;

1 Introduction

Hi everyone, this is Simon, Philipp, Benedikt and Annika and we present you our project: Real Time Control of a Quadcopter.

Imagine, you are really good at skiing, do great tricks and so on... and you want to show your friends. Wouldn't it be cool to have a quadcopter with you that follows you an films everything in an optimal way without you having to control it?
But, of course this isn't that easy. Because how should the Copter know the optimal route to follow you, when even you don't know exactly where you're going. And then, it might be windy weather and you cannot predict the direction or the strength of the wind.
So what is needed here, is calculation in real time, to adapt to the changing situations.

Now you have an idea what we wanted to achieve in our project, but how did we proceed?

!!!!!!!! Inhalts-Flussdiagramm!!!!!!!! (alles grau außer Model)

Essentially we could devided our Project in two blocks: The formulation of our model and the real time part.

Lets start with the "'model-part"'.

2 Model

2.1 Problem formulation

It is always good to know the basis at which you are working, so our very first step was to formulate our optimal control Problem theoretically.

We have an objective function J which we want to minimize over some controls$u$, and states x. Then, there are "'normal"' constraints $\tilde{h}$ which the states and controls have too fulfil. So far, it looks all like a normal optimization Problem, but additionally the states and controls are coupled via differential equations. These Differential equations represent the dynamics of the quadcopter.

So, a first step was to transform this into a normal optimization problem, which has only constraints of the form $h(x,u) = 0$.

But for this, we first had to determine the Differential equation.

2.2 Model

We start with our quadcopter. As it is a physical object here on earth, it is pulled down by the gravitational force.
To let the copter fly, it has its four rotors. The two opposite rotors turn in the same direction, the other two are turning the other way round. This ensures that the copter is not turning around the z-Axis the whole time.
Now, these four rotors create forces, that lift the copter up. Additionally there are external forces, like the wind.
The copter lies in the air in a stable way, if the external forces and the gravitational force is balanced by the forces of the rotors.

The second important part are the torques. They describe the rotation of the copter.
Our Copter can turn around the x-, the y- and the z-axis.
Again, there are external torques, and the torques have to balance each other.

Well, and with this, we have already derived the so called Newton-Euler-equations for the quadcopter. They can be written as the differential equation we were searching for.
!!!!!!!! Inhalts-Flussdiagramm!!!!!!!! (Model grün, Quaternionen blau)

2.3 Quaternions

To come from this differential equation to constraints of the form $h(x, u) = 0$ we have to integrate the differential equation. we did this numerically and normally that is not such a big deal, but we had a little problem here, called quaternions.

To make notation easyer, we worked with different coordinate systems. One inertial frame, from which one "observes" the motion of the skier and the coper, and bodyfixed frame, which sticks to the copter. For our calculations, we needed to rotate the vectors from one system to the other. And these rotations we described by quaternions.

What are quaternions? Essentially they are some kind of more complicated complex numbers. They have a real part, ind instead of one, now three imaginary parts. So it is a number $a + ib + jc + kd$ where a, b, c and d are in the reals.
As well as complex numbers quaternions can be written as a vector, now not in $\mathbb{R}^2$ but in $\mathbb{R}^4$ by taking only the coefficients of the real- and the imaginary parts.
And now it comes: If a quaternion has norm 1 it rebresents a rotation in 3-dimensional space. That means: all quaternions, that represent rotations lie on the $\mathcal{S}^3$, the 3-dimensional sphere.

So far, no problem. But the quaternions are part of the differential equations. Now if we solve them numerically we do something like this (Bild). We take the inclination and move along this line for the step size h. But what happens now, is that we leave the sphere. Bad thing! So we had to add a correction term, that pulls the integrated quaternion back to the sphere in every integration step.

Kurz was über DAEs???
Ok, now we have derived our model, so we go on, with the real time part.
!!!!!!!! Inhalts-Flussdiagramm!!!!!!!! (Model-Seite grün, realtime blau)