# Adaptive Quaternion Control of a Quadrotor

Bernhard Paus Graesdal

Massachusetts Institute of Technology

12th May 2020

# Introduction

## Scope of project

- Cascaded controller design for both attitude and position control
- Quaternion based controller to avoid singularities
- Attitude controller augmented with indirect MRAC design

## Goal

Implement project on hardware.

- Physical parameters of the an actual quadrotor are used
- Actuator constraints of real DC motors taken into account

## Sources

Design is very much based on [1] and [2].

# Table of Contents

Introduction

Quaternion Algebra

Quadrotor Model

Attitude Tracking

Attitude Controller

Position Controller

Experimental results

# Quaternion Algebra

Use unit quaternions to describe rotation.
Some key quaternion algebra results:

- Rotation by a quaternion:

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

- Logarithm of unit quaternion gives angle-axis representation:

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

The identity rotation described by $\boldsymbol{q}_{id}$ satisfies $\log_v \pm\boldsymbol{q}_{id} = \mathbf{0}$

# Quaternion Algebra

- Short rotation of a quaternion:

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

- Geodesic distance on SO(3) (shortest distance on a sphere):

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

  Intuition: When $\boldsymbol{q}$ is the short rotation, this is the absolute value of the rotation angle $\alpha$

- Its derivative:

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\|\boldsymbol{q}\|^2_{SO(3)} &= 2\frac{d}{dt}\langle \log_v(\boldsymbol{q}^{+}), \log_v(\boldsymbol{q}^{+})\rangle \\ &= {}^{B}\boldsymbol{\omega}_b^{T} \log_v(\boldsymbol{q}^{+}) \end{aligned} \tag{4}$$

# Quadrotor model

## Coordinate frames

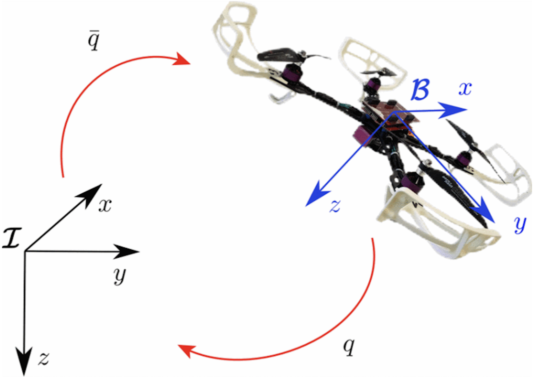


Figure 1: Inertial frame $I$ and the body frame $B$, both defined as NED-frames.

# Quadrotor model

## Angular acceleration (Newton-Euler):

$$\boldsymbol{J}^B\dot{\boldsymbol{\omega}}_b = -^B\boldsymbol{\omega}_b \times \boldsymbol{J}^B\boldsymbol{\omega}_b + {}^B\boldsymbol{\tau}_{ext} \tag{5}$$

- $\boldsymbol{\omega}_b$ - Angular velocity
- $\boldsymbol{J}$ - Inertia matrix
- $\boldsymbol{\tau}_{ext}$ - Sum of external torques applied to the body

$$\boldsymbol{\tau}_{ext} = \boldsymbol{\tau}_{control} + \boldsymbol{\tau}_{dist} \tag{6}$$

# Quadrotor model

Rotational dynamics (Quaternions for rigid bodies):

$$\dot{\boldsymbol{q}} = \frac{1}{2}\,\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ {}^{B}\boldsymbol{\omega}_b \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 \\ {}^{I}\boldsymbol{\omega}_b \end{bmatrix} \otimes \boldsymbol{q} \tag{7}$$

- $\boldsymbol{q}$ - Unit quaternion describing rotation from $B$ to $I$ frame

# Quadrotor model

## Position dynamics

$$\dot{\boldsymbol{x}}_{pos} := \frac{d}{dt}\begin{bmatrix}\boldsymbol{p} \\ \dot{\boldsymbol{p}}\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{p}} \\ \boldsymbol{q} \otimes \frac{{}^{B}\boldsymbol{F}_{th}}{m} \otimes \bar{\boldsymbol{q}} + \boldsymbol{g}\end{bmatrix} \tag{8}$$

- $\boldsymbol{q}$ - Unit quaternion describing rotation from $B$ to $I$ frame
- $\boldsymbol{F}_{th}$ - Total thrust vector in body frame
- $m$ - Quadrotor mass
- $\boldsymbol{g}$ - Gravity vector

# Attitude tracking

## Command attitude

$$\dot{\boldsymbol{q}}_c = \frac{1}{2}\,\boldsymbol{q}_c \otimes \begin{bmatrix} 0 \\ {}^{C}\boldsymbol{\omega}_c \end{bmatrix} \tag{9}$$

Let $\boldsymbol{\omega}_c$ be defined in the command frame $C$ defined by $\boldsymbol{q}_c$.

## Error signal

$$\boldsymbol{q}_{\boldsymbol{e}} = \bar{\boldsymbol{q}}_c \otimes \boldsymbol{q} \tag{10}$$

$${}^{B}\boldsymbol{\omega}_{bc} = {}^{B}\boldsymbol{\omega}_b - {}^{B}\boldsymbol{\omega}_c \tag{11}$$

- $\boldsymbol{q}_e$ - Unit quaternion describing rotation from $B$ to $C$ frame
- $\boldsymbol{\omega}_{bc}$ - Angular velocity of $B$ frame wrt. $C$ frame

# Attitude Tracking

The controller design will stabilize the *error dynamics*.

## Rotational error dynamics

$$\dot{\boldsymbol{q}}_e = \frac{1}{2}\boldsymbol{q}_e \otimes \left( \begin{bmatrix} 0 \\ {}^B\boldsymbol{\omega}_{bc} \end{bmatrix} \right) \tag{12}$$

## Angular velocity error dynamics

$${}^B\dot{\boldsymbol{\omega}}_{bc} = \boldsymbol{J}^{-1}[-{}^B\boldsymbol{\omega}_b \times \boldsymbol{J}\,{}^B\boldsymbol{\omega}_b + \boldsymbol{\tau}_{ext}] - {}^B\dot{\boldsymbol{\omega}}_c - {}^B\boldsymbol{\omega}_b \times {}^B\boldsymbol{\omega}_c \tag{13}$$

# Attitude controller design

## Desired closed-loop error dynamics:

$$\dot{\boldsymbol{q}}_e = \frac{1}{2}\boldsymbol{q}_e \otimes \left(\begin{bmatrix} 0 \\ \boldsymbol{\omega}_{bc} \end{bmatrix}\right) \tag{14}$$

$$\dot{\boldsymbol{\omega}}_{bc} = -k_q \log_v\left(\boldsymbol{q}_e^+\right) - k_\omega \boldsymbol{\omega}_{bc} \tag{15}$$

## Stability and tracking

$$V = \frac{1}{2}k_q\|\boldsymbol{q}_e\|^2_{SO(3)} + \frac{1}{2}\boldsymbol{\omega}_{bc}^T\boldsymbol{\omega}_{bc} \implies \dot{V} = -k_\omega \boldsymbol{\omega}_{bc}^T\boldsymbol{\omega}_{bc} \leq 0 \tag{16}$$

*Global Invariant set theorem:* The fixed point $(\boldsymbol{q}_e, \boldsymbol{\omega}_{bc}) = (\pm\boldsymbol{q}_{id}, \mathbf{0})$ is asymptotically stable [3]. $\rightarrow$ tracking is achieved!

# Attitude controller design

## Desired closed-loop error dynamics:

$$\dot{\boldsymbol{q}}_e = \frac{1}{2}\boldsymbol{q}_e \otimes \left( \begin{bmatrix} 0 \\ \boldsymbol{\omega}_{bc} \end{bmatrix} \right) \tag{17}$$

$$\dot{\boldsymbol{\omega}}_{bc} = -k_q \log_v\left(\boldsymbol{q}_e^{+}\right) - k_\omega \boldsymbol{\omega}_{bc} \tag{18}$$

## Two-fold controller design

$$\boldsymbol{\tau}_c = \boldsymbol{\tau}_b + \boldsymbol{\tau}_a \tag{19}$$

- Baseline controller: achieve desired closed-loop dynamics in the nominal case
- Adaptive controller: restore nominal behaviour in presence of uncertainties

# Attitude controller design - Baseline controller

## Baseline controller

*Assume all parameters perfectly known:* Use feedback linearization to achieve desired closed loop error dynamics:

$$\begin{aligned}\boldsymbol{\tau}_b = {}^B\boldsymbol{\omega}_b \times \boldsymbol{J}\,{}^B\boldsymbol{\omega}_b + \boldsymbol{J}({}^B\dot{\boldsymbol{\omega}}_c + {}^B\boldsymbol{\omega}_b \times {}^B\boldsymbol{\omega}_c \\ -k_q \log_v(\boldsymbol{q}_e^+) - k_w\,{}^B\boldsymbol{\omega}_{bc})\end{aligned} \tag{20}$$

## Resulting error dynamics

$${}^B\dot{\boldsymbol{\omega}}_{bc} = \boldsymbol{J}^{-1}[-{}^B\boldsymbol{\omega}_b \times \boldsymbol{J}\,{}^B\boldsymbol{\omega}_b + \boldsymbol{\tau}_b] - {}^B\dot{\boldsymbol{\omega}}_c - {}^B\boldsymbol{\omega}_b \times {}^B\boldsymbol{\omega}_c \tag{21}$$

$$\implies \dot{\boldsymbol{\omega}}_{bc} = -k_q \log_v(\boldsymbol{q}_e^+) - k_\omega \boldsymbol{\omega}_{bc} \tag{22}$$

Works perfectly in the nominal case!

# Attitude controller design - Adaptive controller

### Purpose of adaptive controller

Restore nominal behaviour in face of uncertainties and disturbances.

### Design approach

1. Find linear reference model which describes nominal case
2. Define adaptive parameters
3. Find adaptive control law
4. Choose adaptive laws to restore reference model

# Attitude controller design - Adaptive controller

## 1. Find reference model to describe nominal case

Insert baseline controller into *real* dynamics (not error dynamics):

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

- $\boldsymbol{A}_m = -(k_w \boldsymbol{I} + [{}^B\boldsymbol{\omega}_c]_\times)$. Always stable: $\boldsymbol{A}_m^T + \boldsymbol{A}_m = -k_w \boldsymbol{I}$
- $\boldsymbol{r}_m = {}^B\dot{\boldsymbol{\omega}}_c - k_q \log_v(\boldsymbol{q}_e^+) + k_w \boldsymbol{\omega}_c$.

# Attitude controller design - Adaptive controller

## Use closed-loop reference model

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

- Improves transient behaviour of the adaptive controller [4]. (largely reduces oscillations)

# Attitude controller design - Adaptive controller

## 2. Define adaptive parameters

$$-\boldsymbol{J}^{-1}[\boldsymbol{\omega}_b \times \boldsymbol{J}\boldsymbol{\omega}_b] = \begin{bmatrix} \frac{J_z - J_y}{J_x}\omega_z\omega_y \\ \frac{J_x - J_z}{J_y}\omega_x\omega_z \\ \frac{J_y - J_x}{J_z}\omega_x\omega_y \end{bmatrix} := \boldsymbol{\Theta}\phi(\boldsymbol{\omega}_b) \tag{25}$$

$$\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3) =: \boldsymbol{J}^{-1} \succ 0 \tag{26}$$

New dynamics:

$$\boldsymbol{J}^B\dot{\boldsymbol{\omega}}_b = -{}^B\boldsymbol{\omega}_b \times \boldsymbol{J}^B\boldsymbol{\omega}_b + {}^B\boldsymbol{\tau}_{ext} \tag{27}$$

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

- $\boldsymbol{\Theta}$, $\boldsymbol{\Lambda}$ - Unknown parameters
- $\phi(\boldsymbol{\omega}_b)$ - Known regressor vector

# Attitude controller design - Adaptive controller

## 3. Find adaptive control law

Letting the parameters deviate from their nominal values, and reformulating dynamics in terms of reference model, yields:

$$\begin{aligned}\dot{\boldsymbol{\omega}}_b = (\boldsymbol{\Theta} - \boldsymbol{\Theta}_D)\phi + \boldsymbol{A}_m\boldsymbol{\omega}_b + \boldsymbol{r}_m + \boldsymbol{\tau}_d \\ + \boldsymbol{\Lambda}(\boldsymbol{\tau}_a + (\boldsymbol{I} - \boldsymbol{\Lambda}^{-1}\boldsymbol{J}_D^{-1})\boldsymbol{\tau}_b)\end{aligned} \tag{29}$$

Choosing the adaptive controller:

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

yields the reference model when parameters equal nominal values (denoted by $_D$).

# Attitude controller design - Adaptive controller

## 4. Choose adaptive laws to restore reference model

*Error model:* Letting $\boldsymbol{e} := \boldsymbol{\omega}_m - \boldsymbol{\omega}_b$ yields:

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

*Adaptive laws:*

$$\dot{\hat{\boldsymbol{\tau}}}_d = -\gamma_\tau \boldsymbol{P} \boldsymbol{e} \tag{32}$$

$$\dot{\hat{\boldsymbol{\Theta}}} = -\boldsymbol{\Gamma}_\Theta \boldsymbol{P} \boldsymbol{e} \phi^T \tag{33}$$

$$\dot{\hat{\boldsymbol{\Lambda}}} = -\boldsymbol{\Gamma}_\Lambda \boldsymbol{P} \boldsymbol{e} \boldsymbol{\tau}_c^T \tag{34}$$

# Attitude controller design - Adaptive controller

## Stability and tracking

$$V = \boldsymbol{e}^T \boldsymbol{P} \boldsymbol{e} + \gamma_\tau^{-1} \tilde{\boldsymbol{\tau}}_d^T \tilde{\boldsymbol{\tau}}_d + \mathrm{Tr}(\tilde{\boldsymbol{\Theta}}^T \boldsymbol{\Gamma}_\Theta^{-1} \tilde{\boldsymbol{\Theta}} + \tilde{\boldsymbol{\Lambda}}^T \boldsymbol{\Gamma}_\Lambda^{-1} \tilde{\boldsymbol{\Lambda}}) \tag{35}$$

$$\implies \dot{V} = -\boldsymbol{e}^T \boldsymbol{Q} \boldsymbol{e} = -k_\omega \boldsymbol{e}^T \boldsymbol{e} \leq 0 \tag{36}$$

*Barbalats Lemma:* Gives $\boldsymbol{e} \to \mathbf{0}$. Tracking is ensured!

# Position controller

## Position dynamics

$$\dot{\boldsymbol{x}}_{pos} := \frac{d}{dt}\begin{bmatrix}\boldsymbol{p}\\ \dot{\boldsymbol{p}}\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{p}}\\ \boldsymbol{q}\otimes\frac{{}^{B}\boldsymbol{F}_{th}}{m}\otimes\bar{\boldsymbol{q}}+\boldsymbol{g}\end{bmatrix} := \begin{bmatrix}\dot{\boldsymbol{p}}\\ \boldsymbol{u}_{pd}+\boldsymbol{g}\end{bmatrix} \tag{37}$$

## Controller

- Assume attitude controller sufficiently fast: $\boldsymbol{q} \cong \boldsymbol{q}_c$
- 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).
- Use feedback linearization

# Position controller

## Controller

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

$$\dot{\boldsymbol{x}}_{pos} = \begin{bmatrix} 0 & \boldsymbol{I} \\ 0 & 0 \end{bmatrix} \boldsymbol{x}_{pos} + \begin{bmatrix} 0 \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{u}_{pos} \tag{38}$$

*Linear control law:*

$$\boldsymbol{u}_{pos} = -\boldsymbol{K}_{pos}(\boldsymbol{x}_{pos} - \boldsymbol{x}_{pos,d}) + \ddot{\boldsymbol{p}}_d \tag{39}$$

$$\implies \ddot{\boldsymbol{e}}_{pos} + \boldsymbol{k}_1 \boldsymbol{e}_{pos} + \boldsymbol{k}_2 \dot{\boldsymbol{e}}_{pos} = 0 \tag{40}$$

- $\boldsymbol{x}_{pos,d}$ is the desired position and velocity.
- $\ddot{\boldsymbol{p}}_d$ is the desired acceleration.
- Choose $\boldsymbol{K}_{pos}$ with LQR.

# Experimantal Results

## Implementation

- Implemented in C++ with the open-source library Eigen [5].
- Forward Euler with a step-size of 0.0001 seconds.

## Physical parameters

$$m = 2.856 \quad \text{kg} \tag{41}$$

$$l_a = 0.2 \quad \text{m} \tag{42}$$

$$\boldsymbol{J} = \begin{bmatrix} 0.07 & 0 & 0 \\ 0 & 0.08 & 0 \\ 0 & 0 & 0.12 \end{bmatrix} \tag{43}$$

# Experimantal Results

## Controller parameters

$$k_q = 60.0, \quad k_\omega = 10.0 \quad k_e = 25.0 \tag{44}$$

$$\mathbf{\Gamma}_\Theta = 2\boldsymbol{I}, \quad \mathbf{\Gamma}_\Lambda = 2\boldsymbol{I}, \quad \gamma_\tau = 350 \tag{45}$$

$$\boldsymbol{K}_{pos} = \begin{bmatrix} 3.16 & 0 & 0 & 2.71 & 0 & 0 \\ 0 & 3.16 & 0 & 0 & 2.71 & 0 \\ 0 & 0 & 3.16 & 0 & 0 & 2.71 \end{bmatrix} \tag{46}$$

## Initialization of adaptive parameters

- *Poor initial estimates:* Between 180% – 350% of the true value.
- The initial adaptive parameter values are calculated from this.

# Flight test 1 - Aggressive attitude maneuvers

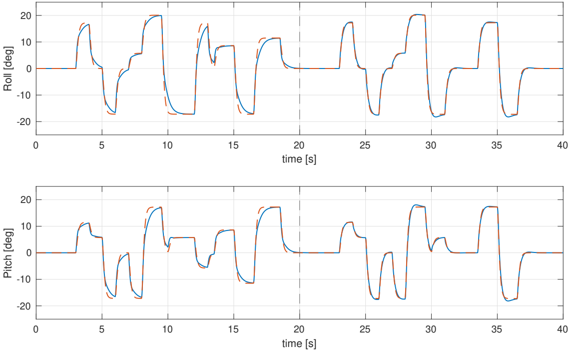


Figure 2: Attitude tracking of the quadrotor. The dashed line marks the point where the adaptive controller is switched on

# Flight test 1 - Actuator inputs

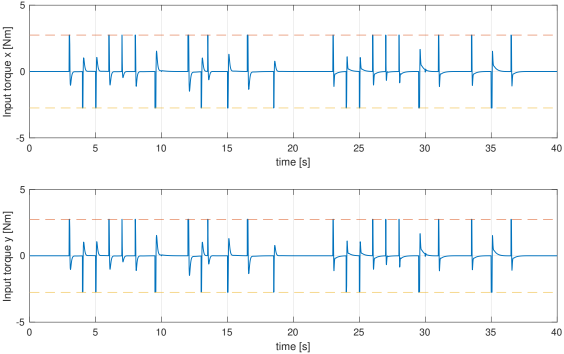


Figure 3: Applied input torques during attitude tracking. Dashed lines show the actual actuator limits, where the input is saturated.

# Flight test 2 - Aggressive attitude maneuvers w/ unknown payload

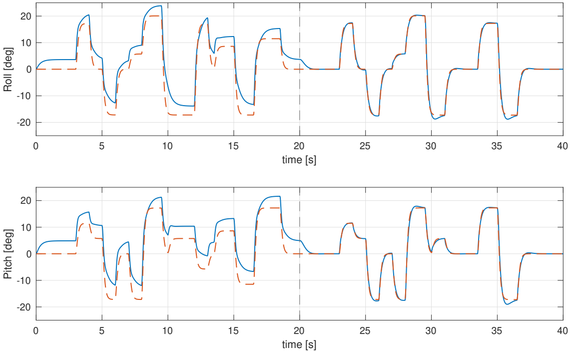


Figure 4: Attitude tracking with an unknown payload attached to two of the arms.

# Flight test 3 - Position trajectory tracking

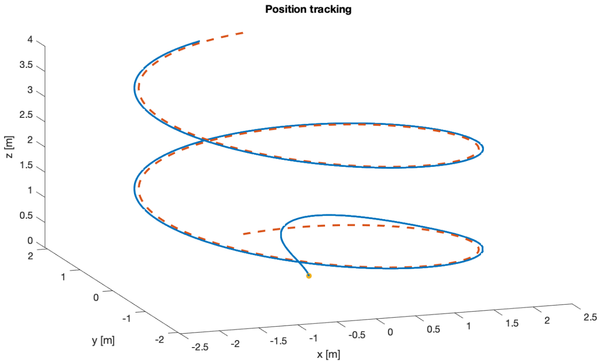


Figure 5: Simple position tracking using only the baseline controller

# Flight test 4 - Position trajectory tracking w/ unknown payload

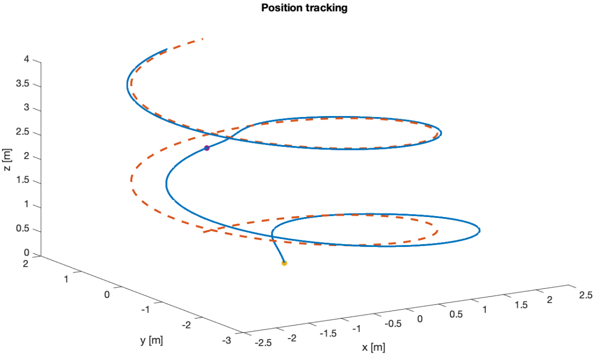


Figure 6: Position tracking with an unknown payload added to two of the quadrotor arms. The adaptive controller is switched on at the point indicated by a red dot.

# Flight test 4 - Position trajectory tracking w/ unknown payload

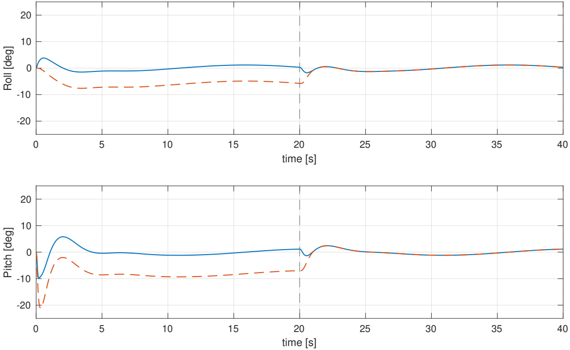


Figure 7: The attitude of the quadrotor when tracking a position reference with an unknown payload added to two of the arms. The adaptive augmentation is switched on at $t = 20$.

# Future work

- Simulate motor dynamics
- Robustness modifications to adaptive controller:
  - Projection operator
  - e-modification
  - ++ [6]
- Input prioritization to deal with saturation [1].
- Adaptive mass parameter in position controller

# Bibliography


- D. Mihailescu-Stoica, R. Acuña, and J. Adamy, "High performance adaptive attitude control of a quadrotor," 06 2019.
- J. Carino, H. Abaunza, and P. Castillo, "Quadrotor quaternion control," *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 825–831, 2015.
- J.-J. E. Slotine and W. Li, "Applied nonlinear control," 1991.
- 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.
- G. Guennebaud, B. Jacob, *et al.*, "Eigen v3." http://eigen.tuxfamily.org, 2010.
- E. Lavretsky and K. A. Wise, "Robust and adaptive control: With aerospace applications," 2012.