Quadrokopter

Echtzeitsysteme 2

Matthias Niessner, Christian Liebscher

July 15, 2008 Erlangen

Was ist der Quadrokopter ?

- Ein autonomes Fluggerät
- Vier an den Eckpunkten eines Vierecks befestigte Rotoren
- Steuerung und Kommunikation durch Funk
- Bekannt als unbemannte Aufklärungs- und Spionagedrohnen

Der Quadrokopter

Anwendungsmöglichkeiten

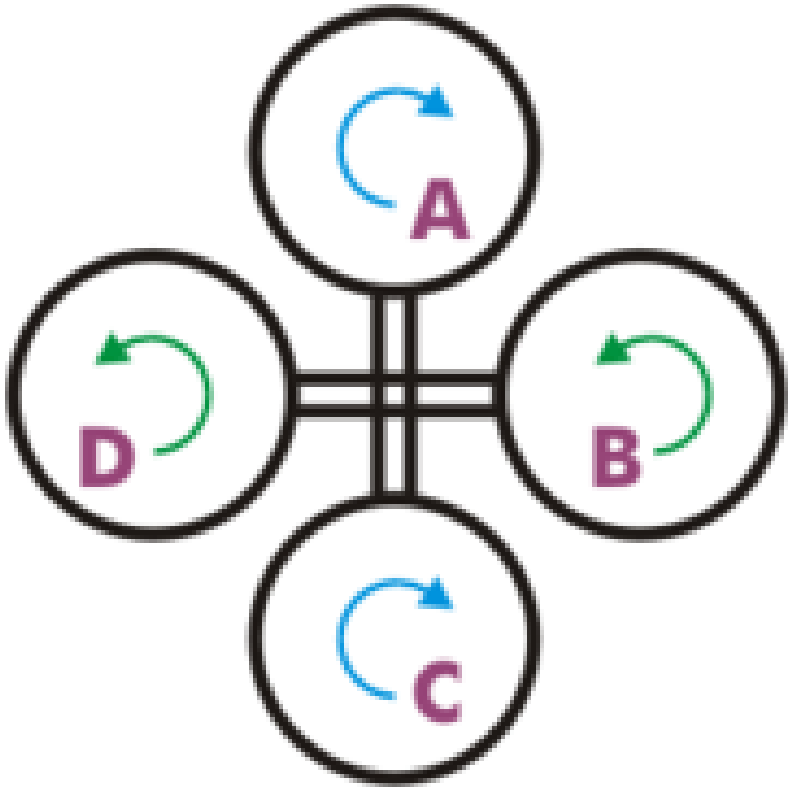
Wozu?

- Keine Anwendung in der bemannten Luftfahrt
- Anwendung als unbemanntes Flugobjekt
- Beliebt im Modellbau
- Einsatz als Spionagedrohnen

→ *vorallem durch letzteres wurde er populär*

Praxisbeispiel

Welche Möglichkeiten werden geboten ?



- Die 4 Rotoren erlauben Bewegungen in alle Richtungen

Hardwarebasis

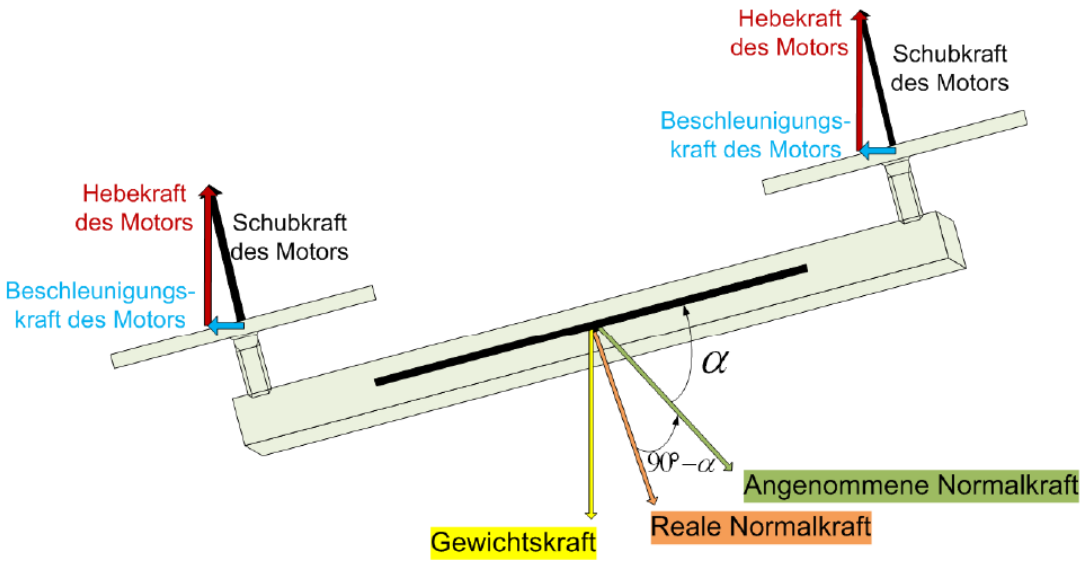
- 1x Tricore TC1796 Board (Hightec)
- 4x Brushless motor Roxxy BL-Outrunner 2827-34 (generating max. 2.4kg thrust)
- 4x Brushless motor controller YGE18
- 1x ADXL213 three axis accelerometer
- 2x IDG300 dual axis gyroscope

Problemstellung...

- Stabile Lagekontrolle des Quadrokopters
- Kontrollierter Flugablauf
- Interaktion mit dem Anwender - Steuerbefehle und Kommunikation

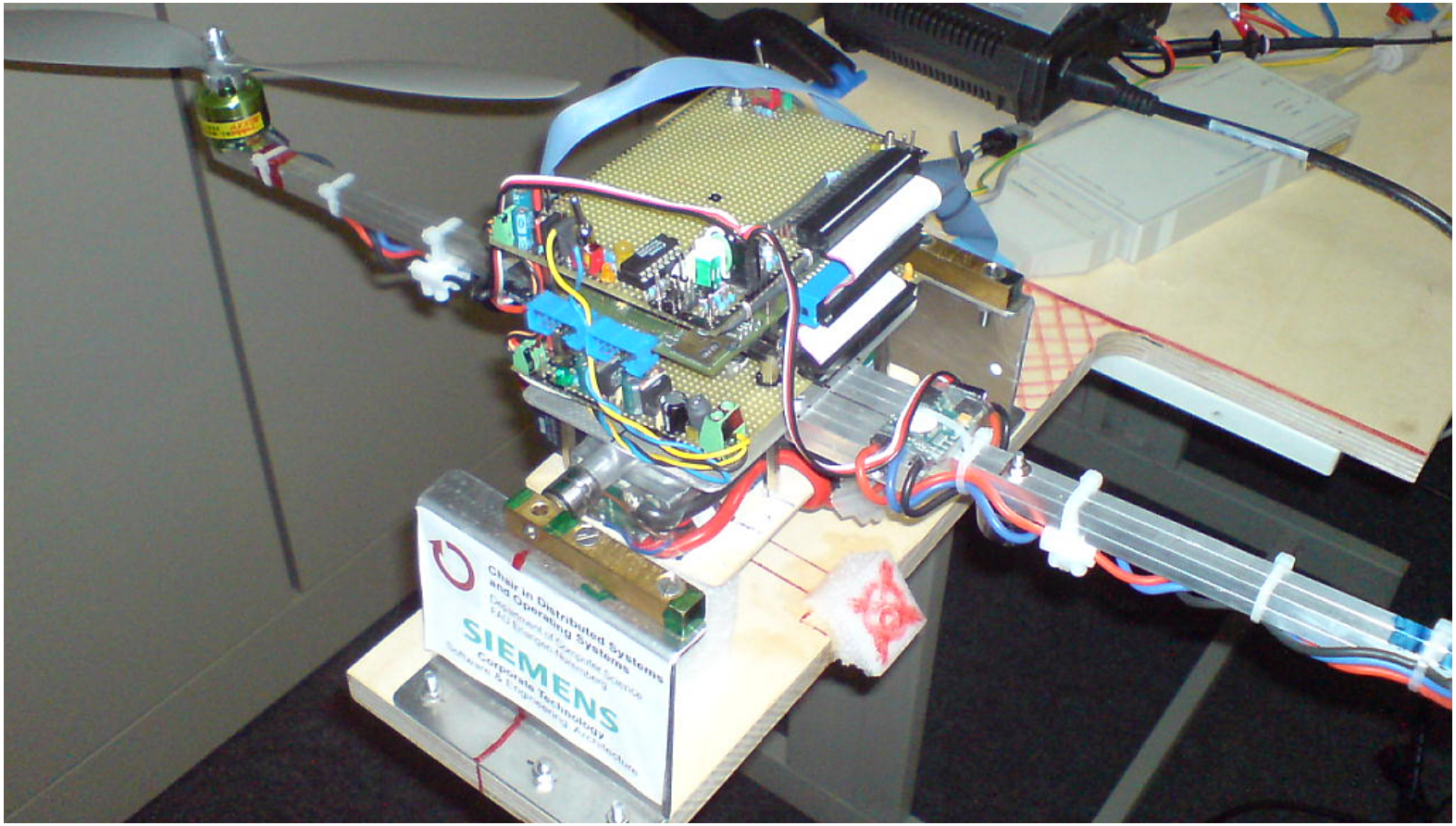
Aufgabenstellung

- Modellierung eines physikalischen Systems
- Implementierung eines Reglers
- Betriebssystembasis OSEK

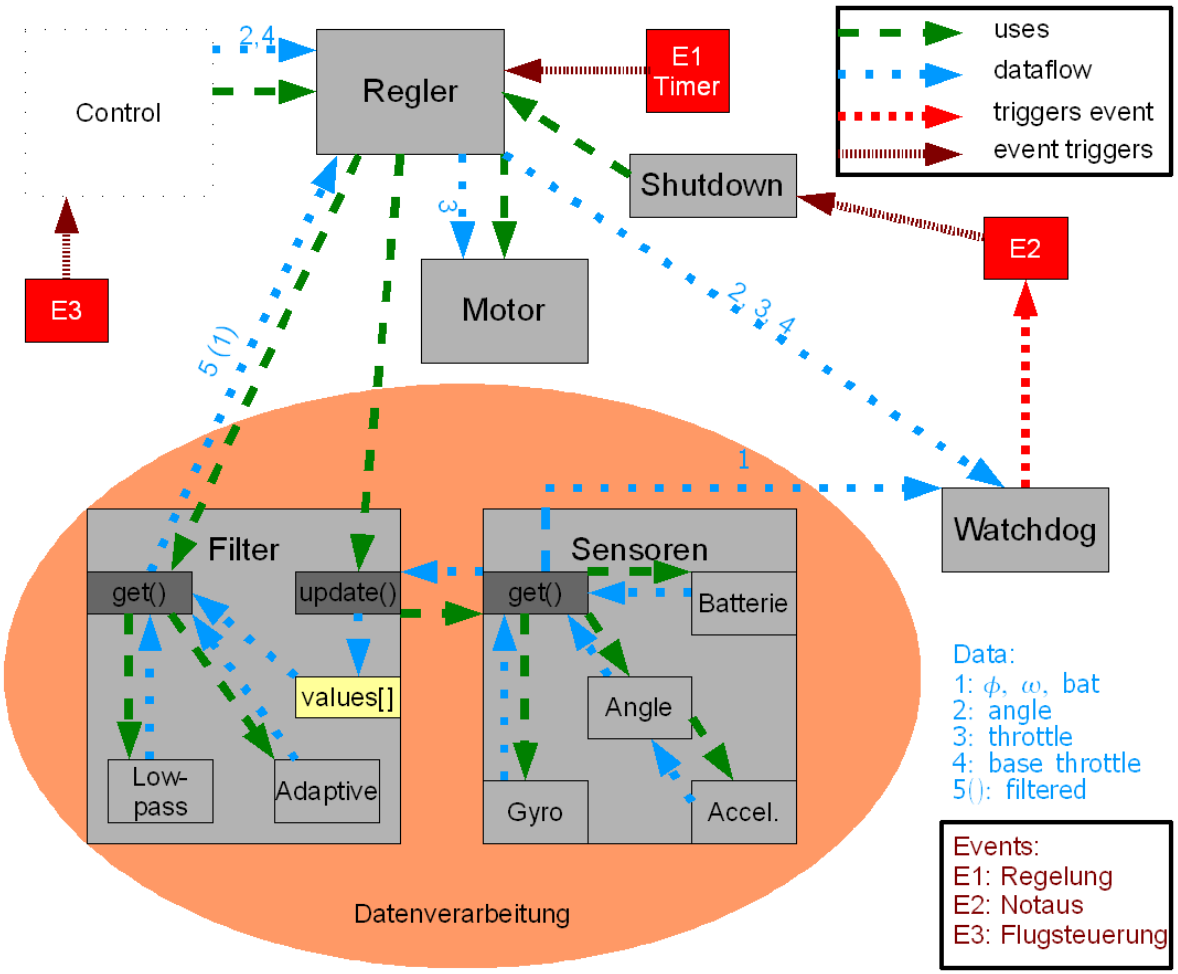


Reduzierung des Copters auf eine Achse

Die Wippe


Chair in Distributed Systems and Operating Systems
SIEMENS
Corporate Technology

Übersicht der Komponenten



Aufgaben des Reglers

- Datenentnahme beim Filter/Sensormodul
- Berechnung des neuen Stellwertes
- Steuerung des Motormoduls bzw. Setzen des neuen Stellwertes

Aufgaben der Datenverarbeitung

Beinhaltet:

- Filter
- Sensoren
  - Gyroskop
  - Accellerator
  - Batteriesensor

→ *Liefert Daten für den Regler*

Aufgaben der Sensoren

Aufgaben des Sensormoduls:

- Schnittstelle zum physikalischen Umfeld
- Messung der Umgebungsdaten
- Abstraktion des aktuellen Zustands
- Bereitstellung der Daten

Aufgaben der Filter

Aufgaben des Filtermoduls:

- Besorgen der Rohdaten von den Sensoren
- Filterung der übergebenen Werte
- Bereitstellung der Daten für die Regelung

Filtertypen

Arten von Filter

- Digitales Tiefpassfilter
  → Unterdrückung irrelevanter Frequenzen
- Adaptives Filter (allerdings nicht implementiert)
  → Rauschminimierung

Aufgaben der Motorkontrolle

Aufgaben des Motorkontrollmoduls:

- Setzt die Vorgaben des Regelers um
- Hardwareschnittstelle zu den Motoren
- Initialisierung des Antriebs

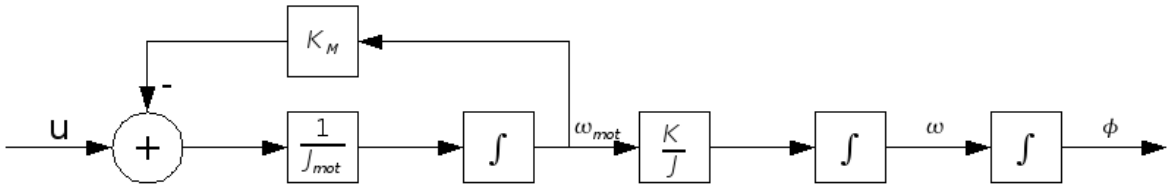
Aufgaben des Kontrollmoduls

Optionales Zusatzmodul zur Einstellung des Neigungswinkels

Aufgaben des Kontrollmoduls:

- Interface für den Benutzer
- Setzen entsprechender Sollwerte in der Regelung

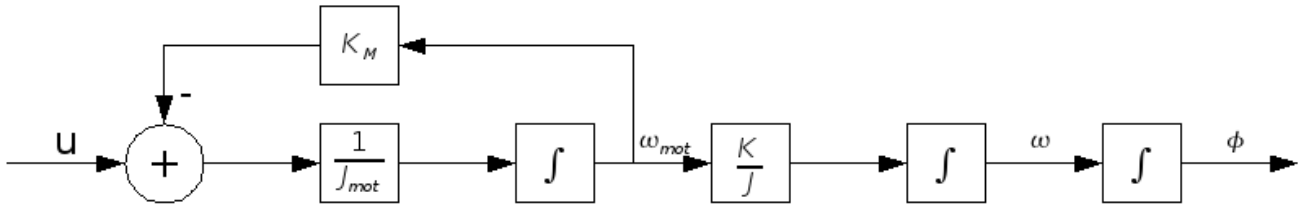
Regelstrecke



→ lineare Näherung an Physik des Copters

- u Motorspannung
- J_{mot} Trägheitsmoment des Motors
- ω_{mot} Winkelgeschwindigkeit des Motors ↔ Drehzahl
- K_M Motorkonstante ($\frac{u}{\omega}$ Spannung pro Drehzahl)
- $\frac{K}{J}$ Rotorkonstante / Copterträgheit (Umrechnung Winkelgeschwindigkeit auf Drehbeschleunigung)
- ω Winkelgeschwindigkeit des Copters
- ϕ Lage des Copters

Regelstrecke



Zu bestimmende Konstanten im physikalischen Modell:

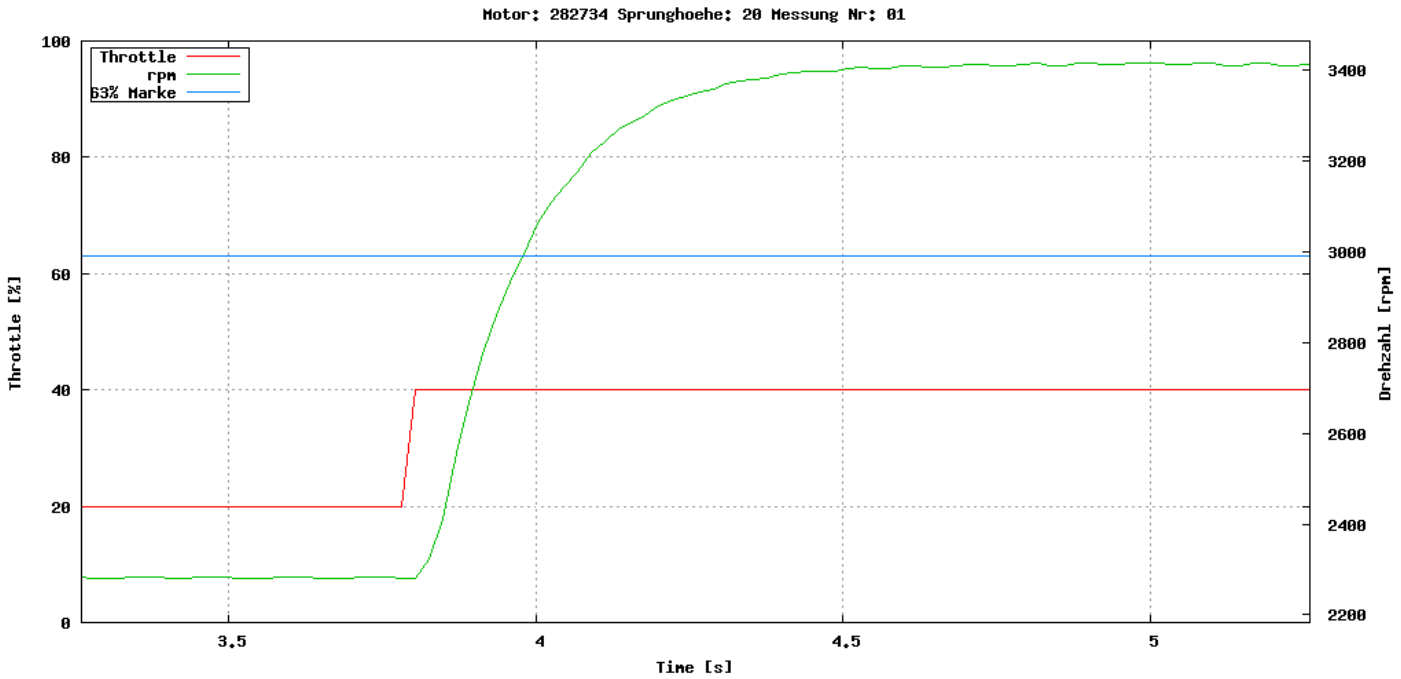
K_M Motorkonstante - entspricht $\frac{U}{\omega}$ mit $[U = \frac{Throttle}{100} * U_{bat}]$

J_{Mot} Motorträgheit - entspricht $[K_M * \tau]$ mit τ = Zeitspanne vom Andrehen des Motors bis zum erreichen von 63% des Sollwertes

$\frac{K}{J}$ Copterkonstante - mit $[K = 2 * K_{Rotor} * l]$ (Rotorkonstante) und der Trägheit J geschätzt nach dem steinerschen Satz

Bestimmung von τ

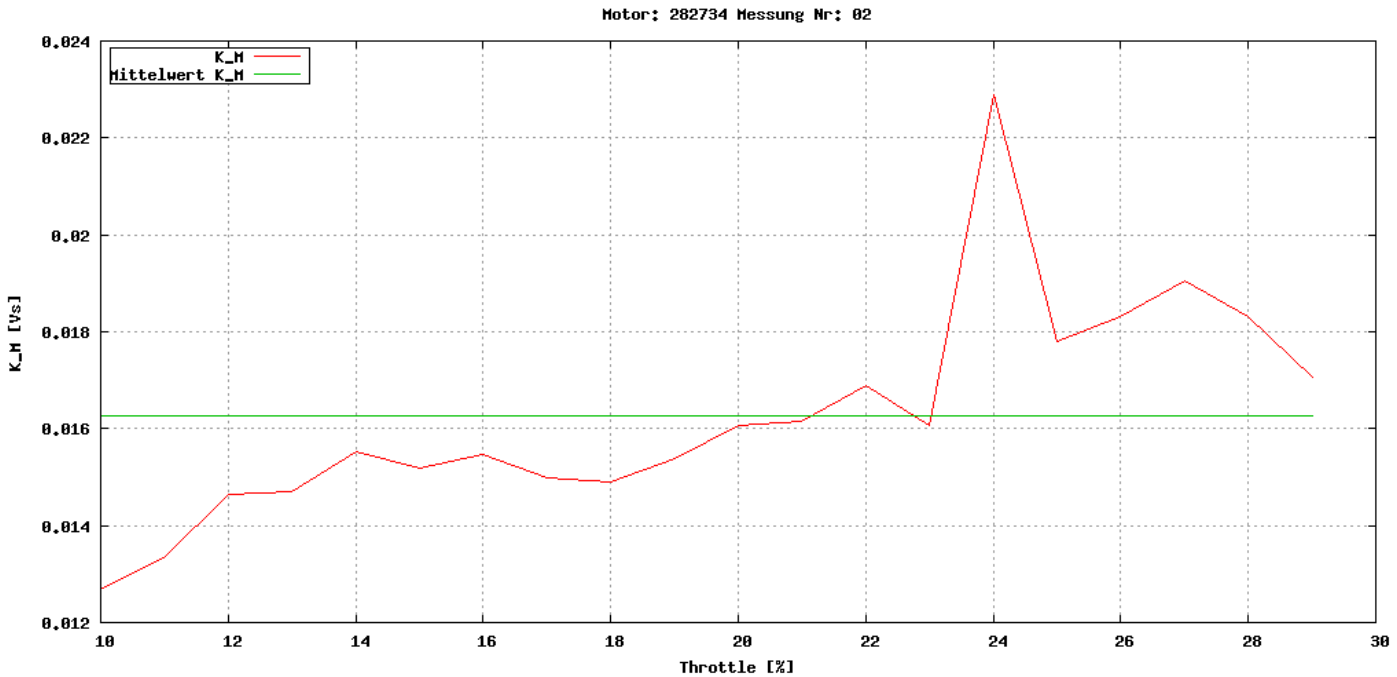
Ablesbar direkt aus den Messungen:



Im Schnitt ergab sich τ mit ca. 185 ms

Bestimmung von K_M

Ablesbar direkt aus den Messungen:



Im Schnitt ergab sich K_M mit ca. 0.0163 Vs

Bestimmung von J_{Mot} und $\frac{K}{J}$

J_{Mot} ist direkt berechenbar aus:

$$J_{Mot} = K_M \cdot \tau = 0.0163 Vs \cdot 0.185 s \approx 0.00302 Vs^2$$

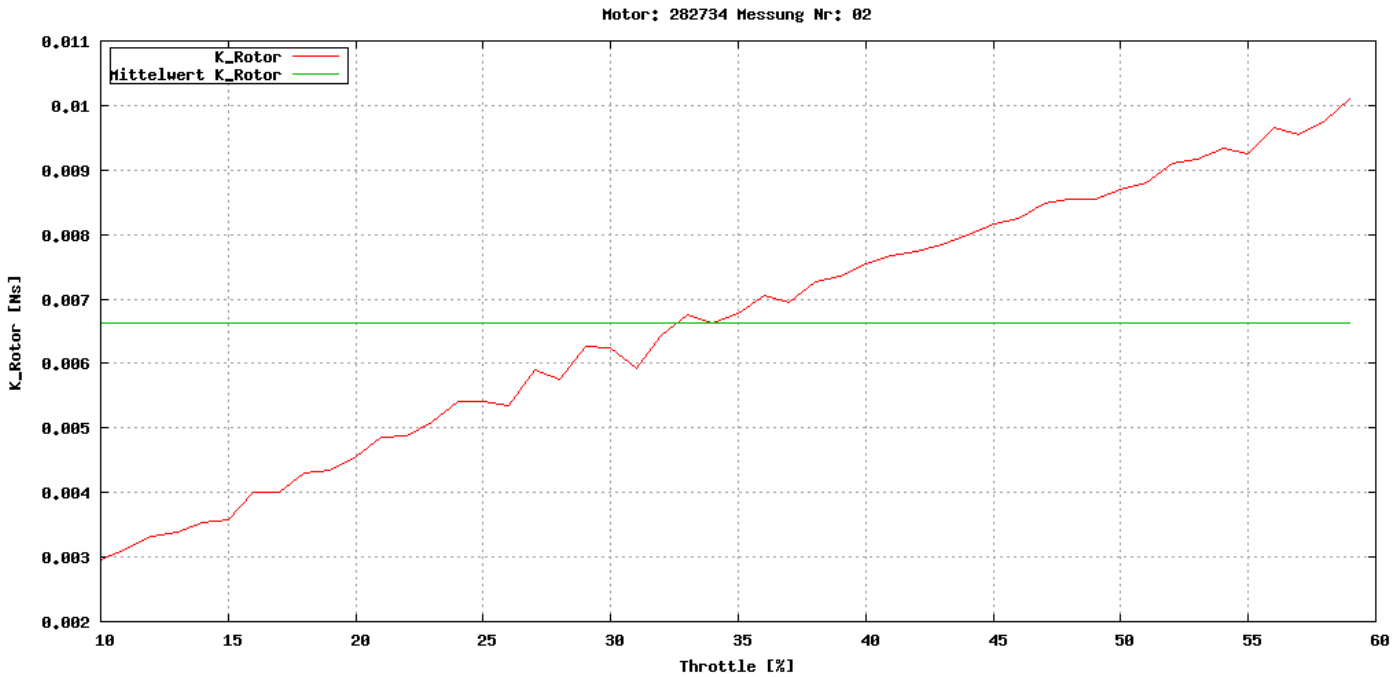
$\frac{K}{J} = \frac{2 * K_{Rotor} * l}{J}$

mit l $\approx$ 0.25 m (Abstand der Motoren vom Mittelpunkt)

mit J $\approx 37.36\ gm^2 \cdot 9.81 \frac{kg}{N}$ (Schätzung der Trägheit)

Bestimmung von K_{Rotor}

K_{Rotor} ist definiert als Kraft pro Umdrehung
→ lineare Mittelung aus den Messungen ablesbar.



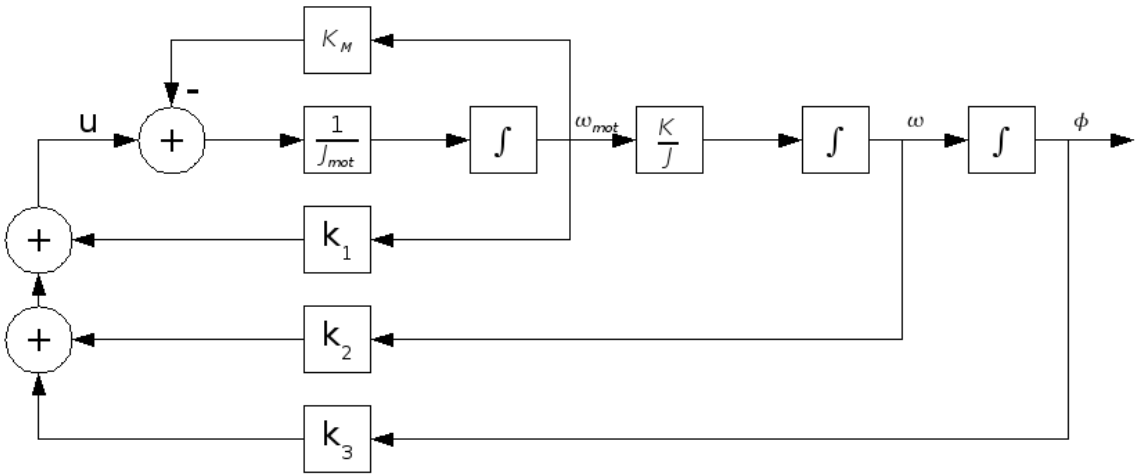
K_{Rotor} ergab sich mit ca. 0.00655 [Ns]

Bestimmung von $\frac{K}{J}$

$\frac{K}{J}$ ergibt sich somit zu:

$$\frac{K}{J} = \frac{2 \cdot 0.00655Ns \cdot 0.25m}{37.36gm^2 \cdot 9.81\frac{kg}{N}} \approx 0.0089Hz$$

Regler



Regelung basiert auf dem physikalischen Modell
→ Drehzahl, Lageänderung, Lage gehen in neuen Stellwert ein.
→ Konstanten k1, k2, k3 sind zu bestimmen.
→ ω und ϕ direkt aus Sensorwerten, ω_{Mot} muss geschätzt werden.

Zustandsraumbeschreibung

Regler lässt sich in eine Zustandsraumbeschreibung überführen

$$\begin{pmatrix} \dot{\omega} \\ \dot{\phi} \\ \dot{\omega}_{Mot} \end{pmatrix} = \begin{pmatrix} 0 & 0 & \frac{K}{J} \\ 1 & 0 & 0 \\ \frac{k2}{J_{Mot}} & \frac{k3}{J_{Mot}} & \frac{k1-K_M}{J_{Mot}} \end{pmatrix} \cdot \begin{pmatrix} \omega \\ \phi \\ \omega_{Mot} \end{pmatrix}$$

Bestimmung der Eigenwerte der Systemmatrix und Auflösen nach den Reglerparametern.

Reglerkonstanten

Nach Auflösen erhält man nun:

$$k1 = (\lambda_1 + \lambda_2 + \lambda_3 + K_M) \cdot J_{Mot}$$

$$k2 = -\left(\frac{\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_2\lambda_3}{\frac{K}{J}}\right) \cdot J_{Mot}$$

$$k3 = \left(\frac{\lambda_1\lambda_2\lambda_3}{\frac{K}{J}}\right) \cdot J_{Mot}$$

Reglerkonstanten

Wählt man die Eigenwerte so, dass diese alle auf der negativen Halbachse liegen erhält man ein BIBO - stabiles System.
→ Copter wird sich nach gewisser Zeit einschwingen.

Im Test haben sich
$\lambda_1 = -1.6,\ \lambda_2 = -0.29 + 0.95j,\ \lambda_3 = -0.29 - 0.95j$
als brauchbar erwiesen.

Somit lauten die Regelparameter:
$k1 \approx -0.006,\ k2 \approx -0.649,\ k3 \approx -0.535$

Nähere Herleitung siehe (1)
BIBO - Bounded Input - Bounded Output

Echtzeitbedingungen und Ereignisse

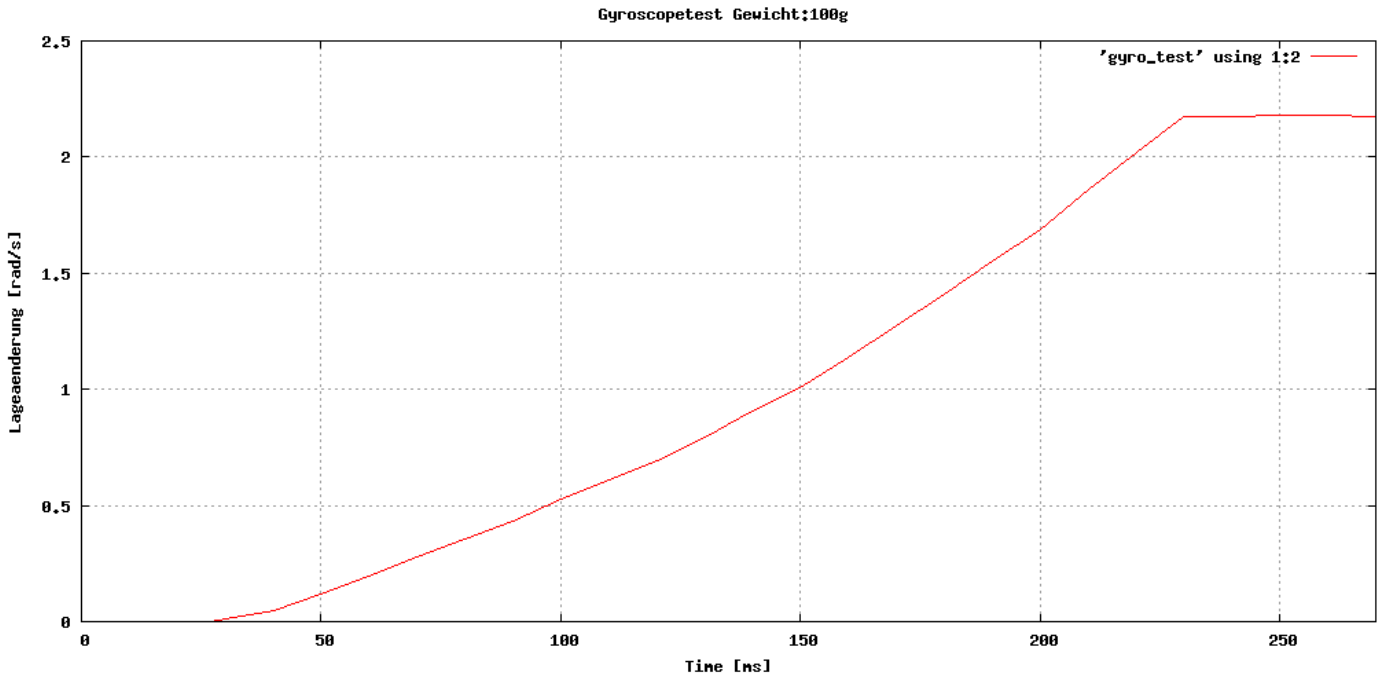
Zu beachtende Ereignisse

- Lageregelung: Alle 22ms, bedingt durch Trägheit der Motorsteuerung → Rechtzeitigkeit sollte hier garantiert werden.
- Flugsteuerung: Vom Anwender erzeugte sporadische Ereignisse. Müssen Deadline einhalten um Reaktionszeit des Systems auf Anwendereingaben konstant gering zu halten.
- Notstop: Kontrolliertes Abschalten des Gesamtsystems innerhalb einer gewissen Zeitspanne.
- Weitere: Einflüsse, wie Wind etc. werden vom Regler ausgeglichen.
- → **Mögliche Umsetzung: zeitgesteuertes System mit Hintergrundbetrieb.**

Gyroskop

Funktion Misst die Winkelgeschwindigkeit, gibt diese in rad/s zurück.

Test Formaler Test schwierig. Definiertes Gewicht an einer Seite des Copters. Abgleich mit Erwartungen.



Accelerator

Funktion Misst die aktuelle Normalkraft, daraus lässt sich anschließend die Lage berechnen.

Test Formaler Test auch schwierig. Mit Winkelmesser ruhende Lage einstellen und mit Sensorwert abgleichen.

Angle

Funktion Berechnet aktuelle Lage in *rad* aus der Normalkraft.

Test Modul setzt nur einfache Berechnungsvorschrift um. Test mit verschiedenen Eingabewerten.

Batterie

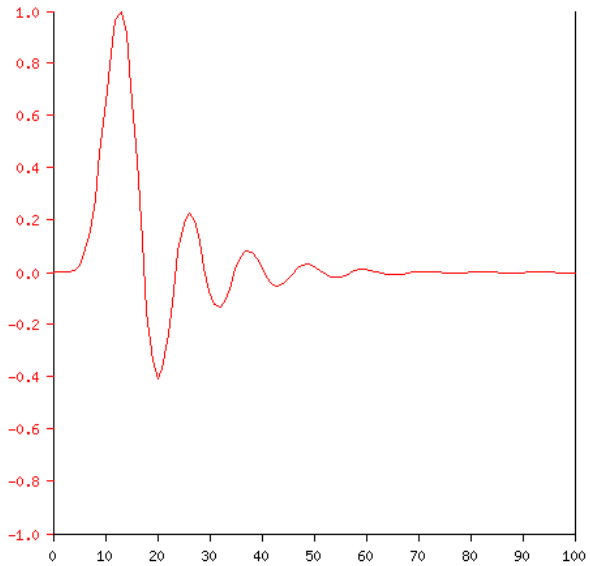
Funktion Misst die aktuelle Eingangsspannung, gibt diese in V zurück.

Test Batterie laden, und entladen. Dabei Messungen mit Multimeter durchführen und mit Sensordaten vergleichen.

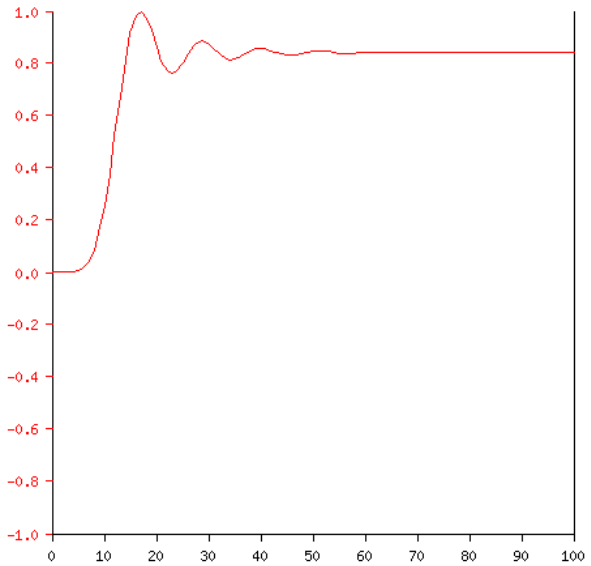
Tiefpass Filter

Funktion Tiefpassfilterung der übergebenen Daten.

Test Impulsantwort und Sprungantwort des Filters überprüfen.



(a) Impulsantwort



(b) Sprungantwort

Adaptive Filter

Funktion Rauschunterdrückung der übergebenen Daten durch Prädiktion der Messwerte.

Test Übergabe von Testdaten und Verifizierung des Ergebnisses durch Nachrechnen. (Matlab o.ä.)

→ Wurde nicht implementiert, da Tiefpass ausreichendes Ergebnis gebracht hat. Adaptives Filter würde dies nicht wesentlich verbessern.

update()

Funktion Aktuelle Sensorwerte vom Sensormodul beziehen, Filterung der Daten und Speicherung in internen Datenstruktur.

Test *values* auslesen und manuell mit Sensordaten abgleichen.

Regler

Funktion Steuert den Motor in Abhängigkeit der gefilterten Daten und der vorgebenen Stellwerte des Kontrollmoduls.

Test Weitergabe einer Fehlstellung durch die Eingabedaten auf den Regler. Erwartung an den Regler ist eine korrekte Reaktion auf das Ereigniss. Verifikation des Ergebnisses durch Nachrechnen.

Test 2 Veränderung des gewünschten Neigungswinkels bei gleichen Sensordaten. Kontrolle ob Regler korrekte Lageänderung einleitet.

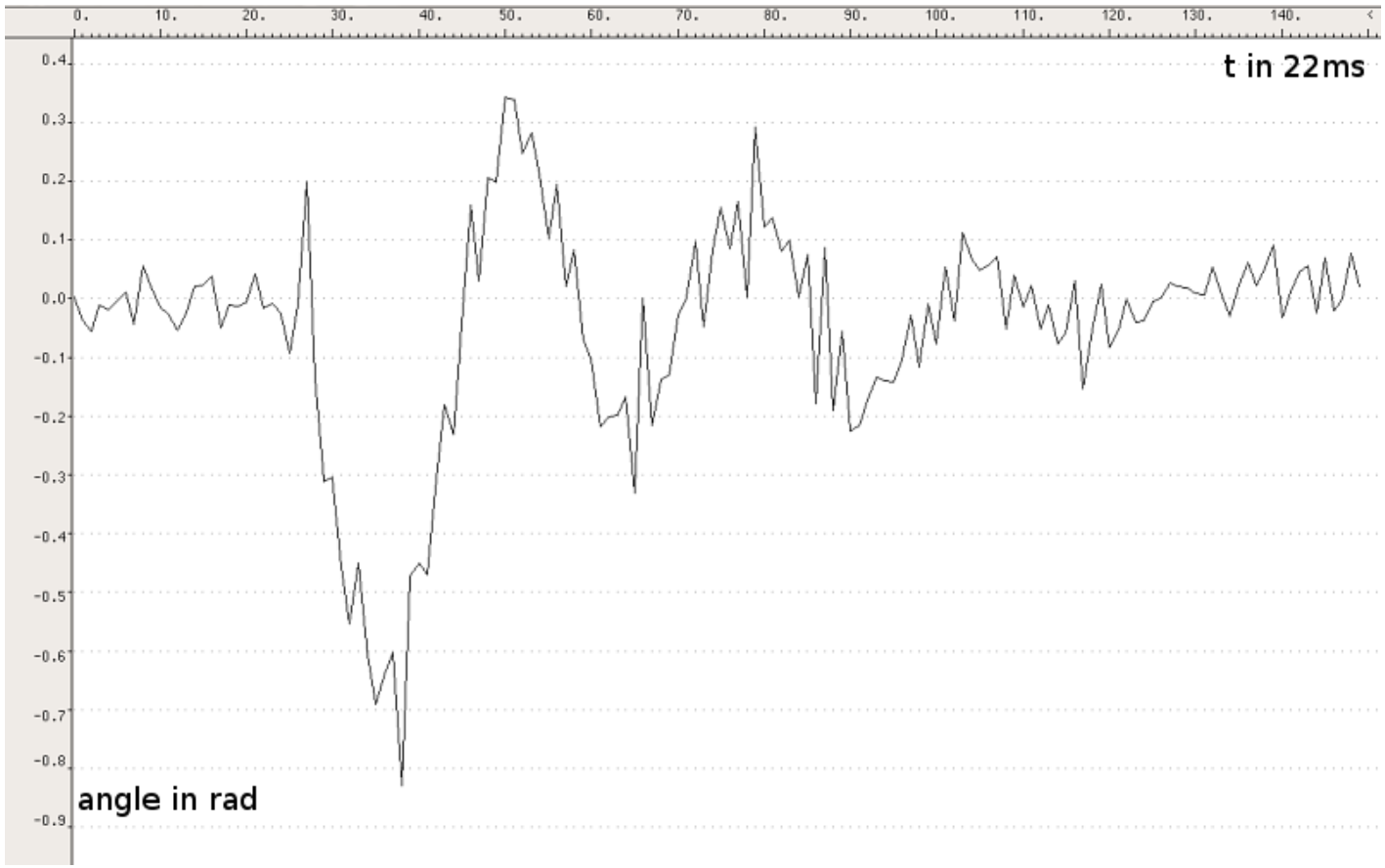
Test 3 Gleichzeitige Änderung der gefilterten Daten (z.B. Windstoß) und des Neigungswinkels. Kontrolle ob Regler korrekte Reaktion zeigt.

Regler

Wie validieren wir die Ausgaben des Reglers?

1. Abgleichen des berechneten Throttles mit nachgerechneten Größen.
2. Nachmessen des vorgegebenen Neigungswinkel am Testobjekt.

Einschwingverhalten



→ Copter schwingt in endlicher Zeit ein

Motor

Funktion Ansteuerung der Rotoren.

Test Übergabe von Stellwerten (Throttle). Resultierende Schubkraft und/oder Drehzahl messen und mit Erwartungen vergleichen.

Abbildung auf ProOSEK-Mittel: Tasks

- Nur 1 Task
- Periode: 2 ms
- Interner Counter
- alle 11 Aktivierungen erfolgt Regelung, sonst nur Auslesen der Sensoren und Filterung.

Abbildung auf ProOSEK-Mittel: Alarme

1 Alarm

- Periode: 2 ms
- Aktiviert den Regelungstask

Zusätzlich:

Alarm zum Warten beim Initalisieren der Motoren
Dabei wird ein Event getriggert

Abbildung auf ProOSEK-Mittel: Synchronisation

In unserer Implementierung unnötig

- Nur 1 Task
- Keine Interrupts (außer vom Timer) - Sensoren werden gepollt

Demonstration

- Demo 1: Lagekontrolle - Einschwingverhalten
- Demo 2: Lageänderung - Vorgabewinkel verändern

  → in 10° Schritten (0°, 10°, -10°, 20°, ...)

Quellen

1. Quadkopter (Studienarbeit Inf4) Benedikt Dremel, Sebastian Harl, Sebastian Kotulla Erlangen 2008
2. http://de.wikipedia.org/wiki/Quadrocopter
3. http://portal.osek-vdx.org/files/pdf/specs/os223.pdf
4. http://www-users.cs.york.ac.uk/ fisher/mkfilter/
5. TC1796 32-Bit Single-Chip Microcontroller User's Manual, V2.0
6. IDG-300 Datasheet
7. ADXL213 Datasheet

Danke für die Aufmerksamkeit!