Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autonomous Systems Lab
Prof. Roland Siegwart

Bachelor-Thesis

Human-Machine Interface for Operating a Blimp

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.[1]

Author(s)

Anton	Ledergerber
Matthias	Krebs

Supervising lecturer

Konrad	Rudin
Javier Alonso	Mora
Paul XXX	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`). 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

[1]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

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

ϕ, θ, ψ	roll, pitch and yaw angle
b	gyroscope bias
Ω_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 LaTeX

Nachfolgend wird die Codierung einiger oft verwendeten Elemente kurz beschrieben. Das Einbinden von Bildern ist in LaTeX nicht ganz unproblematisch und hängt auch stark vom verwendeten Compiler ab. Typisches Format für Bilder in LaTeX ist EPS[1].

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[2].
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

[1] Encapsulated Postscript
[2] 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}{ll|ccc}
 \hline
 Kennzahl & Einheit & ECE & EUDC & NEFZ \\ \hline \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}
```

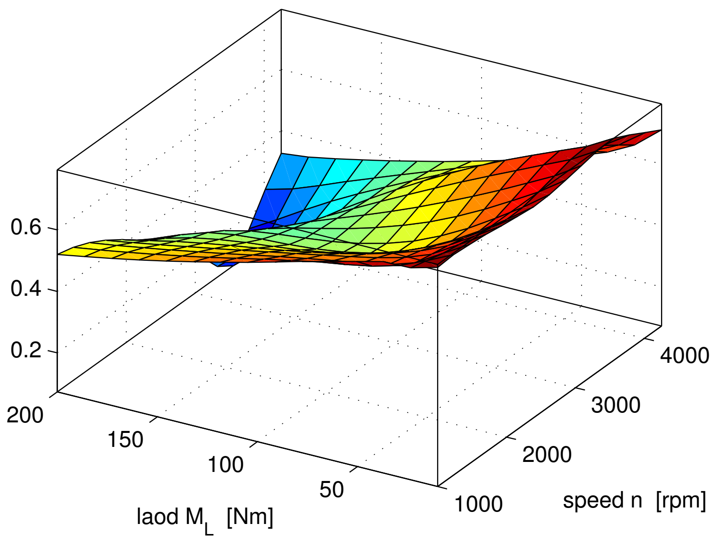


Abbildung 2.1: Ein Bild.

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}
```

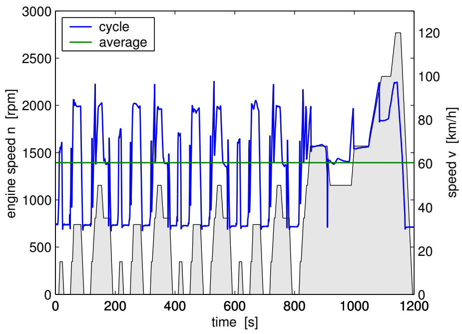
Bemerkung: Ersetzt man den Positionierungsparameter `h` durch `H`, so wird das Gleiten der Abbildung verhindert.

2.6 Mathematische Formeln

Einfache mathematische Formeln werden mit der equation-Umgebung erzeugt:

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

Der Code dazu lautet:



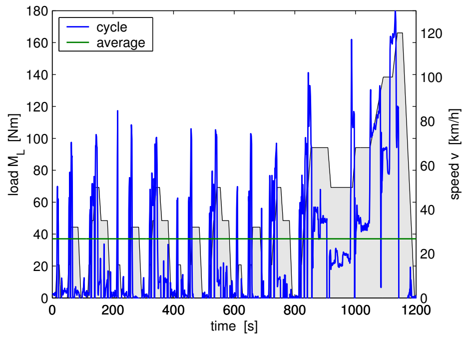


Abbildung 2.2: Zwei Bilder nebeneinander.

```
\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 Protocal, 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 5.1. Indeed, to verify the conclusions, some more situations had to be considered.

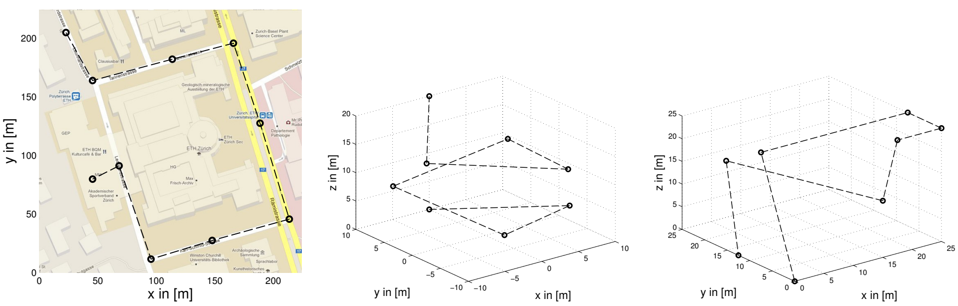
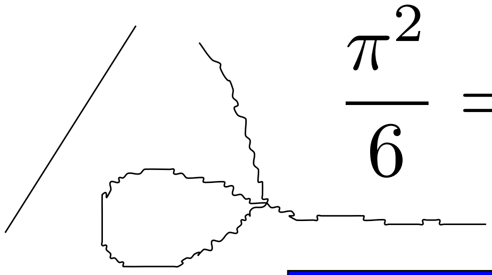


Abbildung 5.1: The experimental environment based on three samples. **Left:** The *road* waypoints represent the need of low overshoots to not touch obstacles beside the streets. Its long straight ways enable high velocities. **Center:** The *helix* waypoints represents the circumnavigation of any obstacle. It yields to high curvatures of the track. **Right:** The *agile* waypoints include both straight sections as high curvatures.

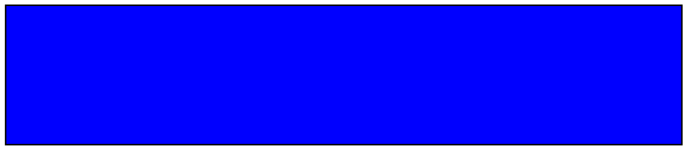
Furthermore, to score (**and optimize?**) the generated trajectories $\tilde{p}(t)$ and the resulting trace of the system $r(t)$, we set up the following criteria. Criteria (i) to (iii) are refered to as *static criteria* as they do not depend any simulation. The remaining ones (*dynamic criteria*) then mainly depend on the used controller[1].
Being L_p the length of the path $\mathbf{p}(u)$ and T_p the time of the trajectory $\tilde{\mathbf{p}}(t)$

$$L_p = \int_{u_{min}}^{u_{max}} du \qquad T_p = \int_{t_{min}}^{t_{max}} dt \tag{5.1}$$

[1]For detailed description of the used notation consider section 5.2



$$\frac{\pi^2}{6} = \lim_{n\to\infty} \sum_{k=1}^{n} \frac{1}{k^2}$$



the criteria are:

i) Average deviation between actual path and chord connection between waypoints

$$J_1 = \int_{u_{min}}^{u_{max}} \|\mathbf{p}(u) - \mathbf{p}_2(u)\| du \cdot L_p^{-1} \tag{5.2}$$

ii) Average curvature of path[2]

$$J_2 = \int_{u_{min}}^{u_{max}} \frac{\|\dot{\mathbf{p}}(u) \times \ddot{\mathbf{p}}(u)\|}{\|\dot{\mathbf{p}}(u)\|^3} du \cdot L_p^{-1} \tag{5.3}$$

iii) Average acceleration of the trajectory

$$J_3 = \int_{t_{min}}^{t_{max}} \|\ddot{\tilde{\mathbf{p}}}(t)\| dt \cdot T_p^{-1} \tag{5.4}$$

iv) Deviation between trajectory and trace of the system[3]

$$J_4 = \int_{t_{min}}^{t_{max}} \|\tilde{\mathbf{p}}(t_{cl}) - \mathbf{r}(t)\| dt \cdot T_p^{-1} \tag{5.5}$$

v) Average acceleration of the system

$$J_5 = \int_{t_{min}}^{t_{max}} \|\ddot{\mathbf{r}}(u)\| dt \cdot T_p^{-1} \tag{5.6}$$

vi) Time synchronity[4]

$$J_6 = \|\tilde{\mathbf{p}}(t_{max}) - \mathbf{r}(t_{max})\| \cdot L_p^{-1} \tag{5.7}$$

5.2 Definition of Trajectories

5.2.1 Paths and Trajectories

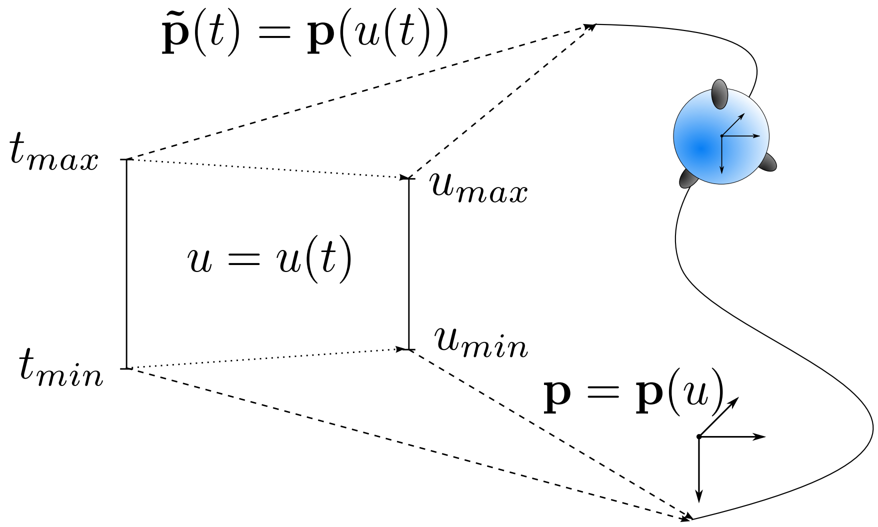
5.2.2 Interpolation and Approximation

If one wants to draw a curve through a set of data points, there exists two ways to do this. First, the curve can pass through all data points no matter how many

[2]For a derivation of curvature see any vetor analysis book, e.g. [13] chapter II, page 71.

[3]The deviation vector between trace and its closest point on the trajectory is always normal to the later.

[4]Time synchronity should be warranted for accurate *trajectory* following. In our task for caputuring time independent imagery, it was only considered as a secondary aspect.



bends it will have, secondly, the curve tries to best fit the data, i.e. a function of a certain order is adopted to best fit the date. This can be done with different methods, e.g with least-squares. Depending on the choice, different curves with different properties are formed (see figure 5.2).

Abbildung 5.2: Interpolation and approximation of waypoints

If the pilot defines the trajectory with a set of waypoints, i.e. data points, he usually wants the UAV to pass through all of them. Therefore the waypoints must be interpolated and not approximated with a suitable curve. Instead of using polynomials of degree BLA BLA, splines were chosen.

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:

$$|a_{max}| = \frac{|F_{res,w}|}{m_{tot}} = 0.96 m/s^2 \tag{5.8}$$

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:

$$|v_{max}| = \sqrt{\frac{|F_{res,w}|}{\frac{1}{2}c_d \rho \pi r^2}} = 2.9 m/s \tag{5.9}$$

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

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

$$|\Psi_{max}| = \frac{|M_{res,w}|}{|\lambda_{max,J_B}|} = 2.06 rad/s^2 \tag{5.10}$$

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[5] are tested to follow the defined trajectories. The *Trajectory following* controller supplies the system's position controller [?] 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[6], 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 to accurate tracking in a safe environment [7].
A position controller with feedforward terms for velocity and acceleration as described in [9] can therefore be used with the reference input

$$[r_{ref}(t),\ \dot{r}_{ref}(t),\ \ddot{r}_{ref}(t)]^T = [\tilde{p}(t),\ \dot{\tilde{p}}(t),\ \ddot{\tilde{p}}(t)]^T \tag{5.11}$$

The controller scheme is shown in figure 5.4.

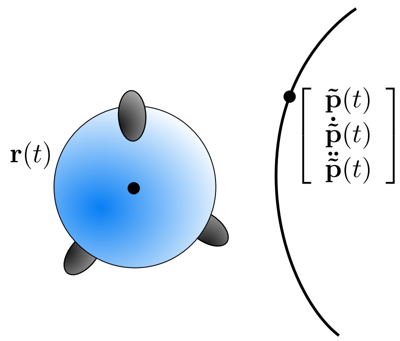


Abbildung 5.3: For a perfect model and a trajectory $\tilde{\mathbf{p}}(t)$ considering all system constraints, the position $\mathbf{r}(t)$ will correctly follow the trajectory.

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

5.5.2 Pure Pursuit Controller

Another commonly used trajectory controller is Pure Pursuit [3]. To consider all dynamics of the trajectory, the reference intput is based on a lookahead point $\tilde{p}(t_{cl} + \Delta T) = \tilde{p}(t_{cl}) + \Delta T \cdot \dot{\tilde{p}}(t_{cl}) + \frac{1}{2}\Delta T^2 \cdot \ddot{\tilde{p}}(t_{cl}) + \mathcal{O}(\Delta T^3)$ which also considers terms of $\mathcal{O}(\Delta T^3)$[7].

[5] [3] provides a good overview to trajectory control.
[6] I.e. saturations of $\dot{r}(t)$ and its derivatives.
[7] Note, that this is simple and robust alternative to any derivative controller.

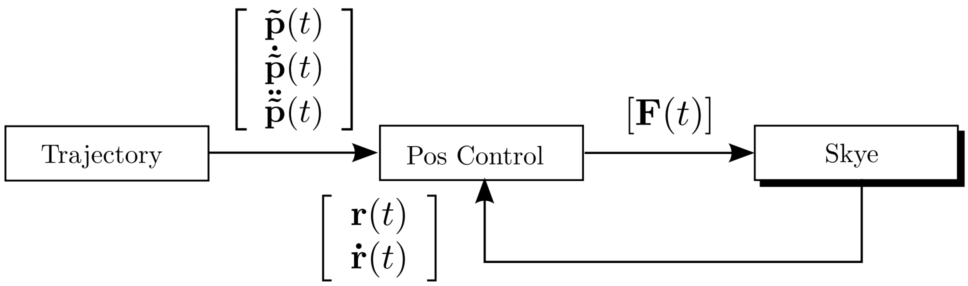


Abbildung 5.4: Trajectory following controller. The value of the parameter t of the trajectory is equal to the current time.

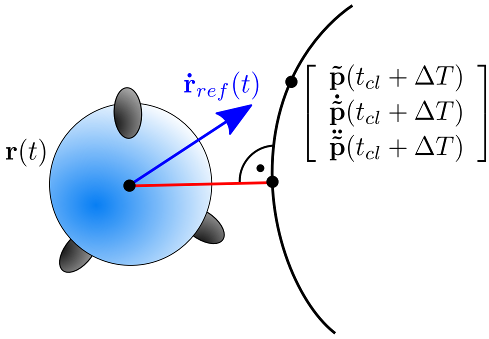


Abbildung 5.5: Pure pursuit yields to extremly awesome tracking.

5.5.3 Cross Track Error Controller

see [5]

5.6 Discussion

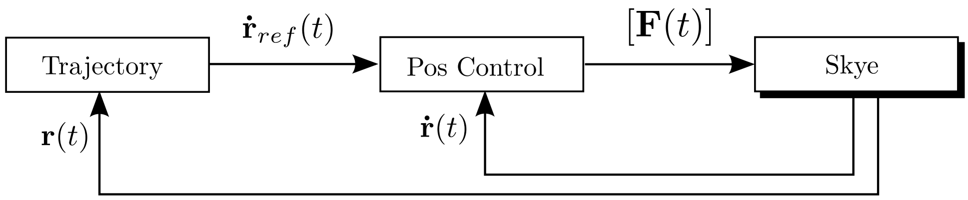


Abbildung 5.6: Pure pursuit yields to extremly awesome tracking.

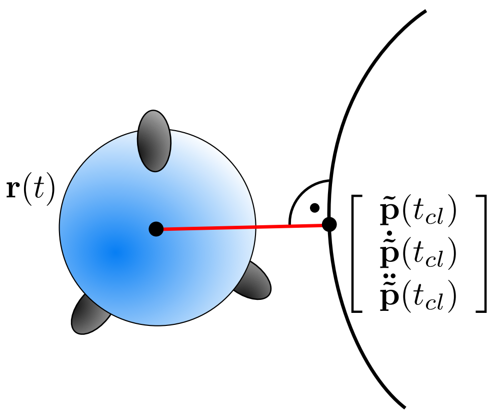


Abbildung 5.7: Cross Track Error Control yields to extremly awesome tracking.

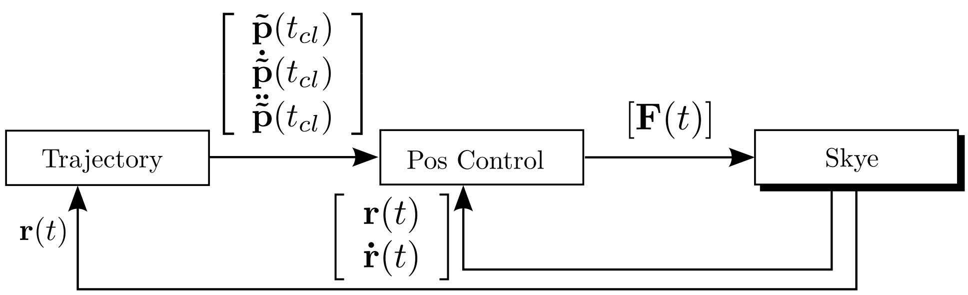


Abbildung 5.8: Cross Track Error Control yields to extremly awesome tracking.

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. Dössegger: *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

[13] U. Stammbach: *Analysis I/II, Teil A.* ETH Zürich, 2005