waypoints shown in figure ?? . Indeed, to verify the conclusions, some more situations were considered.

Furthermore, to score (**and optimize?**) the generated trajectories  $\tilde{p}(t)$  and the resulting trace of the system  $r(t)$ , we set up the following criteria. Criteria 1 to 3 are referred to as *static criteria* as they do not depend any simulation. The remaining ones (*dynamic criteria*) then mainly depend on the used controller.

1. DEVIATION BETWEEN CHORD CONNECTION AND PATH
2. CURVATURES OF PATH
3. ACCELERATION OF TRAJECT
4. DEVIATION BET TRAJ AND TRACE
5. ACCELERATION OF TRACE
6. TIME SYNCHRONITY

### 5.2 Definition of Trajectories

#### 5.2.1 Paths and Trajectories

#### 5.2.2 Interpolation and Approximation

If one wants to draw a line through a set of data points, there exists two ways to do this. On the one hand the line must pass all data points no matter how many bends it will have, on the other hand the line tries to best fit the data, i.e. a function of a certain order is adopted to best fit the data. E.g. this can be done with least-squares. If the pilot defines the trajectory with a set of waypoints, i.e. data points, he usually wants the (UAV already defined!!!!) to pass through all of

them. Therefore the waypoints must be interpolated and not approximated with a suitable curve.

**(!!!Bsp Plot von interpolierenden und approximierenden funktionen!!!)**

## 5.3 Spline Theory

A set of data points can be interpolated with one single curve or with a set of curves defined over a certain interval. For a references to [1], [2] and [?]

**Continuity**

**Boundary Conditions**

**Polynomial Order**

**Parametrization**

### 5.3.1 Piecewise Polynomial Interpolating Splines

**Boundary Conditions**

**Polynomial Order**

**Parameterization**

### 5.3.2 B-Splines

**Boundary Conditions**

**Polynomial Order**

**Parametrization**

## 5.4 Trajectory Generation

### 5.4.1 System Constraints

**Maximum Velocities and Accelerations**

In order to plan a feasible trajectory one has to know the capabilities of the system. Here just a basic derivation for the velocities and accelerations is given, for more details refer to (!!!!Bsc Thesis Joe, Bsc Thesis Andy)

The maximum feasible acceleration in any direction is calculated to be:

$$|a_{max}| = \frac{|F_{res,w}|}{m_{tot}} = 0.96m/s^2 \quad (5.1)$$

Whereas the  $F_{res,w}$  is the force resulting from all four thrusters operated under full load in the worst direction and  $m_{tot}$  is the sum of the masses of the helium, the virtual mass and the mass of the system itself.

The maximum feasible velocity in any direction is calculated to be:

$$|v_{max}| = \sqrt{\frac{|F_{res,w}|}{\frac{1}{2}c_d\rho\pi r^2}} = 2.9m/s \quad (5.2)$$

which is nothing but  $|F_{res,min}| = |F_{drag}|$ .



For trajectories for position and orientation the maximal feasible angular acceleration is also important. It is calculated to be:

$$|\Psi_{max}| = \frac{|M_{res,w}|}{|\lambda_{max,J_B}|} = 2.06 rad/s^2 \quad (5.3)$$

which is quite conservative because it is assumed that worst axis for turning is also the principle axis of the inertia tensor with the highest inertia.

Since the system is almost undamped for rotations, the rotational velocities will never be the limiting factor.

### 5.4.2 Time Parametrization

## 5.5 Controller Implementation

Some commonly used trajectory controllers<sup>1</sup> are tested to follow the defined trajectories. The *Trajectory following* controller supplies the system's position controller [9] with a feed forward reference signal. Although it delivers good results for ideal case, the tracking get worse for the non perfect model case. The *pure pursuit* controller, which is based on a lookahead point as well as the *cross track error* controller dynamically react on model uncertainties and yield therefore to more robust path tracking results.

BLA BLA introduce notation..  $r(t)$  bla.

XXXX see [3] and [4]

### 5.5.1 Trajectory Following

Assuming a perfect model and a trajectory considering all system constraints<sup>2</sup>, the position  $r(t)$  of the system can be assumed to be equal to the trajectory  $\tilde{p}(t)$  at any time. Therefore, a straight forward way of a trajectory controller is to follow the trajectory  $\tilde{p}(t)$  for every time  $t$ . This yields accurate tracking in a safe environment [7].

$$[r_{ref}(t), \dot{r}_{ref}(t), \ddot{r}_{ref}(t)]^T = [\tilde{p}(t), \dot{\tilde{p}}(t), \ddot{\tilde{p}}(t)]^T \quad (5.4)$$

Testing the controller yields good performance.. BLA BLA Graphic figure 5.1

### 5.5.2 Pure Pursuit Controller

Another commonly used trajectory controller is Pure Pursuit [3]. To consider all dynamics of the trajectory, the reference input is based on a lookahead point  $\tilde{p}(t_{cl} + \Delta T) = \tilde{p}(t_{cl}) + \dot{\tilde{p}}(t_{cl})\Delta T + \ddot{\tilde{p}}(t_{cl}) + ORDNING$ .

### 5.5.3 Cross Track Error Controller

see [5]

## 5.6 Discussion

<sup>1</sup>[3] provides a good overview to trajectory control.

<sup>2</sup>I.e. saturations of  $\dot{r}(t)$  and its derivatives.

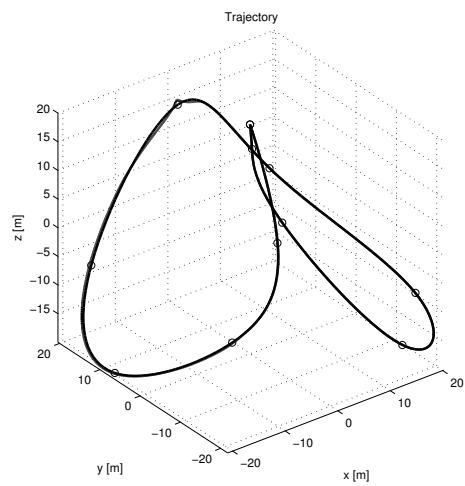


Abbildung 5.1: Trajectory following yields to extremly awesome tracking.

**Kapitel 6**

**Conclusion**

# Literaturverzeichnis

- [1] G. ENGELN-MÜLLGES, K. NIEDERDRENK, R. WODICKA: *Numerik- Algorithmen : Verfahren, Beispiele, Anwendungen*. Springer Verlag, 2011.
- [2] L. BIAGIOTTI, C. MELCHIORRI: *Trajectory Planning for Automatic Machines and Robots*. Springer Verlag, 2008.
- [3] J. M. SNIDER: *Automatic Steering Methods for Autonomous Automobile Path Tracking*. Research Report CMU-RI-TR-09-08, Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania, 2009.
- [4] A. DE LUCA, G. ORIOLO, C. SAMSON: *Feedback Control of a Nonholonomic Car-Like Robot*. In *Robot Motion Planning and Control*, pages 171-249, 1998.
- [5] D. L. WILLIAMS: *Loitering Behaviors of Autonomous Underwater Vehicles*. MSc thesis, Naval Postgraduate School, Monterey, California, 2002.
- [6] T. KAMMERMANN: *Evaluation and implementation of a control device for a ballbot*. BSc thesis, ETH Zurich, 2010.
- [7] S. DÖSEGGGER: *Time-optimal trajectories for a Ballbot*. BSc thesis, ETH Zurich, 2010.
- [8] J. WEICHART: *Agile Blimp Modeling and Simulation Environment*. BSc thesis, ETH Zurich, 2012.
- [9] D. MEIER, L. MÜRI: *Agile Blimp Controller Design*. BSc thesis, ETH Zurich, 2012.
- [10] WON Y. YANG, [ET AL.]: *Applied Numerical Methods Using MATLAB*. Wiley-Interscience, Hoboken, 2005.
- [11] J. BLANCHETTE, M SUMMERFIELD: *C++ GUI Programming with Qt 4*. Prentice Hall, Upper Saddle River, N.J., 2010.
- [12] E. T. Y. LEE: *Choosing nodes in parametric curve interpolation, Computer-Aided Design*. pages 363-370, (<http://www.sciencedirect.com/science/article/pii/0010448589900031>), 1989