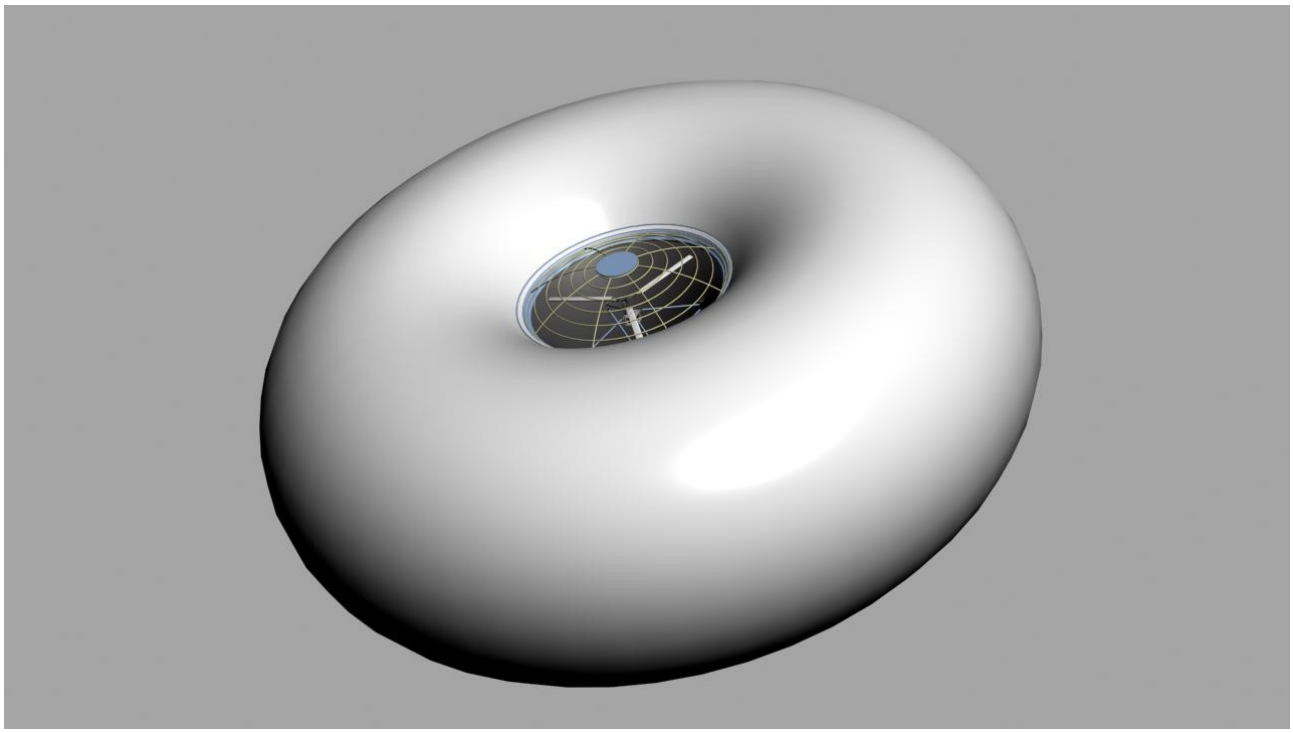
Projekt

Luftschiff-Helikopter-Hybrid



Präsentation

Geschichte

Bilder und Arbeitsproben

Grundlagen

Vorgestellt wird ein Auftriebskörper dessen grundsätzlich symmetrische Auslegung beliebige Orientierung und Stabilisierung ermöglicht.
Der prinzipielle Ansatz ermöglicht auch submarine oder extra-terrestrische Einsätze.

Alle momentan zur Verfügung stehenden Plattformen für beliebige Sensorik oder Aktorik haben konstruktionsbedingte Einschränkungen ihrer Freiheitsgrade
Diese Reduzierung der Freiheitsgrade wird durch die vorgestellte Erfindung aufgehoben.

Der Vergleich eines Fischs mit einer Qualle soll im folgenden die Intention veranschaulichen.

Ein Fisch hat, ähnlich wie ein Flugzeug oder Luftschiff, eine artbedingte Ober- und Unterseite (z,yaw,gier), ebenso die rechte und linke Seite (y,pitch,nick) und eine Vorder- und Hinterseite (x,roll,roll) welche gleichzeitig die Vortriebsrichtung ist.
Diese Anordnung ermöglicht nahezu beliebige Positionierung im Raum (steigen-sinken,drehen rechts-links und begrenztes kippen), jedoch keine freie Orientierung.
Eine Qualle hingegen hat einen rotationssymmetrischen Aufbau. Sie hat noch eine Ober- und Unterseite, ist aber vom Prinzip her fähig, sich mit ihrem innen liegenden Wasserrückstoß-Antrieb in jegliche Raumrichtung zu bewegen.

Wenn man diesen einfachen aber vielversprechenden Ansatz weiterentwickelt, kommt man zu einem rotationssymmetrischen ringförmigen Auftriebskörper, welcher das System von der Schwerkraft entkoppelt, und einem möglichst alle Freiheitsgrade abdeckenden innen liegendem Antrieb.
Diese technologische Umsetzung ermöglicht eine beliebig positionier- und orientierbare Plattform für jegliche denkbare Medien in denen Auftriebskörper einsetzbar sind.

Momentan verfügbare auf irgendeine Weise in flüssigen oder gasförmigen Medien steuerbare Objekte lassen sich in zwei Klassen aufteilen:

Dynamischer Auftrieb, die Schwerkraft wird aktiv durch aufbringen einer Gegenkraft überwunden:

- über Tragflächen, die durch Vortrieb in gewissen Geschwindigkeitsbereichen Auftrieb erzeugen (Flugzeuge, aber auch Drachen, Segler, Gleitschirme, Basejumper, usw.)
- durch lokale Drehflügler, die nach gleichen dynamischen Prinzipien aber mit höherem Energieaufwand sogar freies Schweben im Medium erreichen können (Hubschrauber, Multicopter)

Statischer Auftrieb, die Schwerkraft wird durch ein Auftriebsmedium in einem Auftriebskörper neutralisiert:

- obwohl eine völlige Entkopplung von der Schwerkraft möglich ist, wird in allen bekannten Auslegungen eine systembedingte Begrenzung der Freiheitsgrade vorgenommen (Blimps, Starr- und Halbstarrluftschiffe, Ballone, U-Boote)

Mit den Hybriden Luftschiffen existiert noch eine Mischform, die nur einen Teil des Gewichts durch das Auftriebsmedium aufhebt und den Rest durch Vortrieb mit entsprechenden Auftriebsflächen erzeugt.

Obwohl die prinzipielle konstruktionsbedingte Beschränkung der Freiheitsgrade wohl auf menschlicher ,eher zweidimensionalem Denken entsprechender, Vorstellung beruht, ist sie nicht notwendig.
Der vorgestellte Auftriebskörper ermöglicht diese Befreiung.

Eine weitere Folge dieser Erhöhung der Freiheitsgrade ist auch, das in alle Richtungen gegen Strömungen stabilisiert werden kann.

Bewegen in jede Richtung ↔ Stabilisieren in jede Richtung

Wie bei jedem der Natur ausgesetztem Objekt, kann die Stabilisierung aber nur in den Grenzen der Physik garantiert werden.
Ab gewissen Geschwindigkeiten kann ein Abbruch des Einsatzes notwendig sein, um das Fahrzeug einer Gefährdung zu entziehen.

Aufbau

Ein ringförmiger Auftriebskörper (Reifen - Fig.1,2-1) umschließt eine Stützstruktur (Felge - Fig.1,2-2), welche den Strömungskanal formt und in der der Antrieb aufgebaut ist. Auch sämtliche Bordaggregate sind an dieser Felge befestigt.

Der Antrieb besteht in dieser Auslegung aus zwei gegenläufig drehenden Helikopterrotoren (Fig.1,2-3) mit zyklischer und kollektiver Blattverstellung, und separaten Motor-Getriebe-Einheiten (Fig.1,2-4).

Für den angestrebten leisen Betrieb werden Elektromotoren verwendet.

Dieser Ansatz ermöglicht die Abdeckung von 5 Freiheitsgraden (Fig.3). Eine Drehung um die Rotationssymetrieachse **x** ist nur bei Bewegung möglich, jedoch von geringer Relevanz.

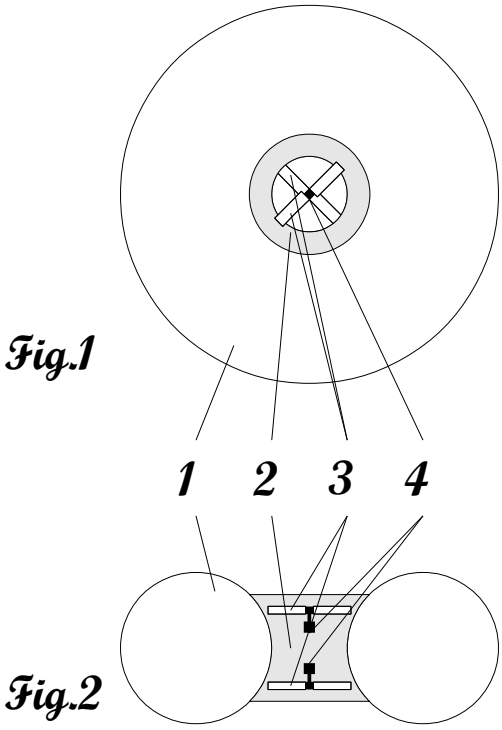
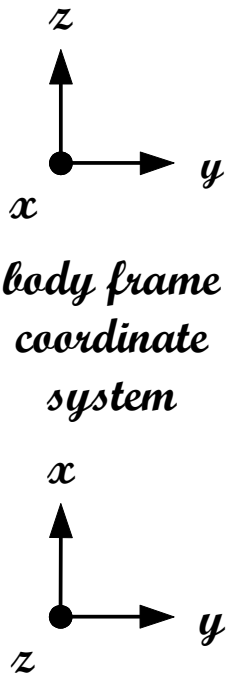


Fig.1

Fig.2



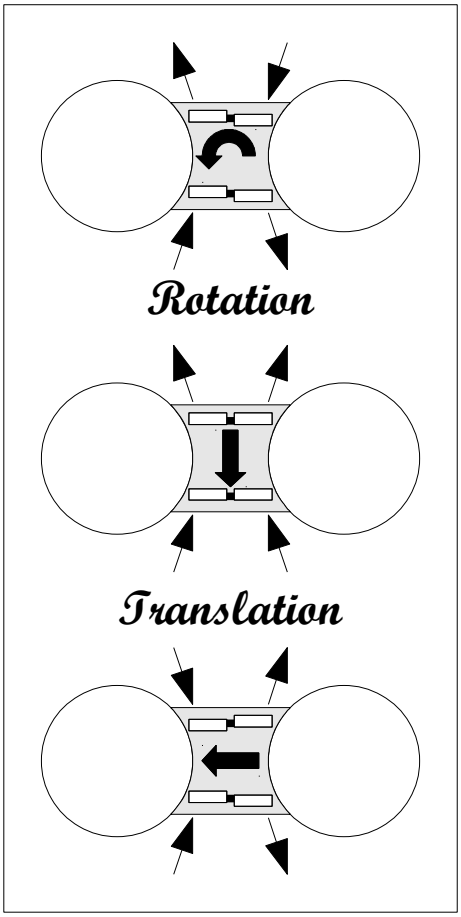


Fig.3

Ein sich frei im Raum bewegendes Objekt benötigt Informationen über seine Lage und Position.

Ein inertiales Navigationssystem (INS) ist daher eine Grundvoraussetzung, egal ob ferngesteuert (RPV - Remotely Piloted Vehicle - Untergruppe der UAV's, nur ferngesteuert), autonom (mit Selbststeuerfähigkeit - Roboter) oder beliebige Mischformen.

Steuerung und/oder Informationsaustausch auf der Erde erfolgen je nach Einsatzgebiet über militärische Frequenzkanäle oder im zivilen Bereich über W-LAN im Nahbereich mit Fallback auf mobile Datennetze und im Extremfall auf Satellitendatenfunk (Iridium usw.)

Um die gewonnen Möglichkeiten der freien Orientierbarkeit nicht einzuschränken, sind alle Nutzlasten, welche eine spezifische Ausrichtung benötigen, in einer Art kardanischen Aufhängung unterzubringen, die sich wiederum beliebig orientieren lässt (Richtfunkantennen, Kameras, Sensoren, Sitze für Personen etc.).

Dieses vorgestellte System ist in weiten Bereichen seiner Tragfähigkeit skalierbar.

Einsatzgebiete

Grundlegende Eigenschaften des vorgestellten Fahrzeugs sind:

- beliebig orientier- und positionierbar im Gegensatz zu allen bisher bekannten Objekten
- geräusch- und vibrationsarm im Gegensatz zu Flugzeugen und Drehflüglern
- dynamische Auftriebsanpassung im Gegensatz zu bekannten Auftriebskörpern
- beliebige Aktuatoren anbaubar ("Roboterarme") mit beliebiger Sensorik

Die möglichen Einsatzgebiete sind sehr vielfältig:
(teilweise mit Angaben von anderen Systemen, welche im jeweiligen Einsatzgebiet heute agieren)

Klassische Aufklärungs- und Überwachungsszenarien

Stabile Langzeitpräsenz in großer Höhe

militärische Drohnen (HALE und MALE) oder Hybrid-Luftschiffe (LEMV)

Geräuscharme Nahaufklärung

Multikopter oder Cipher und ähnliche

Werbeträger

Einmaliges Erscheinungsbild

Luftschiffe (bemannt oder ferngesteuert)

Beliebige luftdurchlässige Leichtbauformen können um das Fahrzeug aufgebaut werden

neu

Audiovisuelle Präsentation

Projektionsplattform für Musikvisualisierungen auf großen Veranstaltungen

neu

SAR (Search and Rescue)

Durch leise Antriebsauslegung kann auch akustisch gesucht werden

neu

Durch freie Orientierbarkeit kann auch an senkrechten oder überhängenden Wänden agiert werden (Berge, Gebäude, Brücken, usw.)

neu

Katastrophenschutz

Kommunikationsrelaisstation

Mapping und Leitsystem für Bodeneinsätze

Minensuche

Suche in Bodennähe ohne Bodenkontakt und Gefährdung von Leben

neu

Forschung

Geologische und geografische Vermessungen an beliebigen Punkten im Raum

Laser - Vermessungen mit Bodenstationen oder Flugobjekten

Sensorenverbringung an entlegene Stellen

Minimalinvasive biologische Beobachtungen und Untersuchungen

(Sammeln im Urwalddach, Tierbeobachtung usw.)

Forschungsplattform für extra-terrestrische Atmosphären oder Unterwasser

Wartungsaufgaben

Stromnetzinspektion und Wartung

Hubschrauber und Ferngläser

Gebäudevermessungen und Inspektion

Brückeninspektionen usw.

Robotik

Jegliche vorstellbare autonome Aktion eines beliebig im Raum orientierbaren Fahrzeugs

Transport

Koppelung von mehreren Fahrzeugen möglich
ein Vergleich mit Ameisen:
kleine Fracht - eine Ameise
große Fracht - mehrere Ameisen

Sport

Wettkämpfe zwischen verschiedenen Fahrzeugen
(Rennen, 3d-Parcour, usw.)

Entertainment

Riesenmarionetten mit Fahrzeugen an den Gelenkpunkten können beliebiges darstellen
neu

Touristik

Personentragende Großplattformen können beliebige Ziele ansteuern
Whale watching aus der Luft

Medien

Kameraplattform
geringere Vibration und Lärmbelästigung als Drehflügler
höhere Manöverierbarkeit als LTA's
Freie Trajektorien mit Wiederholung oder Zeitraffung, -dehnung oder -umkehrung

Vorzüge des LTA (Lighter Than Air) Prinzips

Einen guten Überblick über die Einsatzmöglichkeiten und Vorteile von LTA's bietet der Wikipedia-Artikel des Zeppelin NT unter Einsatzgebiete. http://de.wikipedia.org/wiki/Zeppelin_NT#Einsatzgebiete

Zitat aus Wikipedia-Artikel http://en.wikipedia.org/wiki/Long_Endurance_Multi-intelligence_Vehicle

“While reconnaissance can be undertaken by fighter aircraft, the costs involved for such a flight were estimated in 2010 to be $10,000–20,000 per flight hour, plus an additional $10,000 in recapitalization costs. Helicopters are more affordable than their fighter equivalent, and can intervene like fighters if weapons are needed, but they are noisy and vulnerable, have very low endurance, and are still not cheap to operate. Hybrid airships can operate from any small forward base, like a helicopter. Their operating cost is likely to be better than any other surveillance option, as is their endurance, which can be measured in weeks.”

Projektgeschichte

Im Jahr 2000 hatte ich mit einem Freund aus Marseille die Idee ,ein ferngesteuertes Luftschiff für Luftbildaufnahmen zu entwickeln. Ein Jahr später, auf der Suche nach einem Alleinstellungsmerkmal, hatte ich eines Tages die Eingebung eines ringförmigen Auftriebskörpers.

In den nächsten 5 Jahren, die ich Halbjahresweise in Marseille verbrachte, versuchten wir im geheimen einen Prototyp mit 5 Metern Durchmesser zu entwickeln und zu bauen.

Später entwickelte ich den Prototyp am Bodensee bei meiner Familie selbständig weiter - in Sichtweite des umher fliegenden Zeppelin NT.

Inzwischen lebe ich wieder im Raum Tübingen, wo ich einst Physik studierte.

Von einem Bekannten wurde mir geraten, die Erfindung patentieren zu lassen, um den Erstanspruch zu sichern und mit diesem Rechtsschutz potentielle Geldgeber zu finden.

In diesen Zeitraum fiel auch die Erkenntnis, dass durch die fortschreitende Miniaturisierung der RC-Microhelikopter ein 1 Meter durchmessender Prototyp möglich wurde, welcher für einen Bruchteil der Kosten realisierbar wäre.

Ich bin seit Sommer 2015 im Besitz eines eingetragenen Gebrauchsmusters auf diese Erfindung.

Dieter Herz

Bilder und Arbeitsproben

Marseille

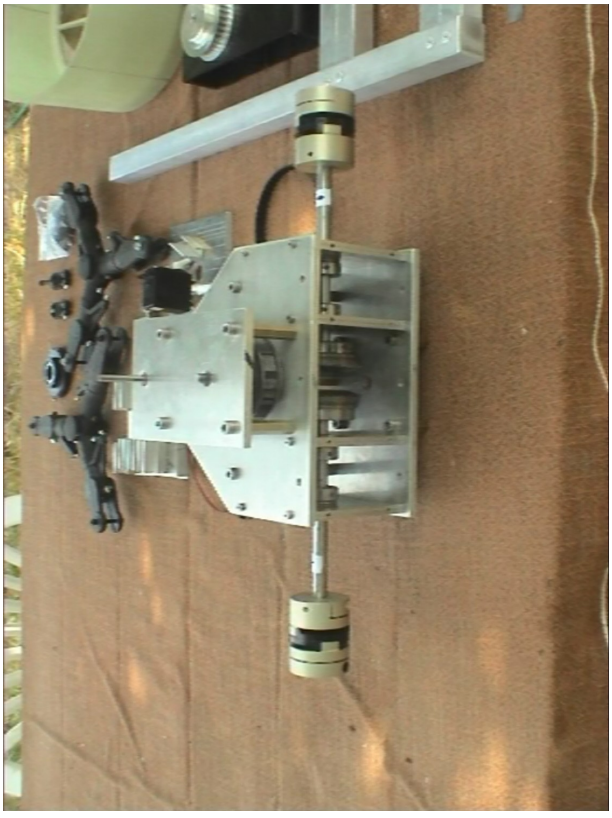
Technischer Aufbau
5 Meter durchmessender Prototyp

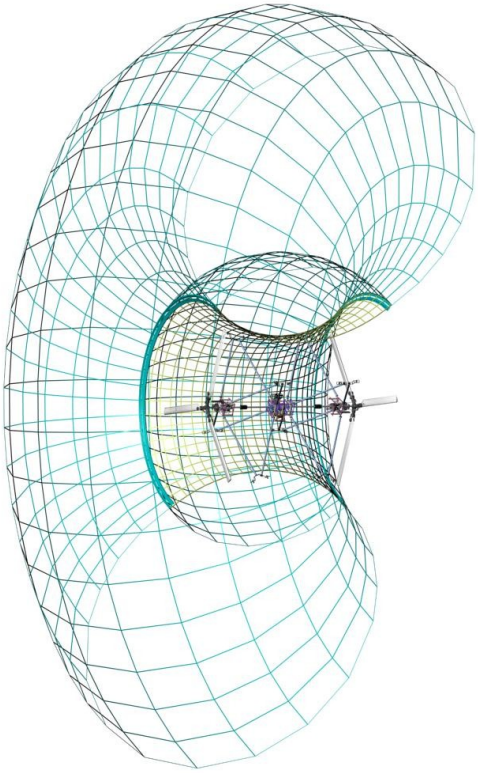
INS
Inertiales Navigations-System

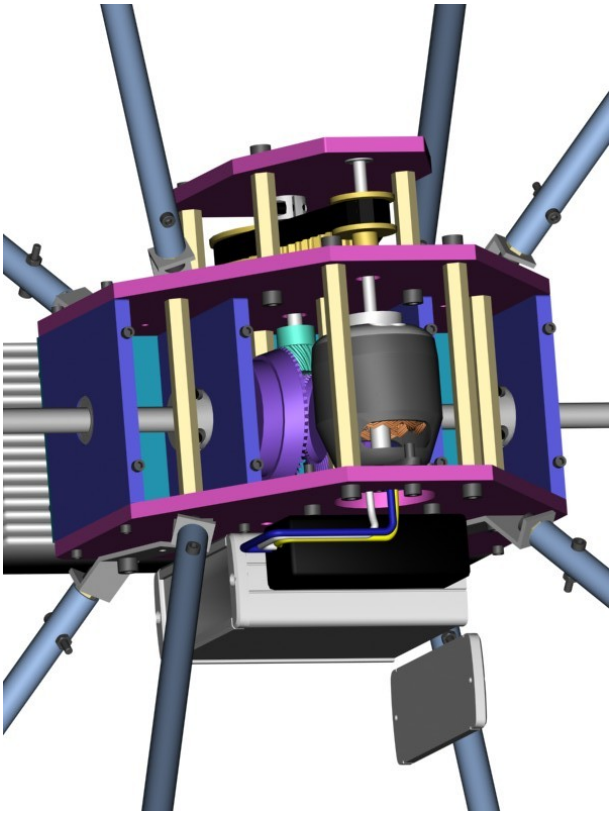
Impressionen

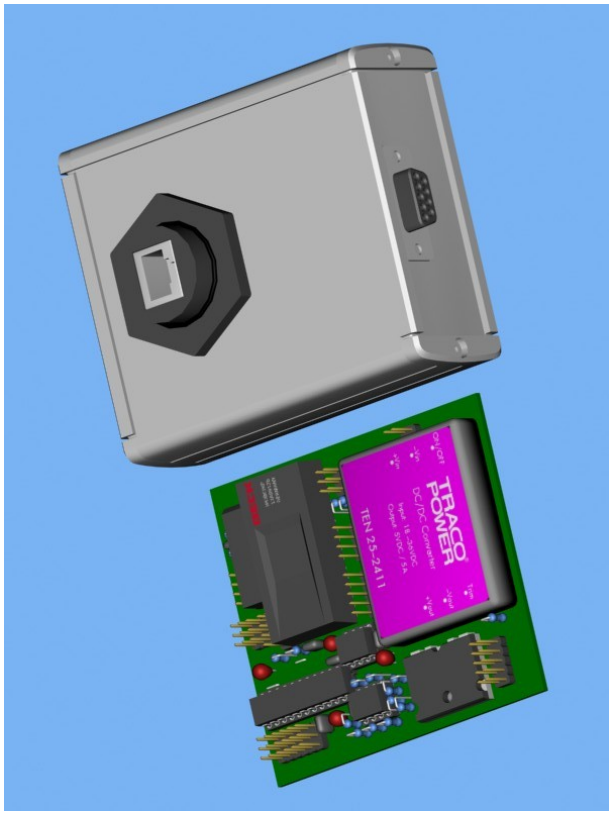
Herleitung einer geschlossenen Lösung zur Ansteuerung der Taumelscheibe einer Helikoptermechanik mit dem Ansatz der parallelen Kinematik

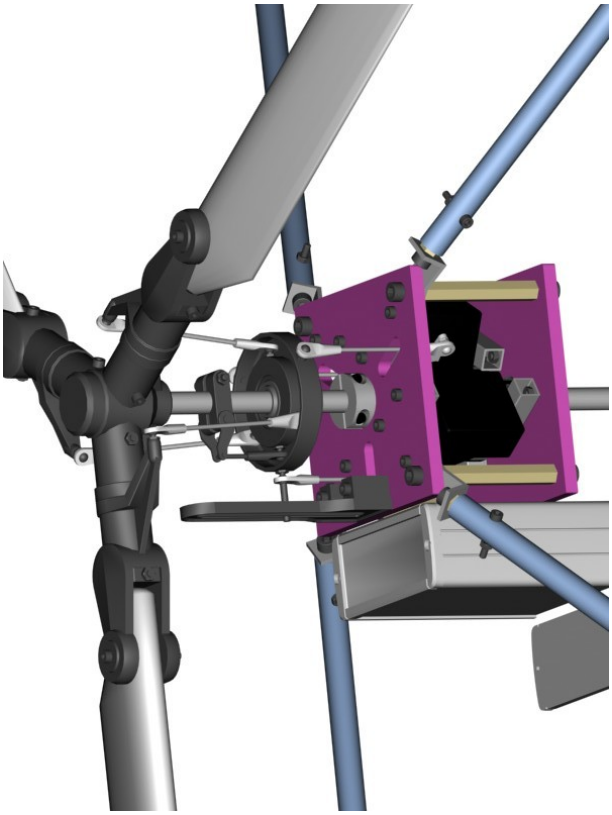
Programmiertechnische Umsetzung vorheriger Herleitung in Pascal

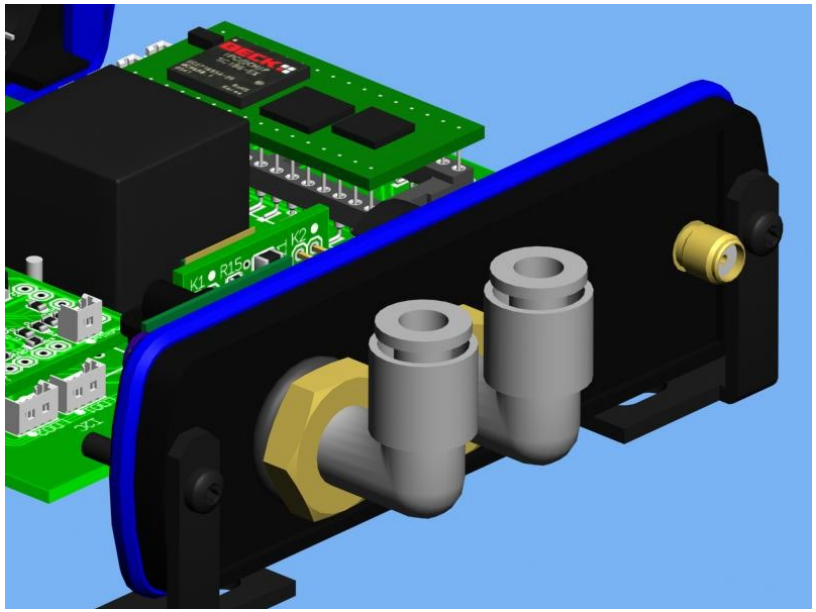


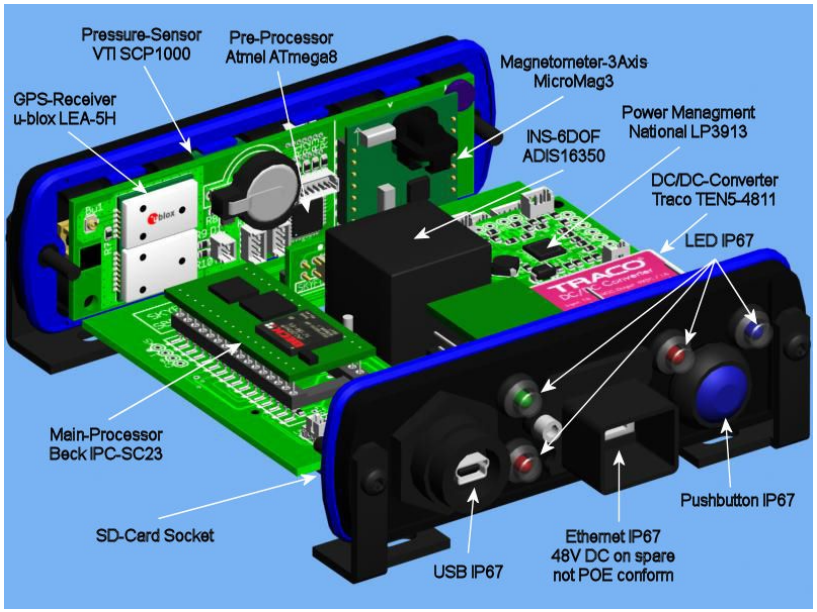


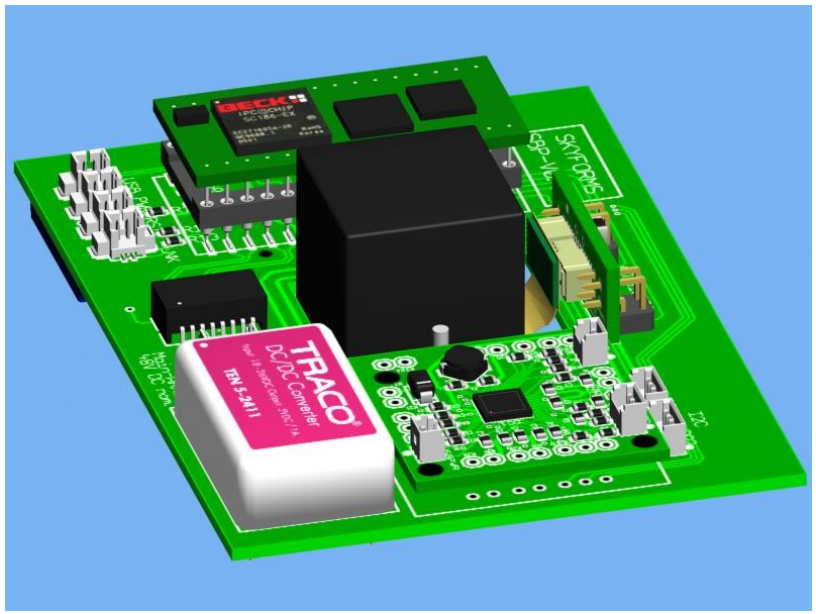


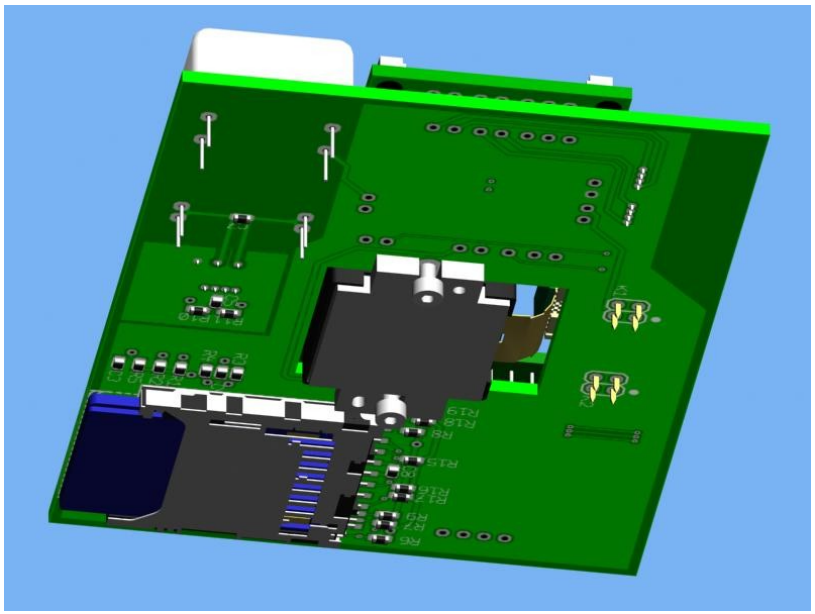

TRACO
POWER






Pressure-Sensor
VTI SCP1000
Pre-Processor
Atmel ATmega8
Magnetometer-3Axis
MicroMag3
GPS-Receiver
u-blox LEA-5H
Power Managment
National LP3913
INS-6DOF
ADIS16350
DC/DC-Converter
Traco TEN5-4811
LED IP67
Main-Processor
Beck IPC-SC23
Pushbutton IP67
SD-Card Socket
Ethernet IP67
48V DC on spare
not POE conform
USB IP67





#1: CaseMode ≔ Sensitive

#2: InputMode ≔ Word

$$\#3:\quad M \coloneqq \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) & 0 \\ \sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\omega) & \sin(\omega) \\ 0 & -\sin(\omega) & \cos(\omega) \end{bmatrix} \cdot \begin{bmatrix} \cos(\varphi) & \sin(\varphi) & 0 \\ -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

M ist die Matrixoperation der Taumelscheibenorientierung
Matrixmultiplikation immer von rechts nach links
drehung um φ (kipprichtung)
kippen um ω
rückdrehung φ

#4: A ≔ [r·COS(τ), r·SIN(τ), 0]

A ist der Vektor des Taumelscheibenrandpunkts der Schubstangenankopplung
r Radius
τ Winkel – bei uns drei Punkte alle 120°

#5: B ≔ [0, 0, p]

B ist der Vektor des Taumelscheibenpitch
p Pitch

#6: C ≔ [COS(τ)·(b + l·COS(ε)), SIN(τ)·(b + l·COS(ε)), h + l·SIN(ε)]

C ist der Vektor der Servo-Schubstangenankopplung
b Servoradius
l Servoarmlänge
h Höhe unter Taumelscheibe
e das was wir eigentlich suchen – der Servoarmwinkel
τ Winkel siehe A (wir setzen die Servos ja wohl direkt unter die Taumelscheibenkopplungen)

#7: M·A + B – C

M*A + B ist der also der Vektor eines der drei gepitschten und gekippten Taumelscheibenrandpunkte und C der Vektor des dazugehörigen Servoarmpunkts.
Die Differenz ist also der Verbindungsvektor – die Schubstange.
Deren Länge ist konstant und gleich h.
Also ist der Betrag des Verbindunsvektors gleich h – im Quadrat leichter

$$\#8:\quad (M\cdot A + B - C)^2 = h^2$$

Sieht ja einfach aus aber ausgerechnet ergibt sich das

$$\#9:\quad -\cos(\varepsilon)\cdot(\cos(\omega)\cdot(2\cdot l\cdot r\cdot\cos(\varphi)^2\cdot\sin(\tau)^2 - 4\cdot l\cdot r\cdot\sin(\varphi)\cdot\cos(\varphi)\cdot\sin(\tau)\cdot\cos(\tau) + 2\cdot l\cdot r\cdot\sin(\varphi)^2\cdot\cos(\tau)^2) + 2\cdot l\cdot r\cdot\cos(\varphi)^2\cdot\cos(\tau)^2 +$$

$$4\cdot l\cdot r\cdot SIN(\phi)\cdot COS(\phi)\cdot SIN(\tau)\cdot COS(\tau) + 2\cdot l\cdot r\cdot SIN(\phi)^2\cdot SIN(\tau)^2 - 2\cdot b\cdot l) +$$

$$SIN(\varepsilon)\cdot(SIN(\omega)\cdot(2\cdot l\cdot r\cdot COS(\phi)\cdot SIN(\tau) - 2\cdot l\cdot r\cdot SIN(\phi)\cdot COS(\tau)) + 2\cdot l\cdot(h - p)) -$$

$$COS(\omega)\cdot(2\cdot b\cdot r\cdot COS(\phi)^2\cdot SIN(\tau)^2 - 4\cdot b\cdot r\cdot SIN(\phi)\cdot COS(\phi)\cdot SIN(\tau)\cdot COS(\tau) +$$

$$2\cdot b\cdot r\cdot SIN(\phi)^2\cdot COS(\tau)^2) + SIN(\omega)\cdot(2\cdot r\cdot(h - p)\cdot COS(\phi)\cdot SIN(\tau) + 2\cdot r\cdot(p -$$

$$h)\cdot SIN(\phi)\cdot COS(\tau)) + COS(\phi)^2\cdot(r^2 - 2\cdot b\cdot r\cdot COS(\tau)^2) -$$

$$4\cdot b\cdot r\cdot SIN(\phi)\cdot COS(\phi)\cdot SIN(\tau)\cdot COS(\tau) + SIN(\phi)^2\cdot(r^2 - 2\cdot b\cdot r\cdot SIN(\tau)^2) + b^2 + h^2 -$$

$$2\cdot h\cdot p + l^2 + p^2 = h^2$$

Das geschickt faktorisiert, die Hauptarbeit, ergibt nach vielen Schritten
mit

#10: $i := SIN(\phi)\cdot COS(\tau) - COS(\phi)\cdot SIN(\tau)$

#11: $2\cdot(COS(\varepsilon)\cdot l + b)\cdot(b + r\cdot(i^2 - 1) - i^2\cdot r\cdot COS(\omega)) + 2\cdot(SIN(\varepsilon)\cdot l + (h - p))\cdot(h - p$

$$- i\cdot r\cdot SIN(\omega)) - (h - p)^2 + l^2 - b^2 + r^2 - h^2 = 0$$

Noch etwas umgeformt und mit b := r − l (es ist wirklich das gleiche)

#12: $COS(\varepsilon)\cdot\left(i^2\cdot(1 - COS(\omega)) - \frac{l}{r}\right) + SIN(\varepsilon)\cdot\left(\frac{h - p}{r} - i\cdot SIN(\omega)\right) + \frac{h - p}{l}\cdot\left(\frac{h - p}{r}\right.$

$$\left. - i\cdot SIN(\omega)\right) + \frac{r - l}{l}\cdot\left(i^2\cdot(1 - COS(\omega)) - \frac{l}{r}\right) + \left(\left(1 - \frac{h^2}{2\cdot l\cdot r}\right) - \frac{(h - p)^2}{2\cdot l\cdot r}\right) =$$

$$0$$

Nun ein paar Substitutionen wie oben mit i (wir brauchen eine möglichst einfache Formel die nach ε gelöst werden kann, gleichzeitig sollte auch noch je nach Veränderlichkeit getrennt werden – schnelles Programm.

#13: $s := \frac{l}{r}$

#14: $COS(\varepsilon)\cdot(i^2\cdot(1 - COS(\omega)) - s) + SIN(\varepsilon)\cdot\left(\frac{h - p}{r} - i\cdot SIN(\omega)\right) + \frac{h - p}{l}\cdot\left(\frac{h - p}{r} -\right.$

$$i \cdot \mathrm{SIN}(\omega)\Big) + \frac{r - l}{l} \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \left(\left(1 - \frac{h^2}{2 \cdot l \cdot r}\right) - \frac{(h - p)^2}{2 \cdot l \cdot r}\right) = 0$$

#15: $t := \dfrac{r - l}{l}$

#16: $$\mathrm{COS}(\varepsilon) \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \mathrm{SIN}(\varepsilon) \cdot \left(\frac{h - p}{r} - i \cdot \mathrm{SIN}(\omega)\right) + \frac{h - p}{l} \cdot \left(\frac{h - p}{r} - i \cdot \mathrm{SIN}(\omega)\right) + t \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \left(\left(1 - \frac{h^2}{2 \cdot l \cdot r}\right) - \frac{(h - p)^2}{2 \cdot l \cdot r}\right) = 0$$

#17: $u := 2 \cdot l \cdot r$

#18: $$\mathrm{COS}(\varepsilon) \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \mathrm{SIN}(\varepsilon) \cdot \left(\frac{h - p}{r} - i \cdot \mathrm{SIN}(\omega)\right) + \frac{h - p}{l} \cdot \left(\frac{h - p}{r} - i \cdot \mathrm{SIN}(\omega)\right) + t \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \left(\left(1 - \frac{h^2}{u}\right) - \frac{(h - p)^2}{u}\right) = 0$$

#19: $v := 1 - \dfrac{h^2}{u}$

#20: $$\mathrm{COS}(\varepsilon) \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \mathrm{SIN}(\varepsilon) \cdot \left(\frac{h - p}{r} - i \cdot \mathrm{SIN}(\omega)\right) + \frac{h - p}{l} \cdot \left(\frac{h - p}{r} - i \cdot \mathrm{SIN}(\omega)\right) + t \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \left(v - \frac{(h - p)^2}{u}\right) = 0$$

#21: $w := h - p$

#22: $$\mathrm{COS}(\varepsilon) \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \mathrm{SIN}(\varepsilon) \cdot \left(\frac{w}{r} - i \cdot \mathrm{SIN}(\omega)\right) + \frac{w}{l} \cdot \left(\frac{w}{r} - i \cdot \mathrm{SIN}(\omega)\right) + t \cdot (i^2 \cdot (1 - \mathrm{COS}(\omega)) - s) + \left(v - \frac{w^2}{u}\right) = 0$$

#23: $m := i^2 \cdot (1 - \mathrm{COS}(\omega)) - s$

#24: $$\mathrm{COS}(\varepsilon) \cdot m + \mathrm{SIN}(\varepsilon) \cdot \left(\frac{w}{r} - i \cdot \mathrm{SIN}(\omega)\right) + \frac{w}{l} \cdot \left(\frac{w}{r} - i \cdot \mathrm{SIN}(\omega)\right) + t \cdot m + \left(v - \frac{w^2}{u}\right) = 0$$

#25: $g := \dfrac{w}{r}$

#26: $\cos(\varepsilon)\cdot m + \sin(\varepsilon)\cdot(g - i\cdot\sin(\omega)) + \frac{w}{l}\cdot(g - i\cdot\sin(\omega)) + t\cdot m + \left(v - \frac{w^2}{u}\right) = 0$

#27: $n := g - i\cdot\sin(\omega)$

#28: $\cos(\varepsilon)\cdot m + \sin(\varepsilon)\cdot n + \left(\frac{w}{l}\cdot n + t\cdot m + v - \frac{w^2}{u}\right) = 0$

#29: $L := \frac{w}{l}\cdot n + t\cdot m + v - \frac{w^2}{u}$

#30: $\cos(\varepsilon)\cdot m + \sin(\varepsilon)\cdot n + L = 0$

Unglaublich oder, das können wir jetzt lösen:

#31: $\varepsilon = -\frac{\pi\cdot\mathrm{SIGN}(m)}{2} - \mathrm{ASIN}\left(\frac{L}{\sqrt{(m^2 + n^2)}}\right) + \mathrm{ATAN}\left(\frac{n}{m}\right) \vee \varepsilon = -\frac{\pi\cdot\mathrm{SIGN}(m)}{2} + \mathrm{ASIN}\left(\frac{L}{\sqrt{(m^2 + n^2)}}\right) + \mathrm{ATAN}\left(\frac{n}{m}\right) - \pi \vee \varepsilon = -\frac{\pi\cdot\mathrm{SIGN}(m)}{2} + \mathrm{ASIN}\left(\frac{L}{\sqrt{(m^2 + n^2)}}\right) + \mathrm{ATAN}\left(\frac{n}{m}\right) + \pi$

Uns reicht der erste Ansatz, die Abschätzung von sign(m) war auch noch nervig also

#32: $\varepsilon = \frac{\pi}{2} - \mathrm{ASIN}\left(\frac{L}{\sqrt{(m^2 + n^2)}}\right) + \mathrm{ATAN}\left(\frac{n}{m}\right)$

na?
Falls du daran zweifelst das #30 = #9

#33: $\mathrm{SOLVE}(-\cos(\varepsilon)\cdot(\cos(\omega)\cdot(2\cdot l\cdot r\cdot\cos(\phi)^2\cdot\sin(\tau)^2 - 4\cdot l\cdot r\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) + 2\cdot l\cdot r\cdot\sin(\phi)^2\cdot\cos(\tau)^2) - 2\cdot l\cdot r\cdot\cos(\phi)^2\cdot\sin(\tau)^2 + 4\cdot l\cdot r\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) - 2\cdot l\cdot r\cdot\sin(\phi)^2\cdot\cos(\tau)^2 + 2\cdot l^2) + \sin(\varepsilon)\cdot(\sin(\omega)\cdot(2\cdot l\cdot r\cdot\cos(\phi)\cdot\sin(\tau) - 2\cdot l\cdot r\cdot\sin(\phi)\cdot\cos(\tau)) + 2\cdot l\cdot(h - p)) + \cos(\omega)\cdot(2\cdot r\cdot(l - r)\cdot\cos(\phi)^2\cdot\sin(\tau)^2 + 4\cdot r\cdot(r - l)\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) + 2\cdot r\cdot(l - r)\cdot\sin(\phi)^2\cdot\cos(\tau)^2) + \sin(\omega)\cdot(2\cdot r\cdot(h - p)\cdot\cos(\phi)\cdot\sin(\tau) + 2\cdot r\cdot(p - h)\cdot\sin(\phi)\cdot\cos(\tau)) + 2\cdot r\cdot(r -$

$$l)\cdot\cos(\phi)^2\cdot\sin(\tau)^2 + 4\cdot r\cdot(l - r)\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) + 2\cdot r\cdot(r -$$

$$l)\cdot\sin(\phi)^2\cdot\cos(\tau)^2 - 2\cdot h\cdot p + 2\cdot l^2 + p^2 = -$$

$$\cos(\varepsilon)\cdot(\cos(\omega)\cdot(2\cdot l\cdot r\cdot\cos(\phi)^2\cdot\sin(\tau)^2 - 4\cdot l\cdot r\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) +$$

$$2\cdot l\cdot r\cdot\sin(\phi)^2\cdot\cos(\tau)^2) + 2\cdot l\cdot r\cdot\cos(\phi)^2\cdot\cos(\tau)^2 +$$

$$4\cdot l\cdot r\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) + 2\cdot l\cdot r\cdot\sin(\phi)^2\cdot\sin(\tau)^2 + 2\cdot l\cdot(l - r)) +$$

$$\sin(\varepsilon)\cdot(\sin(\omega)\cdot(2\cdot l\cdot r\cdot\cos(\phi)\cdot\sin(\tau) - 2\cdot l\cdot r\cdot\sin(\phi)\cdot\cos(\tau)) + 2\cdot l\cdot(h - p)) +$$

$$\cos(\omega)\cdot(2\cdot r\cdot(l - r)\cdot\cos(\phi)^2\cdot\sin(\tau)^2 + 4\cdot r\cdot(r - l)\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau)$$

$$+ 2\cdot r\cdot(l - r)\cdot\sin(\phi)^2\cdot\cos(\tau)^2) + \sin(\omega)\cdot(2\cdot r\cdot(h - p)\cdot\cos(\phi)\cdot\sin(\tau) + 2\cdot r\cdot(p -$$

$$h)\cdot\sin(\phi)\cdot\cos(\tau)) + \cos(\phi)^2\cdot(2\cdot r\cdot(l - r)\cdot\cos(\tau)^2 + r^2) + 4\cdot r\cdot(l -$$

$$r)\cdot\sin(\phi)\cdot\cos(\phi)\cdot\sin(\tau)\cdot\cos(\tau) + \sin(\phi)^2\cdot(2\cdot r\cdot(l - r)\cdot\sin(\tau)^2 + r^2) - 2\cdot h\cdot p +$$

$$2\cdot l^2 - 2\cdot l\cdot r + p^2 + r^2, \varepsilon)$$

#34: true

q.e.d.
falls du’s nicht mehr weißt
quod erat demonstrandum

```
(*****************************************************************************
*
* (C) 2005 by HDSystems
*
* ---------------------------------------------------------------------------
* Module        : servoipc.pas

* Function      : servo-pwm with ipc@chip rtos
*
* Compiler      : Borland Pascal 7.0, Model large, integer 16bit
* Author        : Dieter Herz
* Date          : 30.01.06
* Version       : V1.00
* ---------------------------------------------------------------------------
* History       :
*
*  Vx.yy            Author  Changes
*
*****************************************************************************)
{ some important compiler settings d- r- s- for sc12 }
{$D-,R-,S-}
{$A+,B-,E-,F-,G+,I-,N-,O-,P-,Q-,T-,V-,X+}

unit servoipc;

interface

uses dos, hlapiipc;

const
  { swash plate math constants }
  Cpihalbe        = Pi / 2;
  Cradtoservo     = 128 / (Pi/4);
  Cservoanzahl    = 3;
  Cservoanzmax    = 4;
  Cservoringcount = 5;
  Cservostep      = Pi * 2 / Cservoanzahl;
  Cservoarmlaenge = 20;
  Ctaumradius     = 35;
  Cservotiefe     = 60;
  Cservoradius    = Ctaumradius - CServoarmlaenge;
  Causdrucks      = Cservoarmlaenge / Ctaumradius;
  Causdruckt      = Cservoradius / Cservoarmlaenge;
  Causdrucku      = 2 * Ctaumradius * Cservoarmlaenge;
  Causdruckv      = 1 - ( Cservotiefe * Cservotiefe / Causdrucku );
  Cmaxphi         = Pi;
  Cmaxomega       = 15.0 * Pi / 180;
  Cmaxpitch       = 5.0;

  { PIO }
  { PIOs for servos 7,8,9,10 #0000011110000000:
    Cpioservos = $0780;
    Cpioservo : array[1..Cservoanzmax] of byte = (7,8,9,10);  }
  { PIO numbering amd 23-20 for 7-10 with ipc ( > 16 so in pio-regs 1)
  #0000000011110000 }
  Cpioservos = $00F0;
  Cpioservo : array[1..Cservoanzmax] of byte = ($80,$40,$20,$10);

```

```
58
59    { PIO direct registers }
60    Cpmode0 = $FF70;                    { PIO Mode 0 Register }
61    Cpmode1 = $FF76;                    { PIO Mode 1 Register }
62    Cpdir0  = $FF72;                    { PIO Direction 0 Register }
63    Cpdir1  = $FF78;                    { PIO Direction 1 Register }
64    Cpdata0 = $FF74;                    { PIO Data Register 0 }
65    Cpdata1 = $FF7A;                    { PIO Data Register 1 }
66
67    { Timer direct registers }
68    Ct0con  = $FF56;                    { Timer 0 Mode/Control }
69    Ct1con  = $FF5E;                    { Timer 1 Mode/Control }
70    Ct2con  = $FF66;                    { Timer 2 Mode/Control }
71    Ct0cnt  = $FF50;                    { Timer 0 Count }
72    Ct1cnt  = $FF58;                    { Timer 1 Count }
73    Ct2cnt  = $FF60;                    { Timer 2 Count }
74    Ct0cmpa = $FF52;                    { Timer 0 Maxcount Compare A }
75    Ct0cmpb = $FF54;                    { Timer 0 Maxcount Compare B }
76    Ct1cmpa = $FF5A;                    { Timer 1 Maxcount Compare A }
77    Ct1cmpb = $FF5C;                    { Timer 1 Maxcount Compare B }
78    Ct2cmpa = $FF62;                    { Timer 2 Maxcount Compare A }
79    Ctcserv = $E000;                    { Timer - conf for servo #1110000000000000 }
80
81  type
82    Tservo  = 1..Cservoanzahl;          { servos }
83    Tservos = array[Tservo] of integer;
84    Tpulse  = record
85                length: integer;
86                pio   : byte;
87              end;
88    Tpulses = array[Tservo] of Tpulse;
89
90  const
91    Cservotimer = Ctimer0;
92    servocount : 1..Cservoringcount = Cservoringcount;
93    servostart : integer = 0;
94    servostops : integer = 0;
95    servobasis : integer = 6250;
96
97  var
98    servos: Tservos;
99    pulses: Tpulses;
100   servosin,servocos : array [1..Cservoanzahl] of real;
101   servoactiv, servovalid: boolean;
102
103 function arcsin(x: real): real;
104 function arccos(x: real): real;
105 procedure SinCos(x: real; var sinx, cosx: real);
106 function servoinit: integer;
107 procedure servodeinit;
108 procedure servoint;
109 procedure servocycle;
110 procedure swashinit;
111 procedure swashmath (TaumPhi,TaumOmega,TaumPitch: real);
112 procedure pulsemath;
113
114 implementation
115
```

```
116  function int2str (num: integer): string;
117    var
118      s: string;
119    begin
120      str (num, s);
121      int2str := s;
122    end;
123
124  function arcsin(x: real): real;
125    begin
126      if abs(x) = 1.0 then
127        arcsin := x * Cpihalbe
128      else
129        arcsin := ArcTan(x / sqrt(1.0 - x*x));
130      end;
131
132  function arccos(x: real): real;
133    begin
134      arccos := Cpihalbe - arcsin(x);
135    end ;
136
137  procedure SinCos(x: real; var sinx, cosx: real);
138    begin
139      sinx := Sin(x);
140      cosx := Cos(x);
141    end ;
142
143  { init the pwm on ipc@chip }
144  function servoinit: integer;
145    var
146      tempptr: pointer;
147    begin
148      swashmath(0,0,0);
149  {    pfeenpios(Cpioservos, Cpiooutlo);
150      halinittimer(Cservotimer, Ctconfsei, 0);
151      halinittimext(Cservotimer, Ctconfedd); }
152      portw[Cpdata1] := portw[Cpdata1] and not Cpioservos;
153      portw[Cpmode1] := portw[Cpmode1] or Cpioservos;
154      portw[Cpdir1]  := portw[Cpdir1] and not Cpioservos;
155      portw[Ct0cnt]  := 0;
156    end;
157
158  { deinit the pwm on ipc@chip }
159  procedure servodeinit;
160    begin
161      swashmath(0,0,0);
162    end;
163
164  { servo interupt called by timer after pulse-times reached }
165  procedure servoint;
166    begin
167      portw[Cpdata1] := portw[Cpdata1] and not pulses[servocount].pio;
168      inc(servostops);
169    end;
170
171  { servo cycle - must be called every 4ms from $af-interupt }
172  procedure servocycle;
173    begin
```

```
174       if servoactiv then
175         begin
176           if servocount < 5 then
177             inc(servocount)
178           else
179             servocount := 1;
180           if servocount <= Cservoanzahl then
181             begin
182               inc(servostart);
183               portw[Ct0cnt]  := 0;
184               portw[Ct0cmpa] := pulses[servocount].length;
185               portw[Cpdata1] := portw[Cpdata1] or pulses[servocount].pio;
186               portw[Ct0con]  := Ctcserv;
187             end;
188         end;
189     end;
190
191   { a few sin/cos for swash plate operation }
192   procedure swashinit;
193     var
194       snr : Tservo;
195     begin
196       for snr := 1 to Cservoanzahl do              { a few needed fixed sin/cos }
197         SinCos(Cpihalbe + (snr - 1) * Cservostep,servosin[snr],servocos[snr]);
198     end;
199
200   { swash plate mathematics - phi, omega and pitch in rad - servo}
201   procedure swashmath (TaumPhi,TaumOmega,TaumPitch: real);
202     var
203       Snr                        : Tservo;
204       TaumPhisin,TaumPhicos,
205       TaumOmegasin,TaumOmegacos,
206       Ausdruckw,Ausdruckg,
207       Mteil,Lteil1,Lteil2        : real;
208       Servoeps,I,L,M,N           : array [Tservo] of real;
209     begin
210       if abs(TaumOmega) > Cmaxomega then
211         begin
212           if TaumOmega < 0 then
213             TaumOmega := -Cmaxomega
214           else
215             TaumOmega :=  Cmaxomega;
216         end;
217       if abs(TaumPitch) > Cmaxpitch then
218         begin
219           if TaumPitch < 0 then
220             TaumPitch := -Cmaxpitch
221           else
222             TaumPitch :=  Cmaxpitch;
223         end;
224       SinCos(TaumPhi,TaumPhisin,TaumPhicos);
225       SinCos(TaumOmega,TaumOmegasin,TaumOmegacos);
226       Mteil     := 1 - TaumOmegacos;
227       Ausdruckw := Cservotiefe - TaumPitch;
228       Ausdruckg := Ausdruckw / Ctaumradius;
229       Lteil1    := Ausdruckw / Cservoarmlaenge;
230       Lteil2    := Causdruckv - ( Sqr(Ausdruckw) / Causdrucku );
231       for Snr := 1 to Cservoanzahl do
```

```
232         begin
233           I[Snr] := TaumPhisin*servocos[Snr] - TaumPhicos*servosin[Snr];
234           M[Snr] := Sqr(I[Snr]) * Mteil - Causdrucks;
235           N[Snr] := Ausdruckg - I[Snr] * TaumOmegasin;
236           L[Snr] := Causdruckt * M[Snr] + Lteil1 * N[Snr] + Lteil2;
237           Servoeps[Snr] := Cpihalbe - ArcSin (L[Snr]/Sqrt(Sqr(M[Snr])+Sqr(N[Snr])))
238                            + ArcTan (N[Snr]/M[Snr]);
239           servos[Snr] := Round (Servoeps[Snr] * Cradtoservo + 128)
240         end;
241       writeln('Servo1 ' + int2str(servos[1]));
242       writeln('Servo2 ' + int2str(servos[2]));
243       writeln('Servo3 ' + int2str(servos[3]));
244       pulsemath;
245     end;
246
247   { build the correct timing for the servo-pulses }
248   { servo-pulses range from 1 to 2 ms every 20ms at 5 mhz timer freq
249     and 1 ms in 1 byte resolution is roughly 4 µs per bit means 20 timer counts }
250   { we want half range so 10 timer counts and 1,25 ms offset }
251   { 1,2 ms offset may be better because of isr-delay - 5*1200}
252   procedure pulsemath;
253     var
254       snr : Tservo;
255     begin
256       for snr := 1 to Cservoanzahl do
257         begin
258           pulses[snr].length := servos[snr] * 10 + servobasis;
259           pulses[snr].pio := Cpioservo[snr];
260         end;
261     end;
262
263
264   end.
```