Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autonomous Systems Lab
Prof. Roland Siegwart

Master-Thesis

Path Planning for Dynamic Maneuvers with Micro Aerial Vehicles

Autumn Term 2014

Supervised by:
Markus Achtelik
Michael Burri

Author:
Fabian Neuschäfer

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Path Planning for Dynamic Maneuvers with Micro Aerial Vehicles

is original work which I alone have authored and which is written in my own words.[1]

Author(s)

First name Last name

Supervising lecturer

First name Last name

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.

Contents

Abstract

The goal of this Master-Thesis is to develop a numerical robust trajectory-planning algorithm for aggressive multi-copter flights in dense environments. The trajectory generated by this algorithm is represented by polynomials which are jointly optimized. The cost function of the optimization consists of the total trajectory-time as well as the total quadratic snap (second derivation of the acceleration). Including the snap into the cost function guaranties a trajectory without abrupt or expensive control inputs.

Furthermore the process of exploring the state space using the Rapidly-Exploring Random Tree (RRT) algorithm is embedded into the numerical robust algorithm. The sampling points oft the RRT (or RRT*) algorithm are then used as the vertices in the polynomial optimization.

Symbols

Symbols

ϕ, θ, ψ	roll, pitch and yaw angle

Indices

x	x axis
y	y axis

Acronyms and Abbreviations

ETH	Eidgenössische Technische Hochschule
UAV	Unmanned Aerial Vehicle
RRT	Rapidly-Exploring Random Tree
QP	Quadratic Programming

Chapter 1

Introduction

1.1 State of the Art

A lot of research has been done in the field of Unmanned Aerial Vehicles (UAV) in the last years leading to a strong improvement in planning [1] as well as in control [[2], [3]]. Another research field is machine learning [4] which is suitable to enhance the performance of aerobatic maneuvers but seams to have a downside regarding motion planning and trajectory generation in dense environments.

Speaking of trajectory planning, there are two different strategies which are pursued. On the one hand, the geometric and the temporal planning are decoupled [5] on the other hand, geometric and temporal information are coupled and the trajectory is the result of a minimization problem. For the couplet problem one can make use of the differential flatness of a quadrocopter to derive constraint on the trajectory. Then formulate a cost-function which could be the trajectory-time [3] or the total snap [6] (second derivation of acceleration).

Another aspect of planning is exploring the state space in the first place. A strong tool to do so are incremental search techniques as for instance the A* [7] or the RRT* algorithm [8]. The sampling points of the solution of the incremental search can then be used as the vertices for the polynomial optimization.

1.2 Quadratic Programming

1.2.1 Constrained Quadratic Programming

Quadratic Programming (QP) is a special case of optimization problem in which a quadratic function is optimized with respect to its optimizations variables (which are represented with the vector x in Equation 1.1)

$$f(x) = \frac{1}{2} \cdot x^T Q x + c^T x \tag{1.1}$$

The optimization is performed under linear constraints on the optimizations variables. Whereas a distinction between equality($E\mathbf{x} = \mathbf{d}$) and inequality constraints ($A\mathbf{x} \leq \mathbf{b}$) has to be made.
In case there are only equality constrains, the solution to the QP is given by the linear system in Equation 1.2 :

$$\begin{bmatrix} Q & E^T \\ E & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x} \\ \lambda \end{bmatrix} = \begin{bmatrix} -\mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{1.2}$$

1.2.2 Unconstrained Quadratic Programming

The constrained QP gets ill-conditioned for a large amount of segments or for high order polynomials (which both lead to large and sometimes sparse matrices). To reduce the number of optimization variables, and therefore the size of the matrices, the constrained QP can be converted into a unconstrained QP.
In other words, the polynomial coefficients are no longer the optimization variables but the free endpoint derivatives are optimized. The exact formulas of the unconstrained QP can be seen in Section (..)

insert section !!!!!!!!!!!!!!!!!!!!!!

Chapter 2

Polynomial Trajectory Optimization

2.1 Polynomial Trajectory

Regarding the differentiability of polynomials, they are a profound choice to represent a trajectory. Especially for the use in a differentially flat representation of the UAV dynamics. (Flatness in the proper sense of system theory means that all the states and inputs can be expressed in terms of the flat output and a finite number of its derivative).
Furthermore, the differentiability of polynomials enables the possibility to check the derivatives of the trajectory for bounding violations to avoid input saturation. This saturation-check can be perform during trajectory optimization and therefore guarantees the feasibility of the resulting trajectory.

2.2 Optimization

The goal is to optimize a trajectory which passes through way-points (also called vertices) which are defined in advance. This way-points can be chosen manually or by a path-finding algorithm such as RRT* which will be discussed in Chapter [RRT* ref einsetzten!!!!]
Furthermore, not only the way-points (therefore the position) can be fixed in advance but also its derivatives (such as speed, acceleration etc.). The position and its derivatives are then utilized as the equality constrains for a QP (explained in Section 1.2).

2.2.1 Cost Function

Optimization in terms of trajectory planning means minimization of a cost function. The cost function in this case is a combination of temporal and geometric cost. The geometric cost penalizes the (square) of the derivatives of the trajectory. In this Master Thesis the geometric cost is represented by the squared snap which guarantees a trajectory without abrupt control inputs.
The temporal cost is simply the total trajectory-time multiplied by a user chosen factor k_T which determines the aggressiveness of the resulting trajectory.

To express the geometric cost in a compact way on can make use of the Hessian matrix Q. The Hessian matrix is defined as a squared matrix of second-order partial derivatives which follows from differentiation a function with respect to each of its coefficients (in this instance the polynomial coefficients). The geometric cost function $J(T)$ for a fixed time for one segment can now be written as

$$J(T) = p^T \cdot Q(T) \cdot p \tag{2.1}$$

where $Q(T)$ is the Hessian matrix for a fixed segment-time T and p is the vector containing the coefficients of the polynomial.

If the trajectory consists of more than one segment the Hessian matrix has to extended to a block-diagonal matrix and the geometric cost function for multiple segments with fixed bud individual segment-times can be written as

$$J = \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix}^T \cdot \begin{bmatrix} Q_1(T_1) & & \\ & \ddots & \\ & & Q_n(T_n) \end{bmatrix} \cdot \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} \tag{2.2}$$

2.2.2 Polynomial Optimization as a Constrained QP

In a first, intuitive approach the equality constraints on the endpoint derivatives (mentioned in Section 2.2) are utilized in a constrained QP. Therefore a mapping matrix E between endpoint derivatives and polynomial coefficients is needed. The resulting formula for the i^{th} segment can be written as

$$E_i \cdot p_i = d_i \tag{2.3}$$

where p is the vector containing the polynomial coefficients and d is the vector containing the endpoint derivatives. Regarding the total number of segments of the trajectory, Formula 2.3 can be written in matrix form:

$$\begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_n \end{bmatrix} \cdot \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ d_n \end{bmatrix} \tag{2.4}$$

The constrained QP is suitable for a small amount of segments but gets ill-conditioned for a large amount of segments and therefore large matrices. Especially if there are matrices which are close to singularity and have coefficients which are close to zero, the constrained QP can get numerical unstable.

2.2.3 Polynomial Optimization as a Unconstrained QP

To avoid the numerical instability of a constrained QP the optimization problem is converted into a unconstrained QP. Therefore the polynomial coefficients p_i from Formula 2.2 have to be substituted by the endpoint derivatives d_i which are now the new optimizations variables. The cost function of the unconstrained QP can now be written as

$$J = \begin{bmatrix} d_1 \\ \vdots \\ d_n \end{bmatrix}^T \cdot \begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_n \end{bmatrix}^{-T} \cdot \begin{bmatrix} Q_1 & & \\ & \ddots & \\ & & Q_n \end{bmatrix} \cdot \begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_n \end{bmatrix}^{-1} \cdot \begin{bmatrix} d_1 \\ \vdots \\ d_n \end{bmatrix} \tag{2.5}$$

where Q_i is the Hessian matrix according to the i^{th} segment-time.

As mentioned above, the endpoint derivatives are the new optimization variables. Do to the equality constrains some of the endpoint derivatives are already specified consequently reducing the number of optimizations variables. Expediently, the endpoint derivatives are divided in fixed derivatives d_f and unspecified derivatives d_p and then reordered using the matrix C which consists of zeros and ones. After reordering the endpoint derivatives Formula 2.5 can be rewritten as

$$J = \begin{bmatrix} d_f \\ d_p \end{bmatrix}^T \underbrace{C^T E^{-T} Q E^{-1} C}_{\text{R}} \begin{bmatrix} d_f \\ d_p \end{bmatrix} \tag{2.6}$$

where the product of the reordering matrix C, the mapping matrix E and the Hessian matrix Q han be expressed as a single Matrix R. The matrix R for his part can be defided into four submatrices according to the fixed and unspecified endpoint derivatives which modifies Formula 2.6 as follows:

$$J = \begin{bmatrix} d_f \\ d_p \end{bmatrix}^T \begin{bmatrix} R_{ff} & R_{fp} \\ R_{pf} & R_{pp} \end{bmatrix} \begin{bmatrix} d_f \\ d_p \end{bmatrix} \tag{2.7}$$

Partially differenting Formula 2.7 with respect to the unspecified derivatives d_p and equate it to zero yields the optimized/minimized unspecified derivatives d_p^*

$$d_p^* = -R_{pp}^{-1} \cdot R_{fp}^T \cdot d_f \tag{2.8}$$

as a function of the fixed derivatives d_f and two of the submatrixes (R_{pp}, R_{fp}) of R.

2.2.4 Penalty on Time

So fare, only the geometric cost (i. e. the squared snap) was discussed. Minimization of the geometric cost ensures a smooth trajectory without abroupt input signal but has no effect on the agressivity of a trajectory. Therefore Formula 2.7 has to be extended by the temporal cost which results in the total cost J_{total}:

$$J_{total} = \begin{bmatrix} d_f \\ d_p \end{bmatrix}^T \begin{bmatrix} R_{ff} & R_{fp} \\ R_{pf} & R_{pp} \end{bmatrix} \begin{bmatrix} d_f \\ d_p \end{bmatrix} + k_T \cdot \sum_{i=1}^{N} T_i \tag{2.9}$$

where k_T is a user specified penalty on time and T_i ist the segment-time of the i^{th} segment.

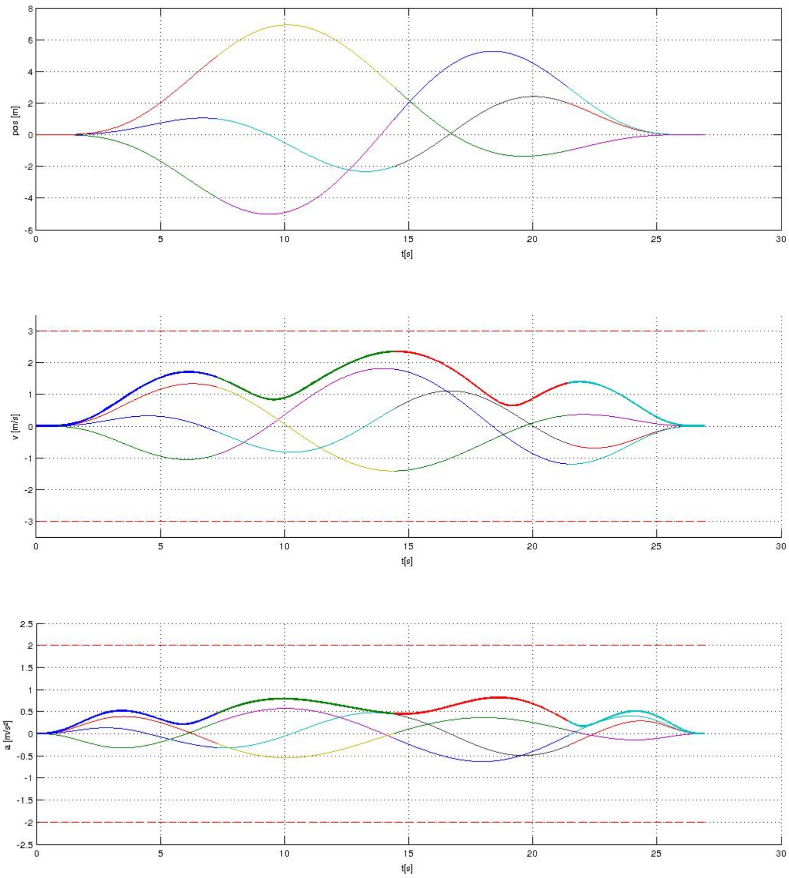

Figure 2.1: Ein Bild.

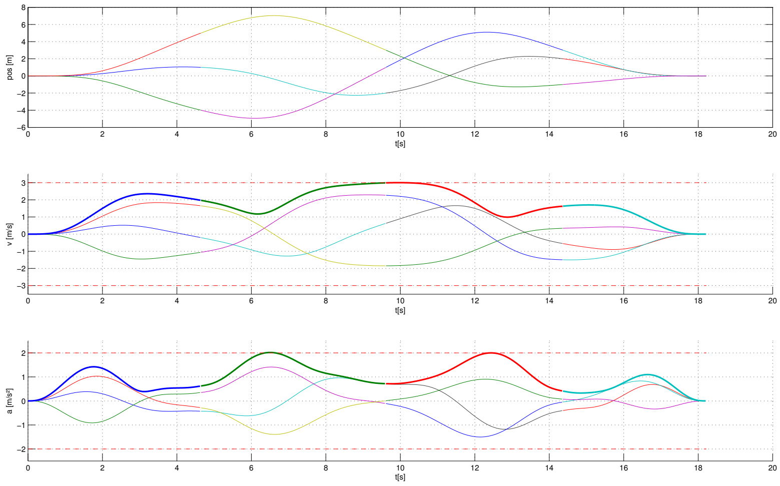

Figure 2.2: Ein Bild.

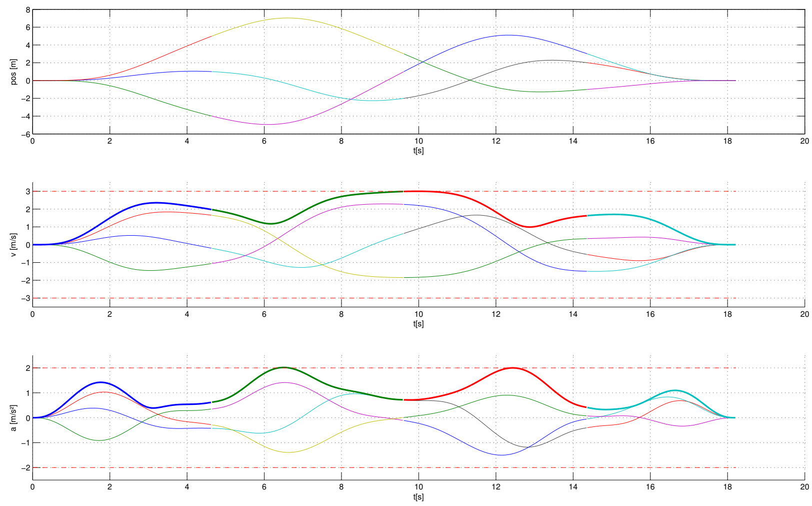

Figure 2.3: Ein Bild.

Chapter 3

RRT

3.1 General

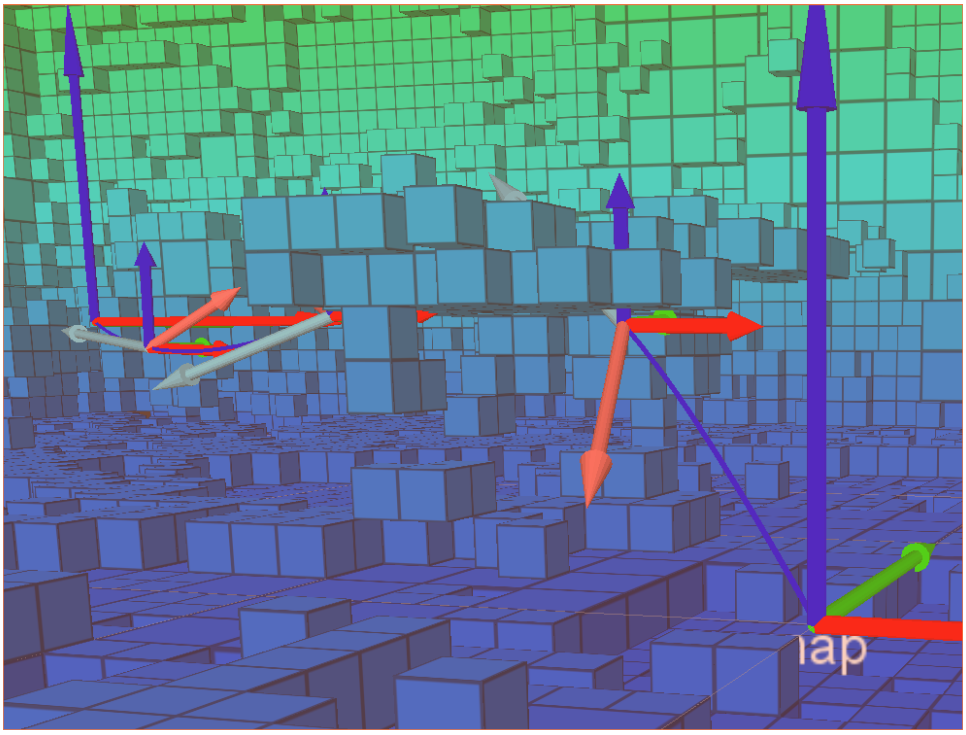

Figure 3.1: Ein Bild.

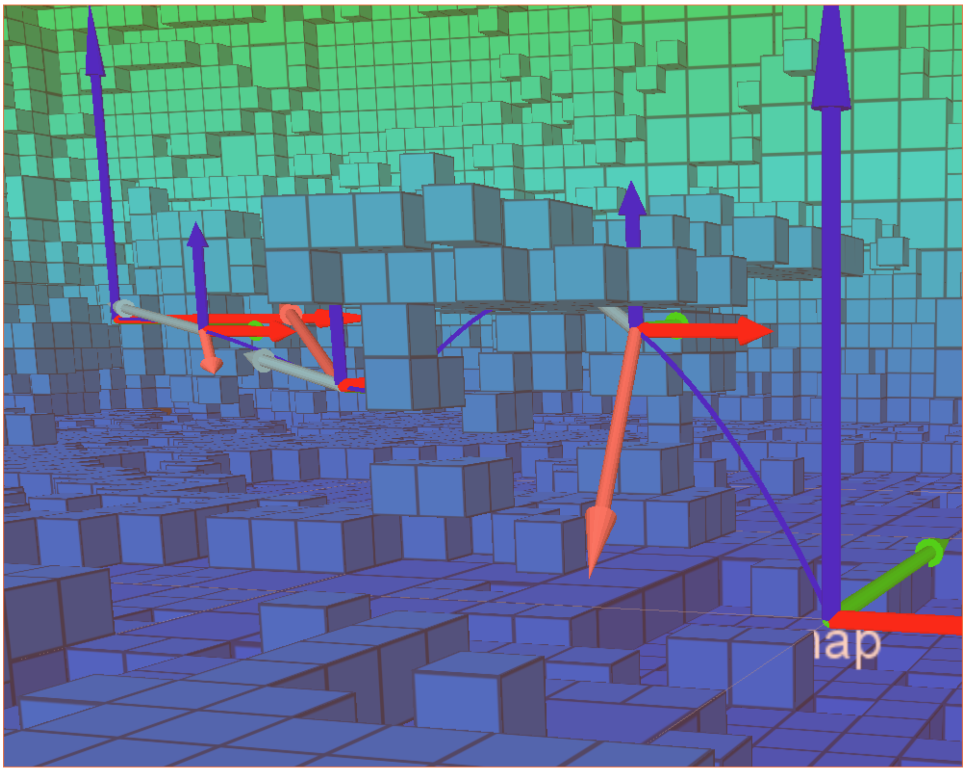

Figure 3.2: Ein Bild.

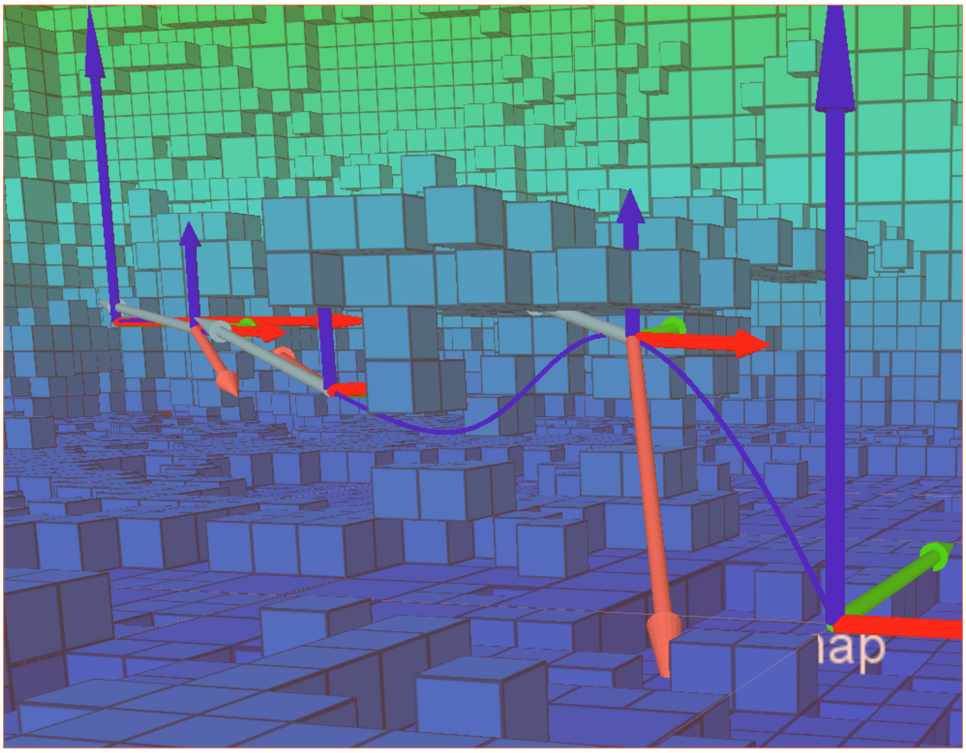

Figure 3.3: Ein Bild.

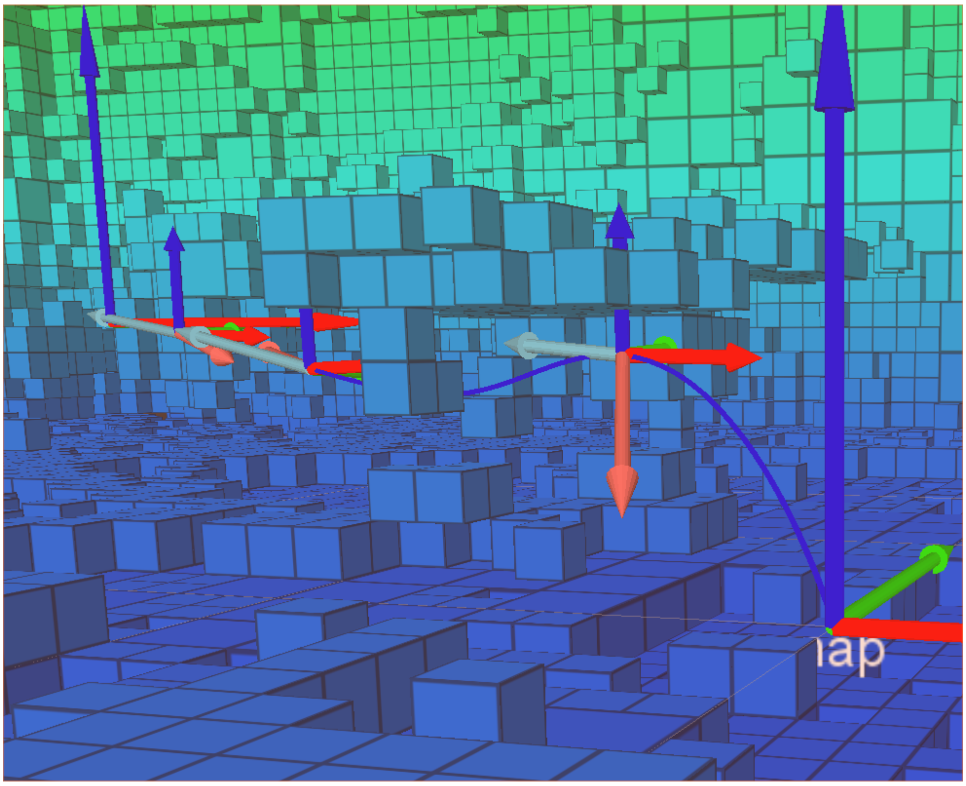

Figure 3.4: Ein Bild.

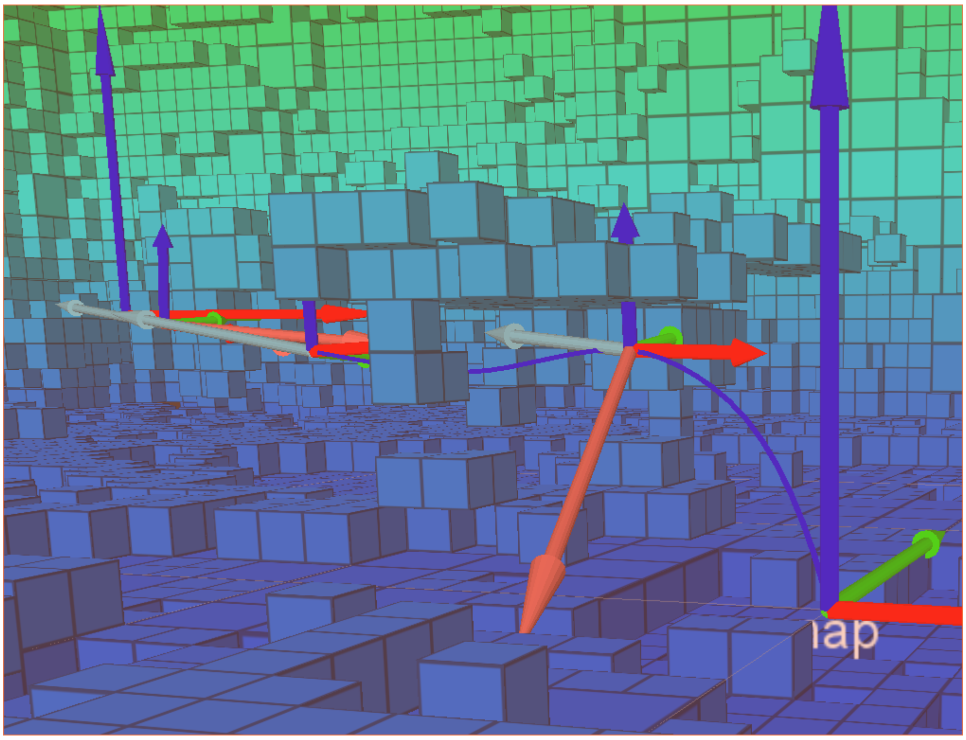

Figure 3.5: Ein Bild.

Chapter 4

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 hï¿½ngt auch stark vom verwendeten Compiler ab. Typisches Format fï¿½r Bilder in LaTeX ist EPS[1].

4.1 Gliederungen

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

4.2 Referenzen und Verweise

Literaturreferenzen werden mit dem Befehl `\cite{.}` erzeugt. Ein Beispiel: [**?**].
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 4.

4.3 Aufzï¿½hlungen

Folgendes Beispiel einer Aufzï¿½hlung ohne Numerierung,

- Punkt 1
- Punkt 2

wurde erzeugt mit:

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

Folgendes Beispiel einer Aufzï¿½hlung mit Numerierung,

[1] Encapsulated Postscript
[2] Bla bla.

1. Punkt 1
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}
```

4.4 Erstellen einer Tabelle

Ein Beispiel einer Tabelle:

Table 4.1: 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}
```

4.5 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}
```

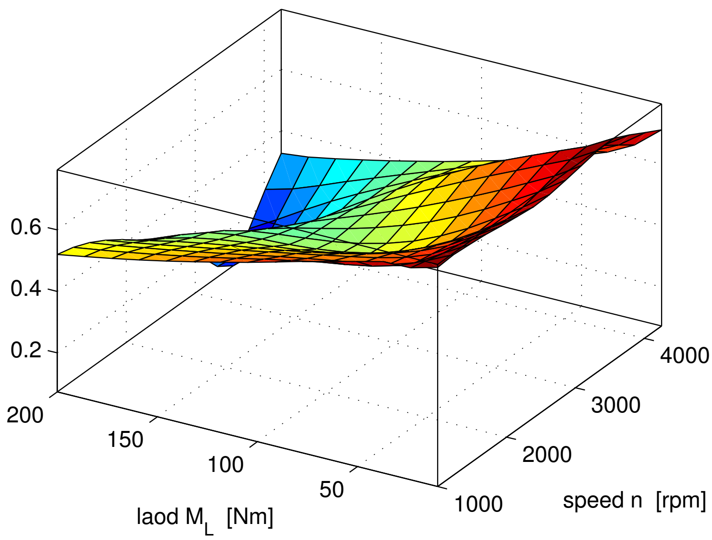

Figure 4.1: Ein Bild.

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.

4.6 Mathematische Formeln

Einfache mathematische Formeln werden mit der equation-Umgebung erzeugt:

$$p_{me0f}(T_e, \omega_e) \;=\; k_1(T_e) \cdot (k_2 + k_3 S^2 \omega_e^2) \cdot \Pi_{max} \cdot \sqrt{\frac{k_4}{B}}\,. \tag{4.1}$$

Der Code dazu lautet:

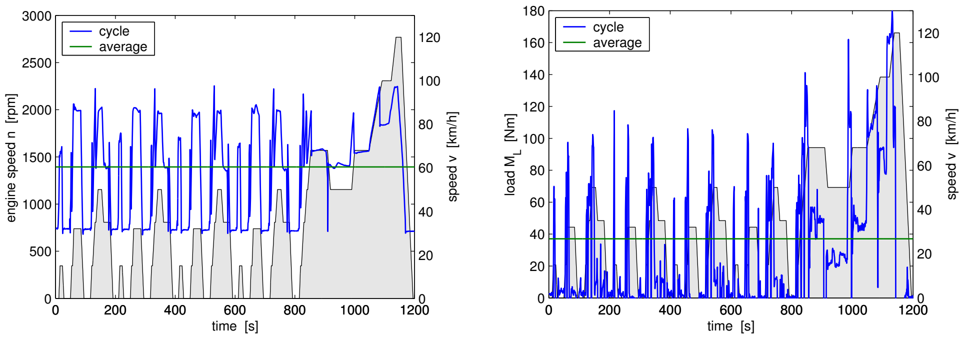

Figure 4.2: Zwei Bilder nebeneinander.

```
\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 Ausdrï¿½cke im Text werden mit $formel$ erzeugt (zB: $a^2+b^2 = c^2$).

4.7 Weitere nï¿½tzliche Befehle

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

Appendix A

Irgendwas

Bla bla . . .

Appendix B

Nochmals irgendwas

Bla bla ...

Bibliography

[1] R. He, A. Bachrach, M. Achtelik, A. Geramifard, D. Gurdan, S. Prentice, J. Stumpf, and N. Roy: *On the design and use of a micro air vehicle to track and avoid adversaries*. The Int. Journal of Robotics Research, vol. 29, pp. 529-546, 2010.

[2] D. Colling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke: *A prototype of an autonomous controller for a quadrotor UAV*. In Proceedings of the European Control Conference, Kos, Greece, 2007, pp. 1-8.

[3] M. Hehn and R. D'Andrea : *Quadrocopter trajectory generation and control*. In International Federation of Automatic Control (IFAC), World Congress 2011, 2011.

[4] S. Lupashin, A. Schollig, M. Sherback, and R. D'Andrea: *A simple learning strategy for high-speed quadrocopter multi-flips*. In Proc. of the IEEE Int. Conf. on Robotics and Automation, Anchorage, AK, May 2010, pp. 1642-1648.

[5] Y. Bouktir, M. Haddad, and T. Chettibi: *Trajectory Planning for a Quadrotor Helicopter*. In Mediterranean Conference on Control and Automation, Jun. 2008, pp. 1258-1263

[6] D. Mellinger and V. Kumar: *Minimum snap trajectory generation and control for quadrotors*. In International Conference on Robotics and Automation, 2011, pp. 2520-2525.

[7] M. Likhachev, G. Gordon and S. Thrun: *ARA*: Anytime A* with Provable Bounds on Sub-Optimality* . Advances in Neural Information Processing Systems, vol. 16, 2003

[8] C. Richter, A. Bry, and N, Roy: *Polynomial Trajectory Planning for Quadrotor Flight*. In International Conference on Robotics and Automation, 2013.