Adaptive Quaternion Control of a Quadrotor

2.152 Nonlinear Control - Final Project

Bernhard Paus Graesdal
Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

***Abstract*—In this project, a cascaded controller design based on quaternions for the full control of a quadrotor is tested and implemented in simulation. The attitude controller is augmented with an MRAC adaptive controller design, attempting to restore the nominal behaviour of the quadrotor in case of parameter uncertainties or angular disturbances. The controller design is very much based on the results in [1] and [2]. The ultimate goal of this project is to make the design implementable on hardware, hence physical parameters of an actual quadrotor are used, implementing the actual actuator constraints of the real brushless DC motors used on the quadrotor.**

I. Introduction

This text is structured as follows: The dynamic model used for the quadrotor is shown in III. The biggest part of this project is the synthesis of an adaptive attitude controller, and hence multiple sections are devoted to this controller design: the tracking dynamics are derived in IV, and the attitude controller is synthesized in V. The adaptive augmentation of the attitude controller requires a two times differentiable command signal, hence a trajectory generator is devised in VI. The position controller is designed in VII, completing the controller design. Finally, in VIII, the performance of the controller is implemented and tested in various flight tests in simulation.

II. Quaternion Algebra

This section will not deal with all the algebra regarding quaternions, for this the reader is encouraged to take a look at [1], [2] or [3], However, some key results which will later be important are mentioned here.

First, the rotation of a vector $\boldsymbol{v}$ from the body frame to the inertial frame can be described as

$$^{I}\boldsymbol{v} = \boldsymbol{q} \otimes {}^{B}\boldsymbol{v} \otimes \bar{\boldsymbol{q}} \tag{1}$$

where $\boldsymbol{q}$ is the unit quaternion describing the rotation from the body frame to the inertial frame, $\bar{\boldsymbol{q}}$ denotes its conjugate (which for the unit quaternions is equal to its inverse), and $\otimes$ denotes the standard quaternion product.

Notice that the quaternion logarithm of an *unit* quaternion $\boldsymbol{q}$ is given as:

$$\log(\boldsymbol{q}) := \frac{\alpha}{2}\begin{bmatrix} 0 \\ \boldsymbol{n} \end{bmatrix} \tag{2}$$

In practice, this mapping is useful to change any unit quaternion to its corresponding axis-angle representation, represented by the rotation angle α and the unitary rotation axis $\boldsymbol{n}$. For ease of notation, define the function mapping this relationship directly to a vector in $\mathbb{R}^3$:

$$\log_v \boldsymbol{q} := \frac{\alpha}{2}\boldsymbol{n} \tag{3}$$

Notice also that the identity rotation described by $\boldsymbol{q}_{id}$ satisfies $\log_v \pm\boldsymbol{q}_{id} = \boldsymbol{0}$

It is known that quaternions define a double coverage of SO(3), where $-\boldsymbol{q}$ and $\boldsymbol{q}$ represent the short and long way rotation depending on the sign of q_0. To make sure the shortest rotation is always used when controlling the quadrotor, define the $(\cdot)^+$ operator as:

$$\boldsymbol{q}^+ = \begin{cases} \boldsymbol{q}, & q_0 \geq 0 \\ -\boldsymbol{q}, & q_0 < 0 \end{cases}$$

which returns the short rotation of a quaternion $\boldsymbol{q}$, with an angle smaller than or equal to π.

When measuring rotation error, it can be argued that the shortest distance on a sphere is the most appropriate metric, and is thus the one used in this project. The shortest distance on a sphere is defined as the geodesic distance on SO(3), given by the norm of the logarithm:

$$\|\boldsymbol{q}\|_{SO(3)} := 2\langle \log_v(\boldsymbol{q}^+), \log_v(\boldsymbol{q}^+)\rangle^{\frac{1}{2}} \tag{4}$$

where $\langle\cdot\rangle$ denotes the inner product. Note that when $q_0 \geq 0$, this simplifies to the absolute value of the rotation angle α. Notice also the multiplication of 2 which arise due to the fact that the rotation corresponding to α requires both multiplication by the unit quaternion and its conjugate.

Finally, we will make use of the derivative of the squared geodesic distance on SO(3):

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\|\boldsymbol{q}\|^2_{SO(3)} &= 2\frac{d}{dt}\langle \log_v(\boldsymbol{q}^+), \log_v(\boldsymbol{q}^+)\rangle \\ &= {}^{B}\boldsymbol{\omega}_b^T \log_v(\boldsymbol{q}^+) \end{aligned} \tag{5}$$

where ${}^{B}\boldsymbol{\omega}_b$ corresponds to the angular velocity of the body relative to the inertial frame, and $\boldsymbol{q}$ is the unit quaternion describing the rotation from the body to the inertial frame. Here it has been used that $\frac{d}{dt}\log_v \boldsymbol{q} = \frac{d}{dt}\frac{\alpha}{2}\boldsymbol{n} = \frac{\dot{\alpha}}{2}\boldsymbol{n} =: {}^{B}\boldsymbol{\omega}_b$.

III. Quadrotor Model

The quadrotor dynamics are described through the use of two right-handed orthogonal coordinate frames. We define the inertial frame I as being the North-East-Down (NED) frame, and the body fixed body frame as the body frame B. The coordinate frames can be seen in Fig. 1.

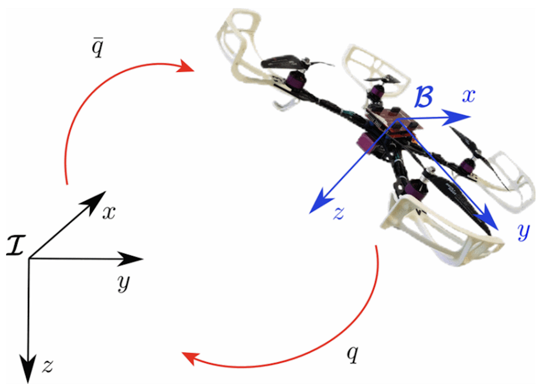

Fig. 1. Image showing the inertial frame I and the body frame B, both defined as NED-frames.

A. Attitude Dynamics

From using the Newton-Euler equations of motion, the angular acceleration of the quadrotor dynamics can be described by the following relationship (where all vectors are described in the body frame):

$$\boldsymbol{J}\,{}^{B}\dot{\boldsymbol{\omega}}_b = -{}^{B}\boldsymbol{\omega}_b \times \boldsymbol{J}\,{}^{B}\boldsymbol{\omega}_b + {}^{B}\boldsymbol{\tau}_{ext} \tag{6}$$

where $\boldsymbol{J} = \text{diag}(J_x, J_y, J_z)$ is the moment of inertia given in the body frame, ${}^{B}\boldsymbol{\omega}_b$ corresponds to the angular velocity of the quadrotor in the body frame, and ${}^{B}\boldsymbol{\tau}_{ext}$ corresponds to the sum of external torques applied to the quadrotor expressed in the body frame. The sum of torques acting on the quadrotor can be further split up into:

$$\boldsymbol{\tau}_{ext} = \boldsymbol{\tau}_c + \boldsymbol{\tau}_d \tag{7}$$

where $\boldsymbol{\tau}_c$ denotes the commanded torque calculated by the controller, and $\boldsymbol{\tau}_d$ is a torque disturbance acting on the quadrotor.

The kinematics of a rotating rigid body can be described by quaternions, and is in [3] given by the following relationship:

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_b \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 \\ {}^{I}\boldsymbol{\omega}_b \end{bmatrix} \otimes \boldsymbol{q} \tag{8}$$

where $\boldsymbol{q}$ is the unit quaternion describing the rotation from the inertial frame to the body frame, and ${}^{I}\boldsymbol{\omega}_b$ and ${}^{B}\boldsymbol{\omega}_b$ are the angular velocities of the quadrotor given in the inertial frame and the body frame, respectively. This gives the total state vector $\boldsymbol{x} = [\boldsymbol{q},\, \boldsymbol{\omega}_b]^T$.

B. Position Dynamics

Let $\boldsymbol{p} = [x, y, z]^T$ be the vector describing the quadrotors position in the inertial frame I. Then, using Newton-Euler equations of motion, the position dynamics of the quadrotor can be described in the inertial (NED) frame as follows:

$$\dot{\boldsymbol{x}}_{pos} := \frac{d}{dt}\begin{bmatrix} \boldsymbol{p} \\ \dot{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{q} \otimes \frac{{}^{B}\boldsymbol{F}_{th}}{m} \otimes \bar{\boldsymbol{q}} + \boldsymbol{g} \end{bmatrix} \tag{9}$$

where $\boldsymbol{g} = [0, 0, 9.81]^T$ denotes the gravity vector defined in the inertial frame, and ${}^{B}\boldsymbol{F}_{th} = [0, 0, F_c]^T$ denotes the total thrust vector of the quadrotor defined in the body frame. Notice the rotation of $\boldsymbol{F}_{th}$ from the body frame to the inertial frame.

C. Input of the Quadrotor

In this project, it is assumed that the input torques and the total input thrust of the quadrotor can be controller directly. That is, the input vector of the system is defined as:

$$\boldsymbol{u} = \begin{bmatrix} \tau_{x,c} \\ \tau_{y,c} \\ \tau_{z,c} \\ F_c \end{bmatrix} \tag{10}$$

For a discussion on the limitations of this, see X.

IV. Attitude Tracking Dynamics

A. Definition of command reference

The attitude control system has to make the quadrotor track a two times differentiable command signal. Consider now a third coordinate frame, the command frame C. Let the commanded attitude be described by $\boldsymbol{q}_c$, where $\boldsymbol{q}_c$ describes the rotation from the command frame C to the inertial frame I, and let the desired angular velocity be described by ${}^{C}\boldsymbol{\omega}_c$ (defined in the command frame C). Let the command reference follow the same dynamics as in (8):

$$\dot{\boldsymbol{q}}_c = \frac{1}{2}\boldsymbol{q}_c \otimes \begin{bmatrix} 0 \\ {}^{C}\boldsymbol{\omega}_c \end{bmatrix} \tag{11}$$

B. Definition of tracking error

Further, define the quaternion error as

$$\boldsymbol{q_e} = \bar{\boldsymbol{q}}_c \otimes \boldsymbol{q} \tag{12}$$

where $\boldsymbol{q}_e$ describes the rotation from the B-frame to the C-frame. Also, define the relative angular velocity (defined in the body frame) between the C-frame and the B-frame by

$$\begin{aligned} {}^{B}\boldsymbol{\omega}_{bc} &= {}^{B}\boldsymbol{\omega}_b - {}^{B}\boldsymbol{\omega}_c && (13) \\ &= {}^{B}\boldsymbol{\omega}_b - \bar{\boldsymbol{q}}_e \otimes {}^{C}\boldsymbol{\omega}_c \otimes \boldsymbol{q}_e && (14) \end{aligned}$$

note that ${}^{C}\boldsymbol{\omega}_c$ is defined in the command frame C, and hence needs to be rotated to the body frame B before subtraction can be applied.

C. Tracking error dynamics

The tracking error dynamics are derived for use in the control synthesis. First, differentiation of the error quaternion (12) gives the following dynamics for the attitude error:

$$\begin{aligned} \dot{\boldsymbol{q}}_e &= \dot{\bar{\boldsymbol{q}}}_c \otimes \boldsymbol{q} + \bar{\boldsymbol{q}}_c \otimes \dot{\boldsymbol{q}} \\ &= \frac{1}{2}\left\{ -\begin{bmatrix} 0 \\ {}^{C}\boldsymbol{\omega}_c \end{bmatrix} \otimes \boldsymbol{q}_c \otimes \boldsymbol{q} + \boldsymbol{q_c} \otimes \boldsymbol{q} \otimes \begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_b \end{bmatrix} \right\} \\ &= \frac{1}{2}\left\{ -\begin{bmatrix} 0 \\ {}^{C}\boldsymbol{\omega}_c \end{bmatrix} \otimes \boldsymbol{q}_e + \boldsymbol{q}_e \otimes \begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_b \end{bmatrix} \right\} \\ &= \frac{1}{2}\left\{ -\boldsymbol{q}_e \otimes \begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_c \end{bmatrix} + \boldsymbol{q}_e \otimes \begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_b \end{bmatrix} \right\} \\ &= \frac{1}{2}\boldsymbol{q}_e \otimes \left(\begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_b - {}^{B}\boldsymbol{\omega}_c \end{bmatrix} \right) \\ &= \frac{1}{2}\boldsymbol{q}_e \otimes \left(\begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_{bc} \end{bmatrix} \right) && (15) \end{aligned}$$

Further, differentiation of (13) in the body frame B gives (by using differentiation of vectors in coordinate frames as defined in [3]):

$$\frac{{}^{B}d}{dt}\boldsymbol{\omega}_{bc} = \frac{{}^{B}d}{dt}\boldsymbol{\omega}_{b} - \frac{{}^{C}d}{dt}\boldsymbol{\omega}_{c} - \boldsymbol{\omega}_{bc} \times \boldsymbol{\omega}_{c} \tag{16}$$

By using (6) and expressing the result in the body frame one obtains the dynamics of the angular velocity error:

$${}^{B}\dot{\boldsymbol{\omega}}_{bc} = \boldsymbol{J}^{-1}[-{}^{B}\boldsymbol{\omega}_{b} \times \boldsymbol{J}\,{}^{B}\boldsymbol{\omega}_{b} + \boldsymbol{\tau}_{ext}] - {}^{B}\dot{\boldsymbol{\omega}}_{c} - {}^{B}\boldsymbol{\omega}_{b} \times {}^{B}\boldsymbol{\omega}_{c} \tag{17}$$

note that also here ${}^{C}\boldsymbol{\omega}_{c}$ needs to first be rotated to the body frame B.

V. Attitude Controller Synthesis

In this section, a controller achieving asymptotic tracking of the command signal will be synthesized. This design is based on the results in [1]. First, a baseline controller is constructed, ensuring perfect tracking for the case when all the parameters are known. Then, an adaptive controller is designed, which will restore the nominal dynamics of the quadrotor in face of parameter uncertainties. The controller is thus divided in the following components:

$$\boldsymbol{\tau}_c = \boldsymbol{\tau}_b + \boldsymbol{\tau}_a \tag{18}$$

where $\boldsymbol{\tau}_b$ corresponds to the desired input torque computed by the baseline controller, and $\boldsymbol{\tau}_a$ corresponds to the input torque computed by the adaptive controller.

A. Baseline controller

Consider the following autonomous error dynamics (expressed in the body frame B):

$$\begin{aligned}\dot{\boldsymbol{q}}_e &= \frac{1}{2}\boldsymbol{q}_e \otimes \left(\begin{bmatrix} 0 \\ \boldsymbol{\omega}_{bc}\end{bmatrix}\right)\\ \dot{\boldsymbol{\omega}}_{bc} &= -k_q \log_v\left(\boldsymbol{q}_e^{+}\right) - k_\omega \boldsymbol{\omega}_{bc}\end{aligned} \tag{19}$$

with $k_\omega > 0$, $k_q > 0$. Consider also the following scalar, positive definite Lyapunov function:

$$V(\boldsymbol{q}_e, \boldsymbol{\omega}_{bc}) = \frac{1}{2}k_q \|\boldsymbol{q}_e\|^2_{SO(3)} + \frac{1}{2}\boldsymbol{\omega}_{bc}^T \boldsymbol{\omega}_{bc} \tag{20}$$

satisfying $V(\pm \boldsymbol{q}_{id}, 0) = 0$. By using (5), its derivative is obtained to be

$$\dot{V} = -k_\omega \boldsymbol{\omega}_{bc}^T \boldsymbol{\omega}_{bc} \leq 0 \tag{21}$$

Let the set R be defined by $R = \left\{\boldsymbol{q}_e, \boldsymbol{\omega}_{bc} \mid \dot{V} = 0\right\} = \{\boldsymbol{\omega}_{bc} = \boldsymbol{0}\}$. By looking at (19), the only invariant set within R is the set containing only the fixed points, $\{(\pm \boldsymbol{q}_{id}, 0)\}$. As the Lyapunov function is radially unbounded, it therefore follows from the Global Invariant Set Theorem [4] that these fixed points are globally asymptotically stable for the dynamics given in (19). Both of these fixed points corresponds to the same identity rotation, hence asymptotic tracking of the commanded attitude is achieved for the dynamics given in (19).

Now, for the controller synthesis, let us design a controller which, in the case where parameters are perfectly known, achieves the closed loop dynamics given in (17). By looking at (17), it is clear that feedback linearization can be used to obtain the desired error dynamics given in (19). Let therefore $\boldsymbol{\tau}_b$ be defined by:

$$\begin{aligned}\boldsymbol{\tau}_b = {}^{B}\boldsymbol{\omega}_b \times \boldsymbol{J}{}^{B}\boldsymbol{\omega}_b + \boldsymbol{J}({}^{B}\dot{\boldsymbol{\omega}}_c + {}^{B}\boldsymbol{\omega}_b \times {}^{B}\boldsymbol{\omega}_c\\ -k_q \log_v(\boldsymbol{q}_e^{+}) - k_w\,{}^{B}\boldsymbol{\omega}_{bc})\end{aligned} \tag{22}$$

which, in the nominal case, will lead to the closed dynamics in (19) and therefore ensure asymptotic tracking of the desired attitude.

B. Adaptive Controller

The goal of the adaptive attitude controller is to restore the nominal dynamics given in (19), in the case where $\boldsymbol{J} \neq \boldsymbol{J}_D$. The adaptive controller design very much resembles standard MRAC adaptive controller design as described in [5].

1) Defining the Adaptive Reference Model: Still assuming $\boldsymbol{J} = \boldsymbol{J}_D$ and inserting the baseline controller given by (22) into the angular velocity dynamics given in (6), and by applying the definition of $\boldsymbol{\omega}_{bc}$ in (13), one obtains:

$$\begin{aligned}{}^{B}\dot{\boldsymbol{\omega}}_b &= [-k_w \boldsymbol{i} - [{}^{b}\boldsymbol{\omega}_c]_\times]{}^{B}\boldsymbol{\omega}_b + {}^{B}\dot{\boldsymbol{\omega}}_c - k_q \log_v(\boldsymbol{q}_e^{+}) + k_w \boldsymbol{\omega}_c\\ &:= \boldsymbol{A}_m{}^{B}\boldsymbol{\omega}_b + \boldsymbol{r}_m\end{aligned} \tag{23}$$

with $\boldsymbol{A}_m = -(k_w \boldsymbol{I} + [{}^{B}\boldsymbol{\omega}_c]_\times)$ and $\boldsymbol{r}_m = {}^{B}\dot{\boldsymbol{\omega}}_c - k_q \log_v(\boldsymbol{q}_e^{+}) + k_w \boldsymbol{\omega}_c$. Noting that $\boldsymbol{A}_m^T + \boldsymbol{A}_m = -2k_w \boldsymbol{I}$, it is clear that $\boldsymbol{A}_m$ is stable since it satisfies the Lyapunov equation $\boldsymbol{A}_m^T \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_m = -\boldsymbol{Q} = -k_w \boldsymbol{I}$ for $\boldsymbol{P} = \boldsymbol{I}$. This result will also come in handy when later synthesizing the adaptive laws.

Further, it can be shown that using a closed-loop reference model leads to improved transient behaviour of the adaptive controller, and can be shown to lead to less oscillatory behaviour of the system [6]. Therefore, the adaptive controller will be designed to ensure tracking of the following time-varying closed-loop reference model:

$$\dot{\boldsymbol{\omega}}_m = \boldsymbol{A}_m \boldsymbol{\omega}_m + \boldsymbol{r}_m - k_e \boldsymbol{e} \tag{24}$$

with $\boldsymbol{e} := \boldsymbol{\omega}_m - \boldsymbol{\omega}_b$ and $k_e > 0$, and $\boldsymbol{A}_m$ and $\boldsymbol{r}_m$ defined as before. For controller synthesis, we will consider only the reference model where $k_e = 0$, as done in [1] and [6].

2) Defining the Adaptive Parameters: Now, for the purposes of the adaptive controller design we look to reformulate (6). First, define the unknown parameter matrix

$$\boldsymbol{\Theta} = \mathrm{diag}(\theta_1, \theta_2, \theta_3) \tag{25}$$

and let $\boldsymbol{\omega}_b = (\omega_1,\, \omega_2,\, \omega_3)^T$. This lets us simplify the following term:

$$-\boldsymbol{J}^{-1}[\boldsymbol{\omega}_b \times \boldsymbol{J}\boldsymbol{\omega}_b] = \begin{bmatrix} \frac{J_z - J_y}{J_x}\omega_z \omega_y \\ \frac{J_x - J_z}{J_y}\omega_x \omega_z \\ \frac{J_y - J_x}{J_z}\omega_x \omega_y \end{bmatrix} := \boldsymbol{\Theta}\boldsymbol{\phi}(\boldsymbol{\omega}_b) \tag{26}$$

where $\boldsymbol{\phi}(\boldsymbol{\omega}_b) = -(\omega_z\omega_y,\, \omega_x\omega_z,\, \omega_x\omega_y)^T$ is a known regressor vector. Further, by defining the control effectiveness matrix

$$\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \lambda_3) = \boldsymbol{J}^{-1} \succ 0 \tag{27}$$

(6) can be reformulated as:

$$\dot{\boldsymbol{\omega}}_b = \boldsymbol{\Theta}\boldsymbol{\phi}(\boldsymbol{\omega}_b) + \boldsymbol{\Lambda}\boldsymbol{\tau}_c + \boldsymbol{\tau}_d \tag{28}$$

with the subtle difference that $\boldsymbol{\tau}_d$ is now defined as a pure angular acceleration disturbance.

3) Adaptive Control Law: Letting $\boldsymbol{\tau}_c = \boldsymbol{\tau}_b + \boldsymbol{\tau}_a$, replacing the true parameters by their nominal values $\boldsymbol{J}_D$ in the baseline controller defined in (22), and by insertion into the newly reformulated dynamics (28), one obtains:

$$\begin{aligned}\dot{\boldsymbol{\omega}}_b &= \boldsymbol{\Theta}\boldsymbol{\phi} + \boldsymbol{\Lambda}(\boldsymbol{\tau}_b + \boldsymbol{\tau}_a) + \boldsymbol{\tau}_d + \boldsymbol{J}_D^{-1}\boldsymbol{\tau}_b - \boldsymbol{J}_D^{-1}\boldsymbol{\tau}_b \\ &= \boldsymbol{\Theta}\boldsymbol{\phi} + \boldsymbol{J}_D^{-1}[\boldsymbol{\omega}_b \times \boldsymbol{J}_D\boldsymbol{\omega}_b] - (k_w\boldsymbol{I} + [{}^B\boldsymbol{\omega}_c]_\times)\boldsymbol{\omega}_b \\ &\quad + {}^B\dot{\boldsymbol{\omega}}_c - k_q\log_v(\boldsymbol{q}_e^+) + k_w\boldsymbol{\omega}_c \\ &\quad + \boldsymbol{\Lambda}(\boldsymbol{\tau}_b + \boldsymbol{\tau}_a) - \boldsymbol{J}_D^{-1}\boldsymbol{\tau}_b + \boldsymbol{\tau}_d \\ &= (\boldsymbol{\Theta} - \boldsymbol{\Theta}_D)\boldsymbol{\phi} + \boldsymbol{A}_m\boldsymbol{\omega}_b + \boldsymbol{r}_m + \boldsymbol{\tau}_d \\ &\quad + \boldsymbol{\Lambda}(\boldsymbol{\tau}_a + (\boldsymbol{I} - \boldsymbol{\Lambda}^{-1}\boldsymbol{J}_D^{-1})\boldsymbol{\tau}_b)\end{aligned} \tag{29}$$

We now perform what is known as the *Algebraic Part* of MRAC controller design, and let the adaptive controller take the form:

$$\boldsymbol{\tau}_a = \hat{\boldsymbol{\Lambda}}^{-1}\left[-(\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}_D)\boldsymbol{\phi} - \hat{\boldsymbol{\tau}}_d\right] - (\boldsymbol{I} - \hat{\boldsymbol{\Lambda}}^{-1}\boldsymbol{J}_D^{-1})\boldsymbol{\tau}_b \tag{30}$$

In the case where all the adaptive parameter estimates are equal to the true parameters, this will make our closed loop system equal to the reference model defined in (23).

4) Adaptive Laws: Now, to the second step in MRAC controller design: The *Analytic Part*. The goal of this step is to recover an error model, such that adaptive laws for the adaptive parameters can be devised. We now insert the the adaptive control law defined in (30) back into (29), this time taking notice of the errors between the adaptive estimates and true parameters. Let $\tilde{\boldsymbol{\Theta}} := \hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}$, $\tilde{\boldsymbol{\Lambda}} := \hat{\boldsymbol{\Lambda}} - \boldsymbol{\Lambda}$ and $\tilde{\boldsymbol{\tau}}_d := \hat{\boldsymbol{\tau}}_d - \boldsymbol{\tau}_d$. Then, after some algebra, one obtains the true closed loop system:

$$\dot{\boldsymbol{\omega}} = \boldsymbol{A}_m\boldsymbol{\omega} + \boldsymbol{r}_m - \tilde{\boldsymbol{\Theta}\boldsymbol{\phi}} - \tilde{\boldsymbol{\tau}}_d - \tilde{\boldsymbol{\Lambda}}\boldsymbol{\tau}_c \tag{31}$$

Letting $\boldsymbol{e} := \boldsymbol{\omega}_m - \boldsymbol{\omega}_b$ yields our error dynamics:

$$\dot{\boldsymbol{e}} = \boldsymbol{A}_m\boldsymbol{e} + \tilde{\boldsymbol{\Theta}}\boldsymbol{\phi} + \tilde{\boldsymbol{\tau}}_d + \tilde{\boldsymbol{\Lambda}}\boldsymbol{\tau}_c \tag{32}$$

From this, the adaptive laws are chosen as:

$$\dot{\hat{\boldsymbol{\tau}}}_d = -\gamma_\tau\boldsymbol{P}\boldsymbol{e} \tag{33}$$

$$\dot{\hat{\boldsymbol{\Theta}}} = -\boldsymbol{\Gamma}_\Theta\boldsymbol{P}\boldsymbol{e}\boldsymbol{\phi}^T \tag{34}$$

$$\dot{\hat{\boldsymbol{\Lambda}}} = -\boldsymbol{\Gamma}_\Lambda\boldsymbol{P}\boldsymbol{e}\boldsymbol{\tau}_c^T \tag{35}$$

where $\gamma_\tau > 0$, $\boldsymbol{\Gamma}_\Theta = \boldsymbol{\Gamma}_\Theta^T \succ 0$ and $\boldsymbol{\Gamma}_\Lambda = \boldsymbol{\Gamma}_\Lambda^T \succ 0$ are adaptive gains to be chosen by the designer. Note that simply choosing $\boldsymbol{P} = \boldsymbol{I}$ suffices here, as shown in V-B1.

5) Ensuring tracking: Consider now the scalar positive definite Lyapunov function given by

$$V = \boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e} + \gamma_\tau^{-1}\tilde{\boldsymbol{\tau}}_d^T\tilde{\boldsymbol{\tau}}_d + \text{Tr}(\tilde{\boldsymbol{\Theta}}^T\boldsymbol{\Gamma}_\Theta^{-1}\tilde{\boldsymbol{\Theta}} + \tilde{\boldsymbol{\Lambda}}^T\boldsymbol{\Gamma}_\Lambda^{-1}\tilde{\boldsymbol{\Lambda}}) \tag{36}$$

Under consideration of the error model given by (32) and the adaptive laws in (35), its derivative is given by

$$\dot{V} = -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} = -k_\omega\boldsymbol{e}^T\boldsymbol{e} \leq 0 \tag{37}$$

From this it can be concluded that $\boldsymbol{e}$ and all the parameter errors are bounded, and from this it can be easily verified that $\ddot{V}$ is bounded. Therefore, it follows from Barbalat's Lemma [4] that $\boldsymbol{e} \to \boldsymbol{0}$, and tracking of the desired attitude, even in the face of parameter uncertainty, is guaranteed.

VI. Attitude Trajectory Generation

The attitude tracking dynamics in (17), as well as the controller synthesis in V assumes the command signal $\boldsymbol{q}_c$ to be continuous and two times differentiable. In order to guarantee this, a trajectory generator is synthesized.

The trajectory generator expects a reference signal $\boldsymbol{q}_{ref}$ to be continuous, but does not require the signal to be differentiable. The goal of the trajectory generator is to generate $\boldsymbol{q}_c$, $\boldsymbol{\omega}_c$ and $\dot{\boldsymbol{\omega}}_c$ from this signal, and then pass this signal on to the controller to achieve tracking.

Let $\boldsymbol{q}_{rc} := \boldsymbol{q}_r \otimes \boldsymbol{q}_c$, remember (11), and consider the following dynamics:

$$\dot{\boldsymbol{\omega}}_c = -2\omega_0^2\log_v(\boldsymbol{q}_{rc}^+) - 2D\omega_0\boldsymbol{\omega}_c \tag{38}$$

By using the Lyapunov function $V = \omega_0^2\|\boldsymbol{q}_{rc}\|_{SO(3)}^2 + \frac{1}{2}\boldsymbol{\omega}_c^T\boldsymbol{\omega}_c$, the system can be shown to be stable, in a similar manner as in section V. Notice however that here $\boldsymbol{q}_{rc}$ is time varying, hence LaSalle's Invariance Set theorem can not be applied to show asymptotic tracking [4]. It can however be applied to show asymptotic set point stabilization (i.e. in intervals where q_{ref} is actually constant).

Notice that by defining $\boldsymbol{\alpha} := \log_v(\boldsymbol{q}_{rc})$, one obtains $\dot{\boldsymbol{\alpha}} = \frac{1}{2}\boldsymbol{\omega}_c$, and hence (38) can be viewed as a linear, second order system:

$$\ddot{\boldsymbol{\alpha}} + 2D\omega_0\dot{\boldsymbol{\alpha}} + 2\omega_0^2\boldsymbol{\alpha} = 0 \tag{39}$$

hence w_0 and D can be interpreted as the bandwidth and the damping of the trajectory generator, respectively.

(38) together with (11), completes the trajectory generator.

VII. Position Controller Synthesis

The position controller is implemented based on the results in [2]. The goal of the position controller is to make the quadrotor track a two-times differentiable position trajectory $\boldsymbol{r}(t)_T = [x_T,\, y_T,\, z_T]^T$. In the following synthesis, it is assumed that the attitude controller is sufficiently fast, such that $\boldsymbol{q} \cong \boldsymbol{q}_c$. This will allow the position controller to be designed using feedback linearization.

Under the assumption $\boldsymbol{q} \cong \boldsymbol{q}_c$, let $\boldsymbol{u}_{pd} := \boldsymbol{q} \otimes \frac{{}^B\boldsymbol{F}_{th}}{m} \otimes \bar{\boldsymbol{q}}$, i.e. assume that the thrust vector in the inertial frame can be controlled directly. This lets us rewrite the position dynamics in (9) as

$$\dot{\boldsymbol{x}}_{pos} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{u}_{pd} + \boldsymbol{g} \end{bmatrix} \tag{40}$$

Further, choose $\boldsymbol{u}_{pd} = \boldsymbol{u}_{pos} - \boldsymbol{g}$. This yields the linear, time-invariant system:

$$\dot{\boldsymbol{x}}_{pos} = \begin{bmatrix} 0 & \boldsymbol{I} \\ 0 & 0 \end{bmatrix} \boldsymbol{x}_{pos} + \begin{bmatrix} 0 \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{u}_{pos} \tag{41}$$

It is well known that any linear time-invariant system can be stabilized by the linear feedback law

$$\boldsymbol{u}_{pos} = -\boldsymbol{K}_{pos}(\boldsymbol{x}_{pos} - \boldsymbol{x}_{pos,d}) \tag{42}$$

where $\boldsymbol{x}_{pos,d}$ is the desired position and velocity. $\boldsymbol{K}_{pos}$ can be chosen to be optimal by using LQR.

A. Adding a Feed-Forward

To ensure tracking of a two times differentiable trajectory, it is desirable to stabilize the *error dynamics* around the origin. This requries a feed-forward of the acceleration. By letting $\boldsymbol{K}_{pos} = [\boldsymbol{k}_1 \quad \boldsymbol{k}_2]$ and $\boldsymbol{e}_{pos} := \boldsymbol{x}_{pos} - \boldsymbol{x}_{pos,d}$, and by inserting (42) into (41), one obtains:

$$\ddot{\boldsymbol{p}} + \boldsymbol{k}_1 \boldsymbol{e}_{pos} + \boldsymbol{k}_2 \dot{\boldsymbol{e}}_{pos} = 0 \tag{43}$$

thus, by augmenting (42) with a feed-forward term $\boldsymbol{u}_{pos} = -\boldsymbol{K}_{pos}(\boldsymbol{x}_{pos} - \boldsymbol{x}_{pos,d}) + \ddot{\boldsymbol{p}}_d$, the origin becomes an asymptotically stable fixed point of the resulting linear position error dynamics:

$$\ddot{\boldsymbol{e}}_{pos} + \boldsymbol{k}_1 \boldsymbol{e}_{pos} + \boldsymbol{k}_2 \dot{\boldsymbol{e}}_{pos} = 0 \tag{44}$$

Hence, asymptotic tracking of the two times differentiable position trajectory is achieved.

VIII. Simulation Results

A. Implementation

The system and controller design is implemented in simulation. For the implementation, C++ together with the open-source C++ library Eigen [7] is used. For the simulation of the quadrotor model in (6), (8), and (9) are integrated with a simple Forward Euler scheme with a step-size of 0.0001.

The physical parameters used for the simulation are as follows:

$$m = 2.856 \quad \text{kg} \tag{45}$$

$$l_a = 0.2 \quad \text{m} \tag{46}$$

$$\boldsymbol{J} = \begin{bmatrix} 0.07 & 0 & 0 \\ 0 & 0.08 & 0 \\ 0 & 0 & 0.12 \end{bmatrix} \tag{47}$$

where l_a denotes the length from the origin to the propeller along each of the arms of the quadrotor.

The controller parameters are chosen as follows:

$$k_q = 60.0, \quad k_\omega = 10.0 \quad k_e = 25.0 \tag{48}$$

$$\boldsymbol{\Gamma}_\Theta = 2\boldsymbol{I}, \quad \boldsymbol{\Gamma}_\Lambda = 2\boldsymbol{I}, \quad \gamma_\tau = 350 \tag{49}$$

$$\boldsymbol{K}_{pos} = \begin{bmatrix} 3.16 & 0 & 0 & 2.71 & 0 & 0 \\ 0 & 3.16 & 0 & 0 & 2.71 & 0 \\ 0 & 0 & 3.16 & 0 & 0 & 2.71 \end{bmatrix} \tag{50}$$

To make the simulation more realistic, the adaptive controller is initialized with a poor estimate of $\boldsymbol{J}$. For each diagonal entry, the estimated value is between $180\% - 350\%$ of the true value, and the initial values of the adaptive parameters $\hat{\boldsymbol{\Lambda}}$ and $\hat{\boldsymbol{\Theta}}$ are calculated from these poorly estimated initial values.

B. Aggressive Attitude Tracking

For the first test, the quadrotor is tasked with tracking an aggressive square attitude reference. The resulting attitude tracking can be seen in Fig. 2. This experiment shows the difference between the performance of the baseline attitude controller and the adaptive controller, as the adaptive augmentation is not switched on until at $t = 20$. The baseline controller performs surprisingly well, despite the big estimation errors in the moment of inertia of the quadrotor. However, noticeable offsets still occur, and it is clear that due to the uncertainties, just using the baseline controller is not sufficient. When the adaptive controller is switched on, the nominal behaviour is mostly restored, and almost perfect tracking is achieved.

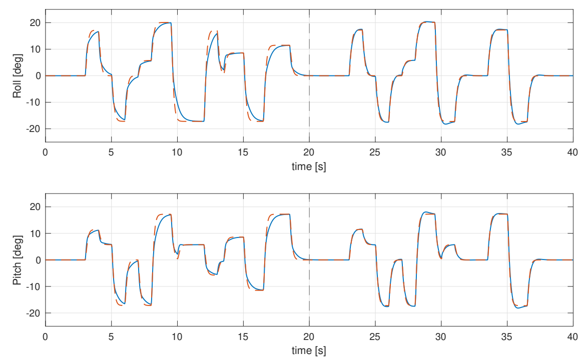

Fig. 2. Attitude tracking of the quadrotor. The dashed line marks the point where the adaptive controller is switched on

From the closeup in Fig. 3, it can be seen that the quadrotor reaches its desired attitude in approximately 0.5 seconds. Notice that from the same figure, one can see how the trajectory generator works by creating a two times differentiable command signal from the original square reference signal. By increasing the bandwidth of the trajectory generator, one can achieve a more aggressive command signal; however, to ensure feasible input values, this bandwidth is kept at a certain level.

From Fig 4, the applied input torques can be seen. The input torques actually reach the actuator limits at multiple points. This illustrates why a sufficiently small bandwidth for the trajectory generator is required to implement this design on a real system.

C. Aggressive Attitude Maneuvers with Unknown Payload

Next, the same attitude trajectory is to be tracked, this time by adding two unknown payloads to two of the quadrotors

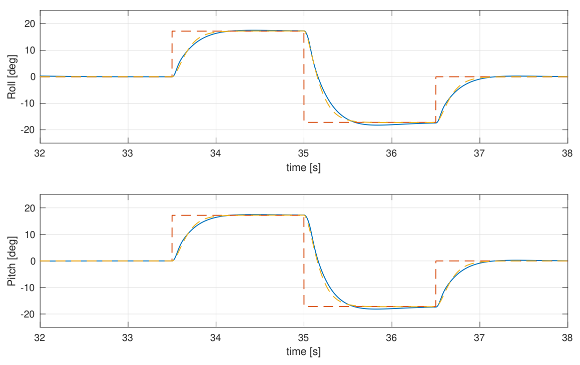

Fig. 3. Closeup of the attitude tracking. the squared dashed line is the initial reference trajectory, while the yellow dashed line shows the trajectory from the trajectory generator. set-points are reached in approximately 0.5 seconds.

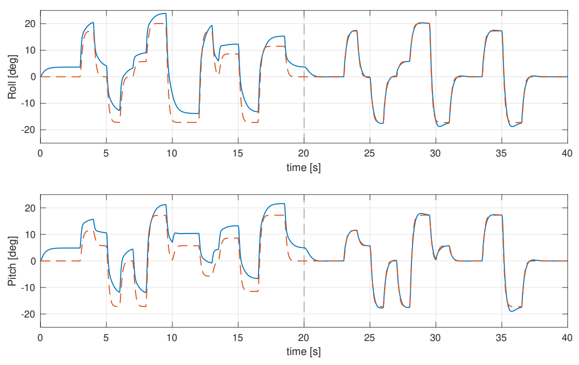

Fig. 5. Attitude tracking with an unknown payload attached to two of the arms.

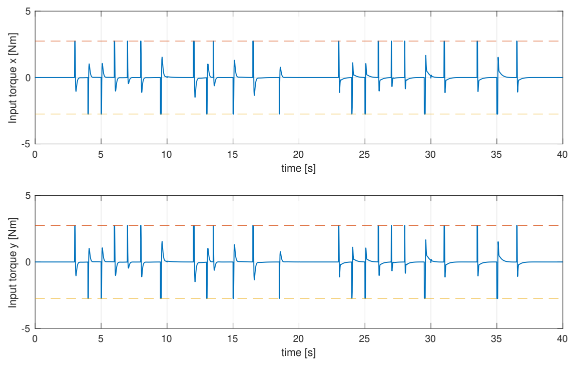

Fig. 4. Applied input torques during attitude tracking. Dashed lines show the actual actuator limits, where the input is saturated.

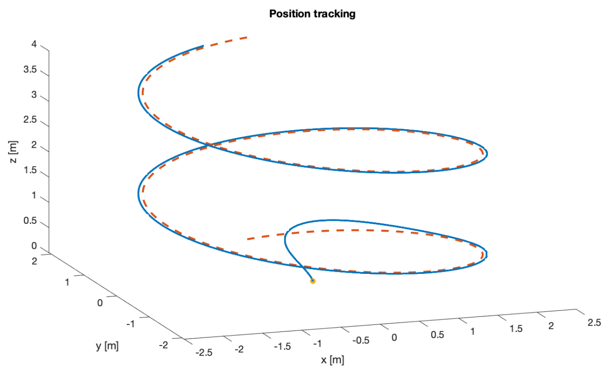

Fig. 6. Simple position tracking using only the baseline controller

arms. The payloads weigh 500 grams each, and are added to the arms exactly below the quadrotors motors, such that the payloads cause a torque disturbance. The result can be seen in Fig. 5. As before, the adaptive controller is only turned on at $t = 20$. This time, the baseline controller does not perform even close to satisfactory, as a large, constant deviation occurs. When the adaptive controller is switched on, this constant deviation is almost immediately compensated for, and the nominal behaviour is restored.

D. Simple Position Tracking

In the case of position tracking, a reference trajectory spiraling upwards is to be followed. This flight test will require much less aggressive attitude maneuvers than the previous task, and it is thus to be expected that the baseline controller will perform much better in this experiment. The resulting trajectory tracking without any adaptive augmentation can be seen in Fig. 6. The quadrotor does not actually start at the desired position trajectory, and in the beginning one can see the quadrotor rushing towards the desired trajectory. However, once at the trajectory, the trajectory tracking is almost perfect.

E. Position Tracking with Unknown Payload

In the last flight test, the same position trajectory is to be followed, but this time, the same unknown weight payload as in VIII-C is added to the quadrotor arms. The result can be seen in Fig. 7. As before, the adaptive controller is only switched on after some time. Before the adaptive controller is switched on, the unknown payload causes the baseline controller to exhibit a large offset in pitch and roll, and thus the quadrotor is not able to track its desired position trajectory. The individual roll and pitch values can be seen in Fig. 8. After switching on the adaptive controller, the quadrotor immediately compensates for the added weight, and restores the nominal behaviour of the quadrotor.

Finally, the adaptive parameter $\hat{\boldsymbol{\tau}}_d$ has been plotted in Fig. 9. This plot clearly shows how, once the adaptive augmentation is switched on, the adaptive parameters immediately move towards to the values which will stabilize the quadrotor around its nominal behaviour.

IX. Conclusion

In this project, a quaternion based, cascaded controller scheme has been devised for a quadrotor. For the attitude tracking, a MRAC adaptive controller augmentation has further been implemented, intended to restore the nominal behaviour

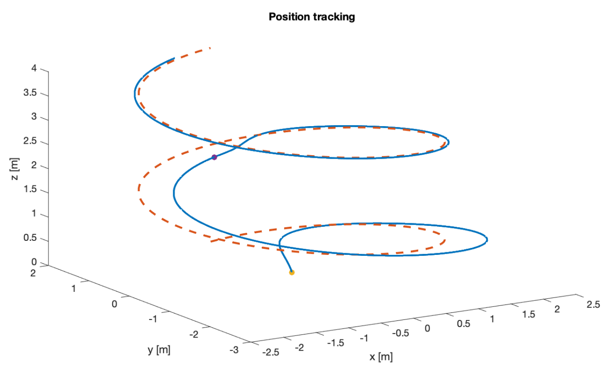

Fig. 7. Position tracking with an unknown payload added to two of the quadrotor arms. The adaptive controller is switched on at the point indicated by a red dot.

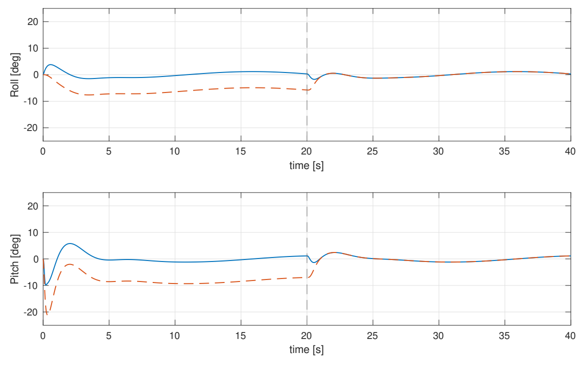

Fig. 8. The attitude of the quadrotor when tracking a position reference with an unknown payload added to two of the arms. The adaptive augmentation is switched on at $t = 20$.

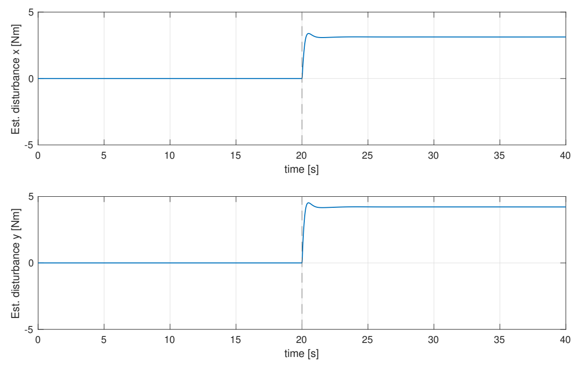

Fig. 9. The estimated angular disturbance when adding an unknown payload. The adaptive augmentation is switched on at $t = 20$.

of the system in case of angular disturbances and parameter uncertainties. The chosen design has been tuned and tested in simulation. As seen in the experiments performed in VIII, the controller design is able to deal with both parameter uncertainties and angular disturbances well, quickly restoring the nominal behaviour of the system.

X. Future Work

The next natural step towards implementing this on hardware would be to extend the quadrotor model to include the motor dynamics. In reality, one controls the individual propeller speeds, where each propeller has its own first-order dynamics. In addition to this, the relationship between the propeller speed and the generated thrust force is quadratic. In this experiment, it is assumed that input torques can be controlled directly, which is a common assumption to make in controller design. However, it still remains to test how much this simplification affects the performance on the real system.

To improve behaviour on the real system, robustness modifications should be implemented, either by using the projection operator and e-modification as in [1], or by any of the other measures presented in [8]. Some sort of prioritizing input saturation should also be implemented to ensure safe flight of the quadrotor, for example by using Pseudo Control Hedging as proposed in [1].

Finally, to complete the adaptive design, it could be desireable to incorporate an adaptive augmentation of the position controller to allow for an unknown mass of the quadrotor, to improve performance in the case of payload changes.

References

[1] D. Mihailescu-Stoica, R. Acuña, and J. Adamy, “High performance adaptive attitude control of a quadrotor,” 06 2019.

[2] J. Carino, H. Abaunza, and P. Castillo, “Quadrotor quaternion control,” *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 825–831, 2015.

[3] E. Coutsias and L. Romero, “The quaternions with applications to rigid body dynamics,” 01 1999.

[4] J.-J. E. Slotine and W. Li, “Applied nonlinear control,” 1991.

[5] K. Narendra and A. Annaswamy, *Stable Adaptive Systems*. Dover Books on Electrical Engineering, Dover Publications, 2012.

[6] T. Gibson, A. Annaswamy, and E. Lavretsky, “On adaptive control with closed-loop reference models: Transients, oscillations, and peaking,” *IEEE Access*, vol. 1, 04 2013.

[7] G. Guennebaud, B. Jacob, *et al.*, “Eigen v3.” http://eigen.tuxfamily.org, 2010.

[8] E. Lavretsky and K. A. Wise, “Robust and adaptive control: With aerospace applications,” 2012.