Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ROBOTICS & PERCEPTION GROUP

Alessio Zanchettin

Autonomous landing on a moving platform

Master Thesis

Robotics and Perception Group
University of Zurich

Supervision

First Supervisor Davide Falanga
Second Supervisor
Prof. Dr. Davide Scaramuzza

September 2016

Contents

Abstract

Compress the introduction in a few key sentences. No more than half a page. The abstract should motivate your work, outline the work that you did, and give some insights into its results.

Nomenclature

Notation

$\mathbf{J}$	Jacobian
$\mathbf{H}$	Hessian
$\mathbf{T}_{WB}$	coordinate transformation from frame B to frame W
$\mathbf{R}_{WB}$	orientation of B with respect to W
${}_W\mathbf{t}_{WB}$	translation of B with respect to W, expressed in coordinate system W

Scalars are written in lower case letters (a), vectors in lower case bold letters ($\mathbf{a}$) and matrices in upper case bold letters ($\mathbf{A}$).

Acronyms and Abbreviations

RPG	Robotics and Perception Group
DoF	Degree of Freedom
IMU	Inertial Measurement Unit
MAV	Micro Aerial Vehicle
ROS	Robot Operating System

Chapter 1

Introduction

Describe the problem and the motivation for this research.

MBZIRC challenge

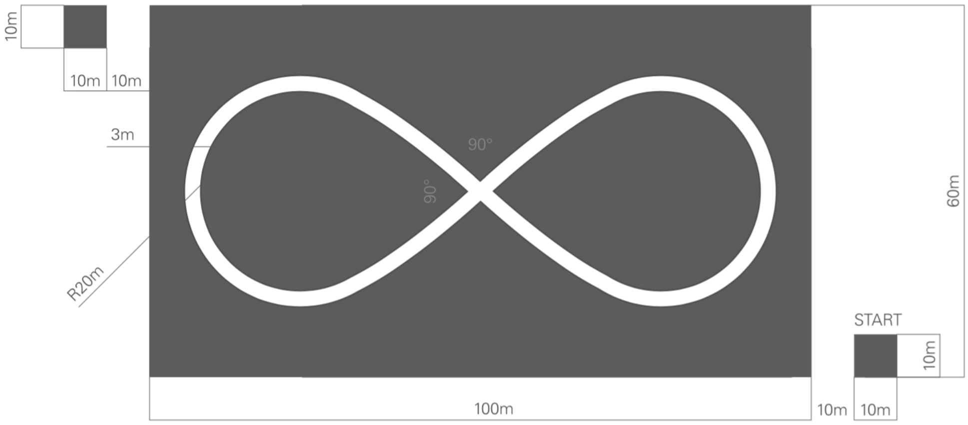

Figure 1.1: Arena of the challenge

1.1 Related Work

Describe the current state of the art. Provide all necessary citations.

Chapter 2

General Framework

2.0.1 SVO & MSF

are calculating the best estimation of the state of quadrotor based on data from a camera and an IMU

SVO front looking camera

explanation about base in fov
Fish eye or not fish eye? pros and cons

2.0.2 Base Detection & Base Tracking

given images taken from a camera on the quadrotor, it is searching the landing platform and estimating its state

2.0.3 Area Exploration & Trajectory Generation

considering the state of the quadrotor and of the landing platform, they are calculating where the quadrotor must go and with which trajectory.

2.0.4 Flight Control & Copilot

given the desired trajectory, they are computing the controls that must be applied to the quad in order to follow such trajectory

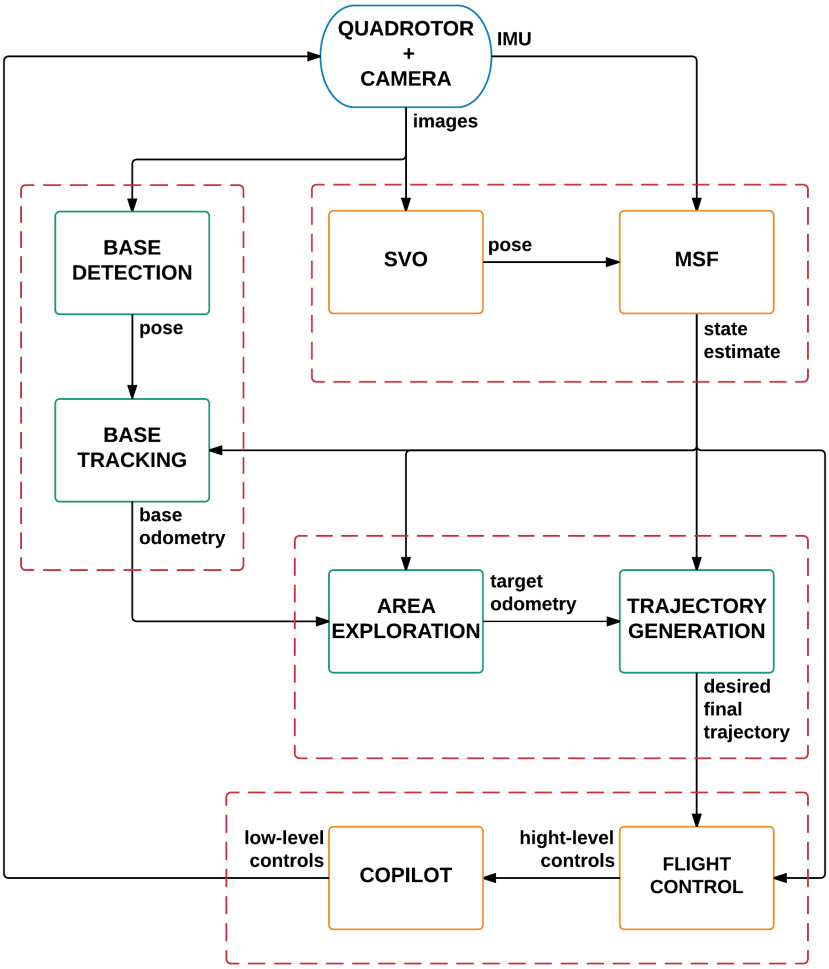

Figure 2.1: Pipeline

Chapter 3

Base Tracking

3.1 Extended Kalman Filter

One part of the work is focused on the estimation of the state of the moving platform. An Extended Kalman Filter is design in order to have the most reliable value of the state of the platform.

Kalman filtering is an algorithm that uses a series of noisy measurements observed over time and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each time frame.
The algorithm works in a two-step process:

- In the prediction step, the Kalman filter produces estimates of the current state variables, along with their uncertainties, based on a model of the system:

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{w}_k \tag{3.1}$$

- Once the outcome of the next measurement is observed:

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k) + \boldsymbol{v}_k \tag{3.2}$$

  these estimates are updated using a weighted average, with more weight being given to estimates with higher certainty.

In the extended Kalman filter, the state transition and observation models don't need to be linear functions of the state but may instead be differentiable functions.
($\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are the process and observation noises which are both assumed to be zero mean multivariate Gaussian noises with covariance $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ respectively. $\boldsymbol{u}_k$ is the control vector).

The algorithm is recursive. It can run in real time, using only the present input measurements and the previously calculated state and its uncertainty matrix;

no additional past information is required.
The Kalman filter does not require any assumption that the errors are Gaussian. However, the filter yields the exact conditional probability estimate in the special case that all errors are Gaussian-distributed.
Initialization

$$\begin{aligned} \boldsymbol{x}_{0|0} &= x_0 \\ \boldsymbol{P}_{0|0} &= P_0 \end{aligned} \tag{3.3}$$

In this case the prediction equations are continuous in time, so for the prediction step of the EKF we have to solve:

$$\begin{aligned} \dot{\hat{\boldsymbol{x}}}(t) &= f(\hat{\boldsymbol{x}}(t), \boldsymbol{u}(t)) \\ \dot{\boldsymbol{P}}(t) &= \boldsymbol{F}(t)\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{F}(t)^\top + \boldsymbol{Q}(t) \end{aligned} \tag{3.4}$$

for $t \in (t_{k-1}, t_k)$ where

$$\begin{aligned} \hat{\boldsymbol{x}}(t_{k-1}) &= \hat{x}_{k-1|k-1} \\ \boldsymbol{P}(t_{k-1}) &= P_{k-1|k-1} \\ \boldsymbol{F}(t) &= \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}(t), \boldsymbol{u}(t)} \\ \hat{\boldsymbol{x}}_{k|k-1} &= \hat{\boldsymbol{x}}(t_k) \\ \boldsymbol{P}_{k|k-1} &= \boldsymbol{P}(t_k) \end{aligned} \tag{3.5}$$

In order to save some computation we can discretize the dynamicin order to have shorter computation during the prediction step of the EKF:

$$\begin{aligned} \hat{\boldsymbol{x}}_{k|k-1} &= f(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k) \\ \boldsymbol{P}_{k|k-1} &= \boldsymbol{F}_{k-1}\boldsymbol{P}_{k-1|k-1}\boldsymbol{F}_{k-1}^\top + \boldsymbol{Q}_k \end{aligned} \tag{3.6}$$

where the state transition matrix is defined to be the following Jacobians:

$$\boldsymbol{F}_{k-1} = \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k} \tag{3.7}$$

While the update equations are discrete in time and they yield to the following update step:

$$\begin{aligned} \boldsymbol{K}_k &= \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top + \boldsymbol{R}_k)^{-1} \\ \hat{\boldsymbol{x}}_{k|k} &= \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k(\boldsymbol{z}_k - h(\hat{\boldsymbol{x}}_{k|k-1})) \\ \boldsymbol{P}_{k|k} &= (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k|k-1} \end{aligned} \tag{3.8}$$

where the observation matrix is defined to be the following Jacobian:

$$\boldsymbol{H}_k = \left.\frac{\partial h}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{k|k-1}} \tag{3.9}$$

3.2 Prediction update: non-holonomic model

The platform is considered as a car and simulated with a non-holonomic model. In this model the state is defined as $\boldsymbol{x} = (x, y, z, \theta, v, \phi)$: It corresponds to the 3 position in a space (x, y, z) and the yaw angle of the platform (θ) w.r.t. the world frame, the forward velocity (v) and the angle of the front wheels (ϕ). The system depends on a parameter L that corresponds to the distance between the front and the back wheels.
In this model the control input are the change in velocity v and in the angle of curvature ϕ.
The equation of motion in continuous time are:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$$

$$\begin{aligned}
\dot{x} &= vcos(\theta) \\
\dot{y} &= vsin(\theta) \\
\dot{z} &= 0 \\
\dot{\theta} &= \frac{v}{L}tan(\phi) \\
\dot{v} &= u_1 \\
\dot{\phi} &= u_2
\end{aligned} \tag{3.10}$$

It is possible to discretize these dynamics in $t \in (t_{k-1}, t_k)$ with a first order finite difference:

$$\dot{\boldsymbol{x}} \approx \frac{\boldsymbol{x}_k - \boldsymbol{x}_{k-1}}{dt} \approx f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k)$$

with $\boldsymbol{x}_k = \boldsymbol{x}(t_k)$, $\boldsymbol{x}_{k-1} = \boldsymbol{x}(t_{k-1})$, $dt = t_k - t_{k-1}$

$$\begin{aligned}
x_k &= x_{k-1} + dt\big(v_{k-1}cos(\theta_{k-1})\big) \\
y_k &= y_{k-1} + dt\big(v_{k-1}sin(\theta_{k-1})\big) \\
z_k &= z_{k-1} \\
\theta_k &= \theta_{k-1} + dt\Big(\frac{v_{k-1}}{L}tan(\phi_{k-1})\Big) \\
v_k &= v_{k-1} + dt\big(u_{1k}\big) \\
\phi_k &= \phi_{k-1} + dt\big(u_{2k}\big)
\end{aligned} \tag{3.11}$$

In order to solve the former system, we have anyway to find a numerical solution. For this purpose we use a RUNGE-KUTTA scheme

3.2.1 Straight and circular path

For now we assume the input u_1 and u_2 are equal to zero, so the platform can be static ($v_f(0) = 0$) can move in a straight line ($v_f(0) \neq 0$ and $\phi(0) = 0$) or in a circle ($v_f(0) \neq 0$ and $\phi(0) \neq 0$)

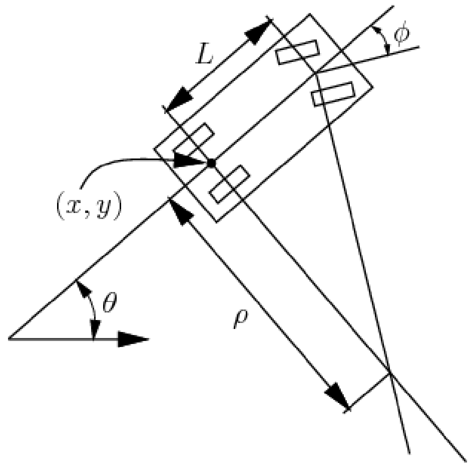

Figure 3.1: Non-holonomic model

3.2.2 Infinity shape path

In the MBZIRC challenge the moving platform will move in an infinity-shape path described in the figure **??**. We need to describe in a mathematical way this shape in order to use this information in the prediction step of the EKF. From the specification of the challenge:

- the car is moving with constant velocity v along the path
- the radius of the trajectory is rm
- the path is making a cross in the middle that creates 4 angles of $\pi/2$

The easiest way to describe this path is to define how the angle θ is changing in function of the space.
It easy to see that the shape can be seen as a combination of a cross and two circles. The cross is simply defined as the union between the two line:

$$\begin{aligned} y &= x \\ y &= -x \end{aligned} \tag{3.12}$$

while the two circles

$$\begin{aligned} y^2 + (x - x_0)^2 &= r^2 \\ y^2 + (x + x_0)^2 &= r^2 \end{aligned} \tag{3.13}$$

It easy to see that if we want the intersections between these two functions to be exactly in the 4 points we have to choose

$$x_0 = \frac{\sqrt{2}}{2} r \tag{3.14}$$

That correspond to the 4 intersections coordinate

$$\Big(\frac{\sqrt{2}}{2}r, \frac{\sqrt{2}}{2}r\Big); \Big(\frac{\sqrt{2}}{2}r, -\frac{\sqrt{2}}{2}r\Big); \Big(-\frac{\sqrt{2}}{2}r, -\frac{\sqrt{2}}{2}r\Big); \Big(-\frac{\sqrt{2}}{2}r, \frac{\sqrt{2}}{2}r\Big) \tag{3.15}$$

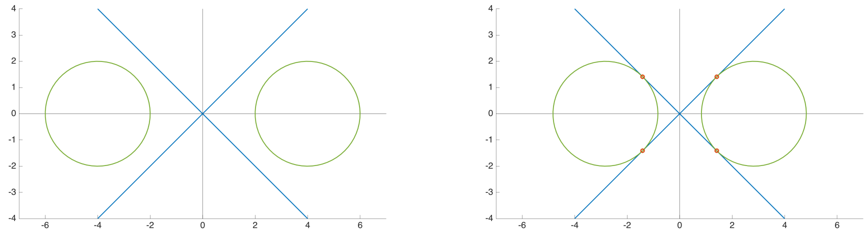

Figure 3.2: How to construct the infinity-shape path

If we travel over the two circumferences the intersections correspond to angles $\theta = \pm\frac{3\pi}{4}$.
Now it is obvious to see that the path is symmetric and it can be divided in 4 parts and describing how the angle is changing in one of this section, the whole trajectory is defined.
We can observe that:

$$\theta(x) = \begin{cases} -\frac{x}{r} & x \in \left[0, \frac{3\pi}{4}r\right] \\ -\frac{3\pi}{4} & x \in \left[\frac{3\pi}{4}r, \frac{3\pi}{4}r + r\right] \end{cases} \tag{3.16}$$

This function define a quarter of the trajectory **??** in function of the radius r of the path.
It is now possible to use it to generate the entire trajectory $(x(t), y(t))$**??**: we know that the length of the path is $l = 4(\frac{3\pi}{4}r + r)$ and given the constant velocity v we can calculate the time to complete the trajectory $T = \frac{l}{v}$ and it is simple to define $\theta(t)$ just stretching or shrinking $\theta(x)$.
So we can define

$$\begin{aligned} \dot{x} &= vcos(\theta(t)) \\ \dot{y} &= vsin(\theta(t)) \end{aligned} \tag{3.17}$$

And finally we have

$$\begin{aligned} x_k &= x_{k-1} + dt\big(v_{k-1}cos(\theta_{k-1})\big) \\ y_k &= y_{k-1} + dt\big(v_{k-1}sin(\theta_{k-1})\big) \end{aligned} \tag{3.18}$$

3.3 Measurement update

3.3.1 Tag Detector

Base design

image

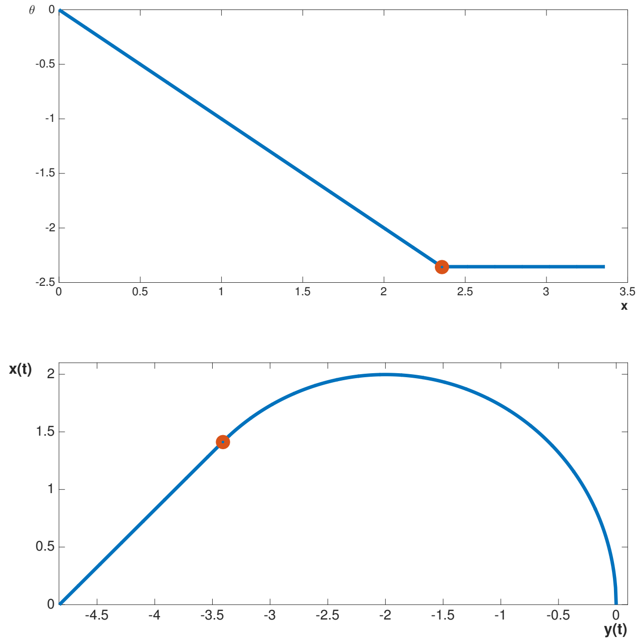

Figure 3.3: The parametrization of a quarter of the path

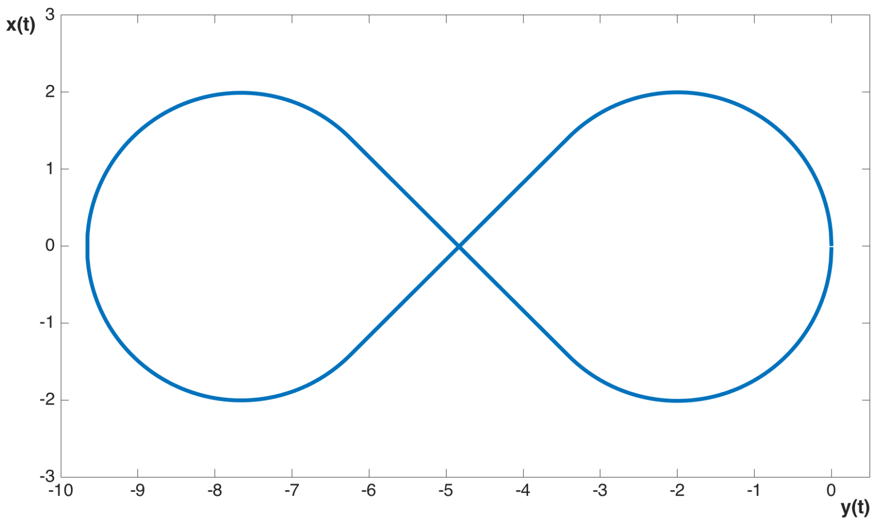

Figure 3.4: Infinity-shape path

April Tag vs Ar Sys

precision comparison vs frequency

Implementation

nodlet

3.3.2 Cross Detector

pnp problem

3.3.3 Covariance Estimation

3.4 Results

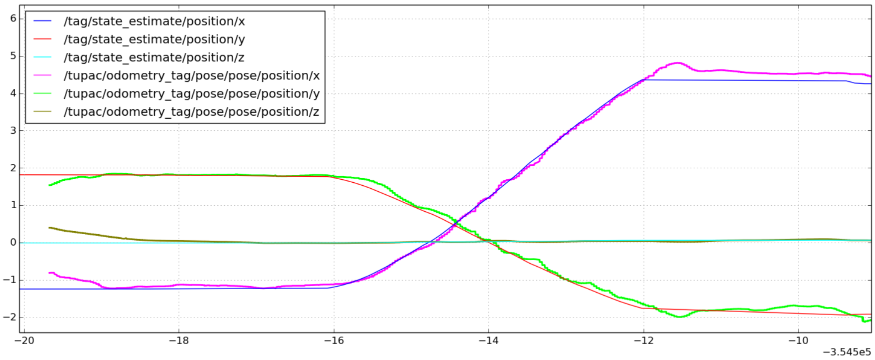

Figure 3.5: EKF 1

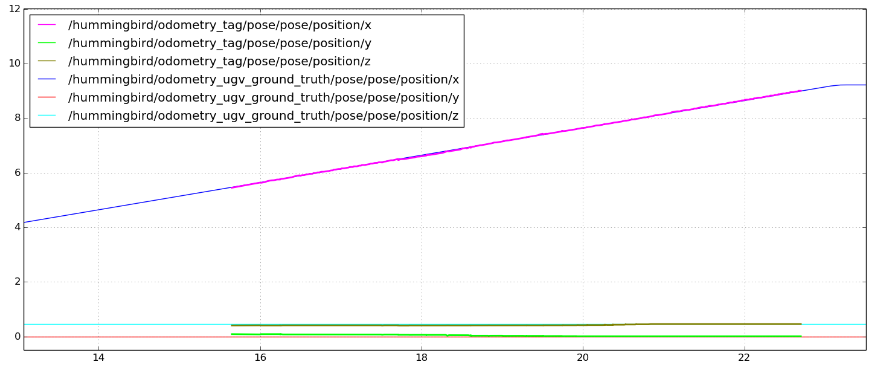

Figure 3.6: EKF 2

Chapter 4

Area exploration

different time exploration imu to switch off time following

4.1 State machine

4.1.1 Phases

First phase - Searching for the base

In this phase the quadrotor starts from a given position and has to find the moving car. Given the rectangle in which the platform can move the UAV follows a list of way-points in order to span the whole area at high altitude. In this way the downward camera can collect information from a large section of the space and the searching of the base is faster.
To find the platform we have the following assumptions:

- the platform is the only white square moving on the arena
- in a small period of time the movement of the platform is approximable with a linear motion

Base on these assumptions, we analyze the images from the down looking camera to find a moving white square and calculate its optical flow to predict its future position. We perform the following passages:

- threshold the image in order to find the white features
- find the edges with the canny edge algorithm
- find the connected edges that define contours of shapes
- select only the shapes with 4 edges
- check if the edges have the same length
- check if the angle between edges is $\frac{\pi}{2}$

At this point we have the position of the squares in the image.
Now we try to calculate the optical flow of these points through the sequence of images and we track only the points that are moving with a velocity comparable to the one known. We can calculate the direction of the platform and then predict where it will be after a time t.

Second phase - Approaching the base

Second phase - Following the base

Second phase - Landing on the base

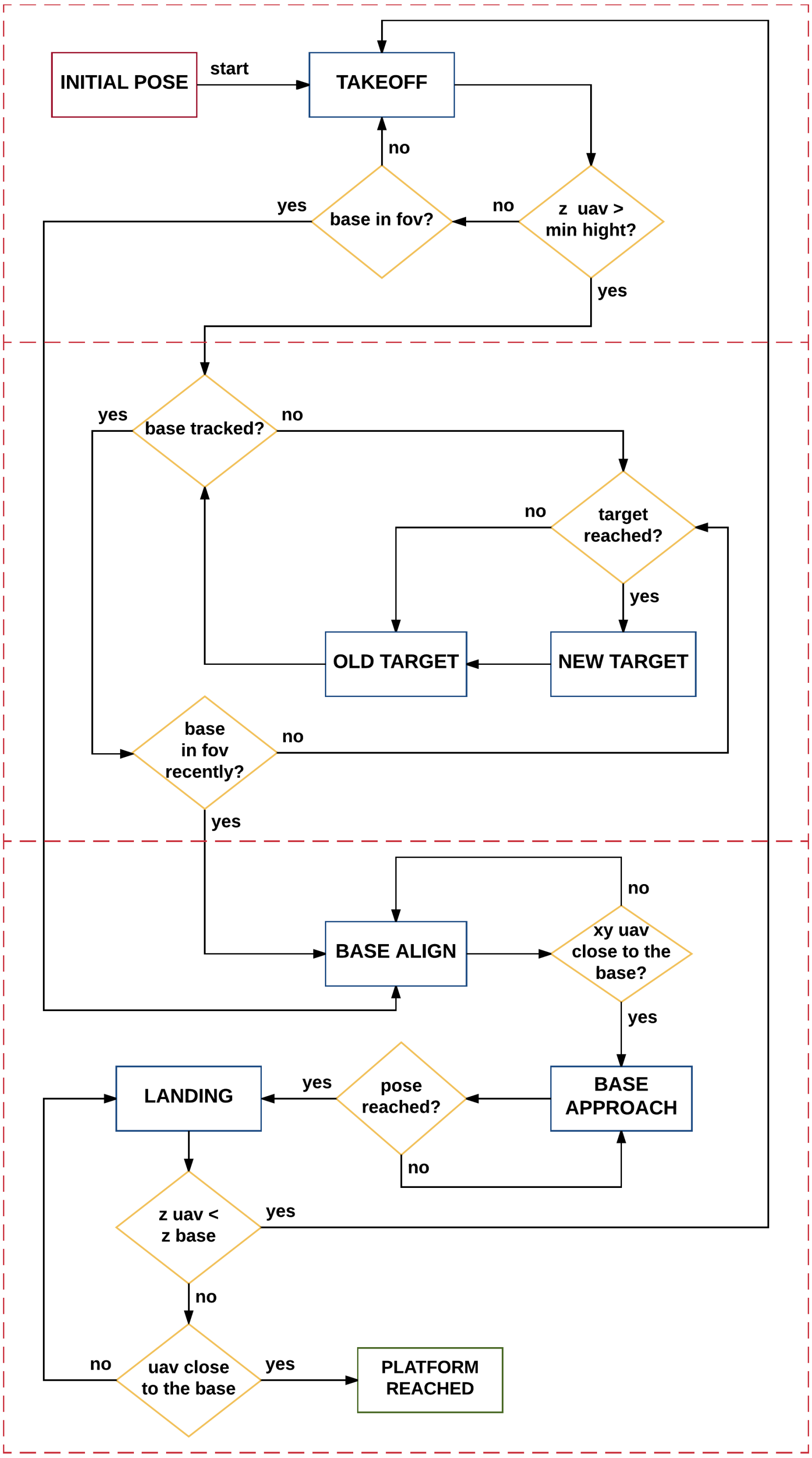

Figure 4.1: Area Exploration

Chapter 5

Trajectory Generator

5.1 Base trajectory prediction

5.2 Rapid Trajectory

how to compute the acceleration... comparision between diff imu and thrust

Compute the acceleration

The Rapid trajectory generator needs an initial and a final state. The initial state is always selected as the current position velocity and acceleration of the quadrotor. From the state estimate of MSF we have the first two information, while we have to compute the acceleration.
There are several ways to make this calculation:

- IMU measurements:
- finite difference of velocity:
- total thrust:

5.3 Results

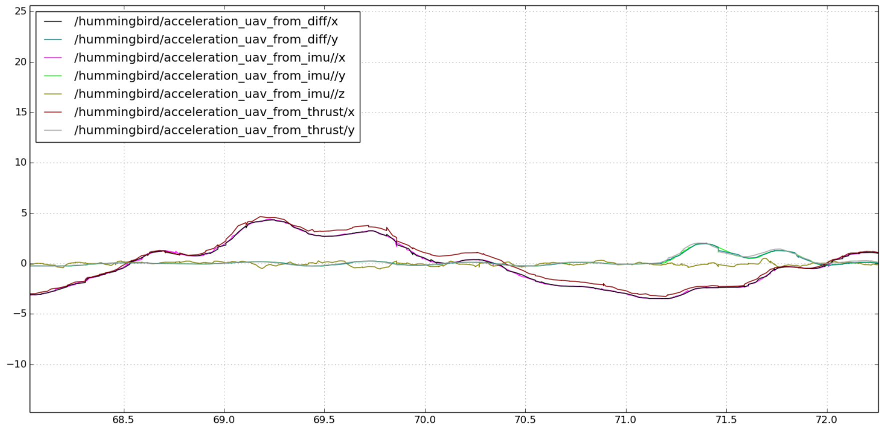

Figure 5.1: Comparison Acceleration

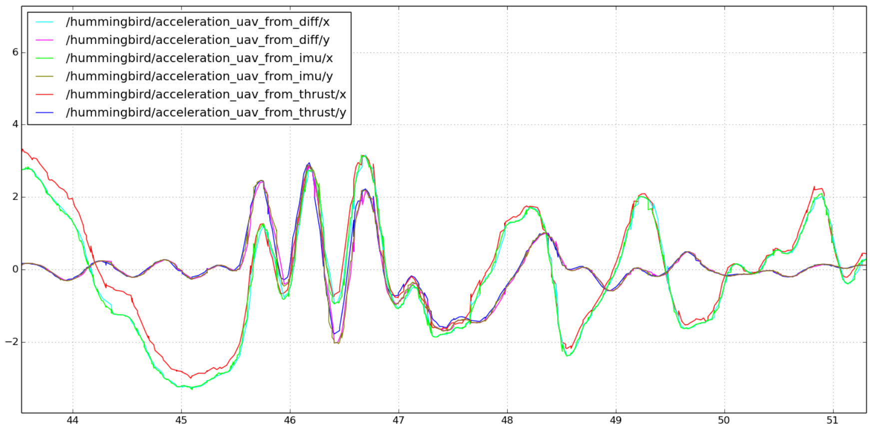

Figure 5.2: Comparison Acceleration Different Mass

Chapter 6

Nonlinear Model Predictive Control

6.1 Model

6.2 Cost Function

6.3 Acado Library

6.4 Learning

6.5 Results

Chapter 7

Experiments

Provide numerical results, plots, and timings. Interpret the data.

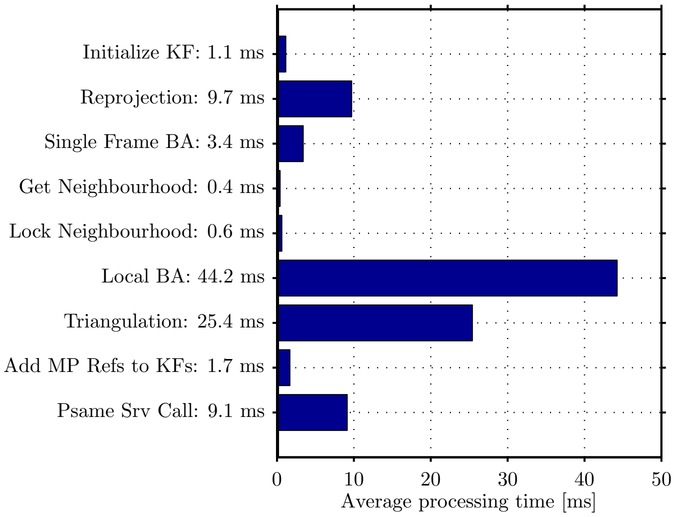

Figure 7.1: Example of a figure.

Chapter 8

Scientific Writing

This chapter gives you some tips on how to write scientifically. It should prevent you from making the most common mistakes people do and help you with creating a well written report.

8.1 General Style

- A report/paper is not a short-story. There is no build-up to a climax. The climax should be in the abstract. Even better, in the title.
- Hierarchical exposition, not linear: this goes in hand with the previous point. A hierarchical exposition means that you start with the core of your work (The main thing your project was about) and then go into details in following sections. Do not build up to the core of your work with too much background/preliminaries as it would be the case in a linear exposition.
- At the beginning of every chapter/major section, you should summarize what the content of the section will be. A person should get a good sense of the report by reading the first paragraph of each section.
- Express your thoughts succinctly. Avoid unnecessary words or phrases and be precise and specific.
- Definitions are useful if they are used often. Do not define something if it is only used once.
- Be generous with your references. Do not compare your results with others by pointing the deficiencies of their work; rather, state how your results are adding to the body of knowledge others have created.
- Notation is extremely important. Good notation facilitates understanding. You do not want the reader to mentally perform translations every time they see a symbol.

8.2 Important Stuff

- Use active verb tense whenever possible: instead of *An analysis of the signal noise is performed using a discrete Fourier transform.* write **We perform an analysis of the signal noise using a discrete Fourier transform.**
- Make short sentences with one statement. Long sentences with multiple statements are complicated and hard to understand. Write to be understood, not to impress!
- Be concrete/specific: instead of *We use a model to predict the state* write **We use a linear model of the attitude dynamics to predict the quadrotor's state at time** $t + \Delta t$.
- Be precise: instead of *We assume the model to be linear*, say **We design a linear model of the system dynamics**. (You assume the *system dynamics* to be linear and hence you create a linear model.)
- Be consistent: this basically applies to every level. Denote the same thing always with the same word, create figures with a similar style, etc.
- Do not make unsubstantiated statements. Do not use *It is common knowledge* or *Several researchers have shown.* Instead use constructs like **Recently, several researchers [?, ?] have shown**.

8.3 Small Things

- Do not use *don't*, *aren't*, etc., use **do not** and **are not**.
- Do not use words like *simply*, *highly*, *just*, *very*, etc.
- Use **because** instead of *due to the fact that*, **to** instead of *in order to*, etc.
- When referencing to figures, sections, etc., use capital letters: see **Figure** 1, see **Section** 2.
- Every figure must be referenced in the text.
- Use ~ to make spaces which LaTeX must not separate: Figure~`\ref{fig:bla}`. This avoids having the word and the number on different lines.
- Put punctuation marks after each formula as if they were text. Separate multiple consecutive formulas by commas and put a dot if you start a new sentence after the formula. For more details, see Section **??**.
- Use `\left(` and `\right)` when you have mathematical expressions that are higher than normal brackets, e.g., $\left(\frac{pV}{RT}\right)$ instead of $(\frac{pV}{RT})$.
- Avoid brackets. If something is important enough to be mentioned it does not need brackets; if not, it does not need to be mentioned at all.
- In English, after a colon (:) you continue with small letters.
- Use *we* to refer to yourself, i.e. **We** developed an algorithm to ...

- Do not use *ours*.
- Number all equations.
- Put details in an appendix.
- Avoid single-sentence paragraphs.

Chapter 9

LaTeX Tips and Tricks

In this chapter, we show some useful tips and tricks when working with LaTeX.

9.1 Using Git

We recommend you to use *Git* also for your LaTeX files such as this report. If you do so, we suggest to write every sentence in your TeX file on a new line. This will make it easier to keep track of changes since *Git* tracks them line by line. So if you change one sentence, *Git* will tell you that only that sentence has changed instead of the entire paragraph otherwise. Furthermore, if you are using the PDF viewer of *texmaker*, you can jump from the PDF directly to the sentence in the TeX file by clicking on it (instead of just jumping to the corresponding paragraph).

9.2 Headings

Your report can be structured using several different types of headings. Use the commands `\chapter`{.}, `\section`{.}, `\subsection`{.}, and `\subsubsection`{.}. Use the asterisk symbol `*` to suppress numbering of a certain heading if necessary, for example, `\section*`{.}.

9.3 References

References to literature are included using the command `\cite`{·}. For example [?, ?]. Your references must be entered in the file `bibliography.bib`. Making changes or adding new references in the bibliography file can be done manually or by using specialized software such as *JabRef* which is free of charge.

Cross-referencing within the text is easily done using `\label`{·} and `\ref`{·}. For example, this paragraph is part of Chapter **??**; more specifically on page **??**.

9.4 Writing Equations

The most common way to include equations is using the `equation` environment. Use `\eqref{·}` to reference an equation, e.g., (**??**).

Embed equations in the text. Thus you must use proper punctuation. You must introduce all symbols that you use. You should define these before you use them. However, they must be introduced in the same sentence at the latest.

Example 1

For n detections and m LEDs on the object, we will obtain N pose candidates,

$$N = 4\alpha \binom{n}{3} \frac{m!}{(m-3)!}, \tag{9.1}$$

where $\alpha \in \{1, 2\}$ is a magic factor.

Example 2

The transformation matrix in homogeneous coordinates, $\mathtt{T}$, is composed of the rotation matrix $\mathbf{R}$ and translation vector $\mathbf{p}$,

$$\mathtt{T} = \begin{bmatrix} \mathbf{R} & \mathbf{p} \\ 0 & 1 \end{bmatrix}, \qquad \text{with} \quad \mathbf{R} \in SO(3), \quad \mathbf{p} \in \mathbb{R}^3. \tag{9.2}$$

9.5 Including Graphics

The easiest way to include figures in your document is to use PDF figures if you use `pdflatex` to compile. Figure **??** was created with the use of the open-source program `ipe`.

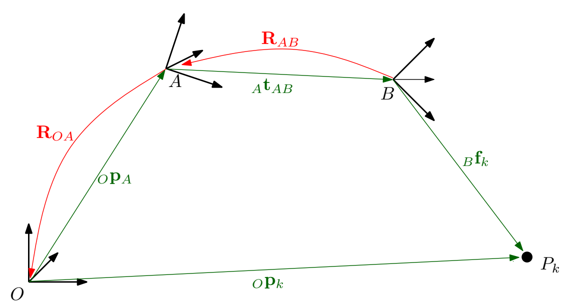

Figure 9.1: Example of a figure.

9.6 Including Matlab Figures

When including figures into your report you want them as a vector graphic such that you can zoom into the figure without getting blurry. Furthermore it is nice when the text in the figure gets substituted by LaTeX such that you have the same font and the same font size. Figure ?? shows an example of such an imported matlab figure. An easy way of achieving this is by using the `matlab2tikz` script. You can find a short example on how to use this script in the `matlab_figures` folder. The `create_figures.m` script creates a plot and then the tikz file which you can include in your document. For using tikz, you need to make use of the `pgfplots` package in your TeX document. More information on using `matlab2tikz` can be found on Matlab Central where you can also download the necessary files (`matlab2tikz.m`, `matlab2tikzInputParser.m`, `updater.m`).

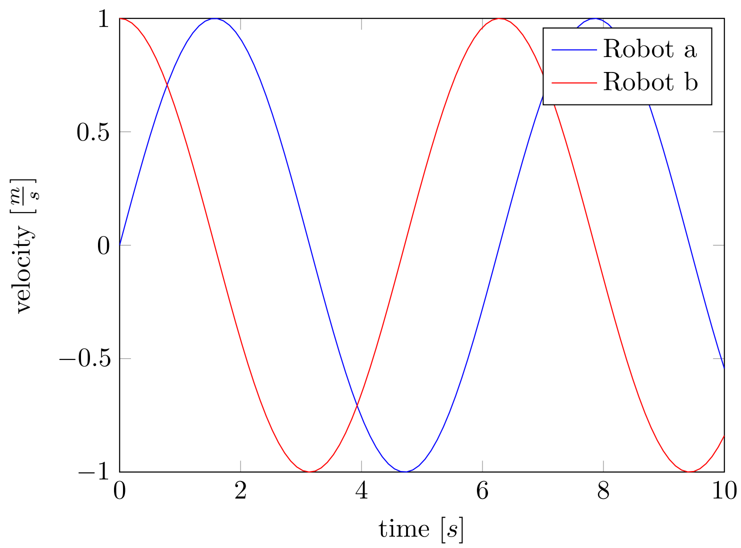

Figure 9.2: Example figure created with `matlab2tikz`.

An alternative which you might want to consider is `matlabfrag` and `mlf2pdf`. Especially when there are many data points in your figure you might run into problems when using tikz. Again, you can find a short example on how to use `mlf2pdf` in the `create_figures.m` scriptin in the `matlab_figures` folder. This script makes use of the two functions `matlabfrag.m` and `mlf2pdf.m` to create a PDF which you can then include into matlab. These two files can be downloaded here and here.

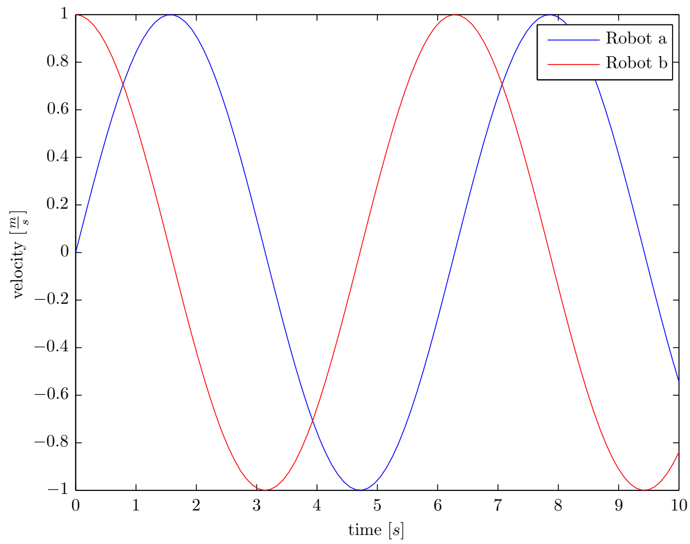

Figure 9.3: Example figure created with `mlf2pdf`.

9.7 Including Code in your Document

You may include samples from your Matlab code using the `lstlistings` environment, for example:

Listing 9.1: Matlab Example

```
% Evaluate y = 2x
for i = 1:length(x)

  y(i) = 2*x(i);

end
```

Listing 9.2: C++ Example

```
// sum all elements in a list
int sum=0;
for(list<int>::iterator it=mylist.begin(); it!=mylist.end(); ++it)
  sum += *it;
```

Chapter 10

RPG Notation Style

This chapter presents some conventions on notation that we use at the Robotics and Perception Group. Try to stick to those conventions since a unique style makes it easier to review the report.

10.1 Variable styles in LaTeX

Use lowercase and bold letters for vectors, e.g. $\mathbf{x}$, uppercase and bold letters for matrices, e.g. $\mathbf{R}$, and lowercase letters with normal weight for scalars, e.g. s.

10.2 Coordinate Systems and Rotations

We use the notation introduced by Prof. Glocker in the course "Mechanik 3" at ETHZ to express coordinate frames, rotations and vectors. Refer to Chapter 5 "Kinematik" in the lecture script for more details [1]. Figure **??** gives an overview of how coordinate transformations and vectors are specified. Observe that the coordinate system in which a vector is expressed is always written as index before the variable, e.g. ${}_B\mathbf{t}_{AB}$ is the vector from A to B expressed the coordinate system B. For the ease of reading, the index for the origin coordinate frame can be omitted: ${}_O\mathbf{t}_k := \mathbf{t}_k$.

[1] http://mitschriften.amiv.ethz.ch/main.php?page=3&scrid=1&pid=87&eid=1

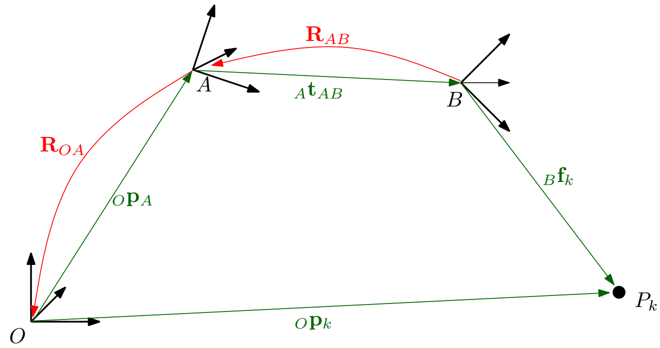

Figure 10.1: Notation overview.

A and B are two adjacent coordinate frames and O is the frame of origin. $\mathbf{R}_{AB}$ describes the coordinate transformation from frame B to frame A, thus it holds that

$$\begin{aligned} {}_O\mathbf{t}_k &= \mathbf{R}_{OB}\ {}_B\mathbf{f}_k, \\ \mathbf{R}_{OB} &= \mathbf{R}_{OA}\ \mathbf{R}_{AB}. \end{aligned}$$

10.3 Measured, estimated and target values

For controllers and estimators please specify the variables as follows in the report:

true value:	$\mathbf{x}$
estimated value:	$\hat{\mathbf{x}}$
measured value:	$\tilde{\mathbf{x}}$
desired value:	$\mathbf{x}_{\text{des}}$
error value:	$\mathbf{x}_{\text{e}}$
equilibrium value:	$\mathbf{x}^*$

Chapter 11

Discussion

Explain both the advantages and limitations of your approach.

11.1 Conclusion

Summarize your work and what came out of it.

11.2 Future Work

How would you extend the work? Can you propose another approach?

Appendix A

Something

In the appendix, you can provide some more data, a tutorial on how to run your code, a detailed proof, etc.

It is, however, not a requirement to have an appendix.

Title of work:

Autonomous landing on a moving platform

Thesis type and date:

Master Thesis, September 2016

Supervision:

First Supervisor Davide Falanga
Second Supervisor
Prof. Dr. Davide Scaramuzza

Student:

Name:	Alessio Zanchettin
E-mail:	zalessio@student.ethz.ch
Legi-Nr.:	97-906-739

Statement regarding plagiarism:

By signing this statement, I affirm that I have read the information notice on plagiarism, independently produced this paper, and adhered to the general practice of source citation in this subject-area.

Information notice on plagiarism:

`http://www.lehre.uzh.ch/plagiate/20110314_LK_Plagiarism.pdf`

Zurich, 16. 8. 2016: ____________________