VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky

DIPLOMOVÁ PRÁCE

2009 **Petr Vojčinák**

VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra měřicí a řídicí techniky

Návrh a realizace robustního regulátoru pro laboratorní model vrtulníku s využitím platformy .NET Framework a prostředí MATLAB & Simulink + .NET Builder

Design and Realization of Robust Controller for Laboratory Model of the Helicopter with Use of .NET Framework, MATLAB & Simulink + .NET Builder

2009 **Petr Vojčinák**

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Prohlašuji, že

- jsem byl seznámen s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména §35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a §60 – školní dílo.

- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně ke své vnitřní potřebě diplomovou práci užít (§35 odst. 3).

- souhlasím s tím, že jeden výtisk diplomové práce bude uložen v Ústřední knihovně VŠB-TUO k prezenčnímu nahlédnutí a údaje o diplomové práci budou zveřejněny v informačním systému VŠB-TUO.

- beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

……

Petr Vojčinák

V Ostravě dne 7. 5. 2009

Poděkování

Touto cestou bych chtěl poděkovat vedoucímu mé diplomové práce Ing. Štěpánu Ožanovi PhD za cenné rady, připomínky, konzultace a laskavé zapůjčení modelu vrtulníku.

Nakonec děkuji své rodině za nesmírnou podporu a pochopení.

Abstrakt

Tato diplomová práce se zabývá problematikou návrhu H∞ robustního řízení laboratorního modelu vrtulníku CE 150 a jeho implementací na .NET platformu pomocí nástroje MATLAB Builder for .NET. Zpočátku se zaměříme na celkový matematický popis reálného modelu vrtulníku, tj. od teoretického modelu až po linearizovaný model. Další část se věnuje tématice robustního řízení a návrhů H∞ regulátorů pro řízení elevačního úhlu a azimutového úhlu. Závěrečná část práce je věnována nástroji MATLAB Builder for .NET a jeho implementací na cílovou platformu .NET Framework ve formě .NET komponenty a následné formulářové aplikace systému Windows.

Klíčová slova

vrtulník CE 150, MIMO systém, SISO systém, momentové rovnice, elevační úhel, azimutový úhel, těžiště, vazba, H∞ robustní řízení, norma, neurčitosti, robustnost a výkonnost, nominální soustava, rozšířená soustava, citlivostní funkce, váhové filtry, H∞ regulátor, MATLAB&Simulink, Robust Control Toolbox, .NET Framework, programovací jazyk C#, formuláře Windows, MATLAB Builder for .NET, MCR, .NET komponenta

Abstract

This diploma thesis is concerned with H∞ robust control problem of CE 150 helicopter laboratory model, and its .NET platform implementation using MATLAB Builder for .NET. At first, we target the general mathematical description of the helicopter model, i. e. from a theoretical model till a linearizated model. The next part of this thesis is devoted to robust control and H∞ controller design for elevation and azimuth angles control. The final part is focused on MATLAB Builder for .NET tool, and its target .NET Framework platform implementation in the form of a .NET component and a follow-on WinForms application.

Key words

CE 150 helicopter, MIMO system, SISO system, moment equation, elevation angle, azimuth angle, center of gravity, coupling, H∞ robust control, norm, uncertainties, robustness and performance, nominal plant, augmented plant, sensitivity functions, weighted filters, H∞ controller, MATLAB&Simulink, Robust Control Toolbox, .NET Framework, C# programming language, Windows Forms, MATLAB Builder for .NET, MCR, .NET component

Seznam použitých symbolů a zkratek

Symboly		
Označení	**Význam**	**Jednotka**
A	kvadraturní signál IRC senzoru	
$\hat{A}$	stavová matice systému neboli systémová matice	
A_n	n-tý člen Taylorova, resp. Maclaurinova rozvoje	
B	moment hybnosti způsobený viskózním třením	$[m^2 \cdot kg \cdot s^{-1}]$
B	kvadraturní signál IRC senzoru	
B_H	moment hybnosti v horizontální ose (vnitřní tření)	$[m^2 \cdot kg \cdot s^{-1}]$
B_M	moment hybnosti hlavního motoru (vnitřní tření)	$[m^2 \cdot kg \cdot s^{-1}]$
B_P	moment hybnosti vrtule při laminárním proudění	$[m^2 \cdot kg \cdot s^{-1}]$
B_T	moment hybnosti vedlejšího motoru (vnitřní tření)	$[m^2 \cdot kg \cdot s^{-1}]$
B_V	moment hybnosti ve vertikální ose (vnitřní tření)	$[m^2 \cdot kg \cdot s^{-1}]$
B_{φ_1}	moment hybnosti těla vrtulníku (bez uvažování směru)	$[m^2 \cdot kg \cdot s^{-1}]$
B_{φ_2}	moment hybnosti těla vrtulníku (s uvažováním směru)	$[m^2 \cdot kg \cdot s^{-1}]$
B_{ψ_1}	moment hybnosti těla vrtulníku (bez uvažování směru)	$[m^2 \cdot kg \cdot s^{-1}]$
B_{ψ_2}	moment hybnosti těla vrtulníku (s uvažováním směru)	$[m^2 \cdot kg \cdot s^{-1}]$
$\hat{B}$	vstupní matice systému neboli matice vstupů a stavů	
C	moment hybnosti způsobený Coulombovým třením	$[m^2 \cdot kg \cdot s^{-1}]$
$\hat{C}$	výstupní matice systému neboli matice stavů a výstupů	
D_P	moment setrvačnosti vrtule při turbulentním proudění	$[m^2 \cdot kg]$
$\hat{D}$	výstupní matice řízení neboli matice přímých vazeb	
$\hat{E}(s)$	Laplaceův obraz regulační odchylky	
F	síla	$[m \cdot kg \cdot s^{-2}]$
F_g	tíhová síla	$[m \cdot kg \cdot s^{-2}]$
$\boldsymbol{F}(j \cdot \omega), \boldsymbol{G}(j \cdot \omega)$	Fourierův obraz obecné přenosové funkce	
$\hat{G}(s)$	Laplaceův obraz přenosové funkce obecné soustavy	
$\hat{G}_0(s)$	Laplaceův obraz přenosové funkce nominální soustavy	
$\hat{G}_1(s)$	Laplaceův obraz přenosu hlavního motoru a vrtule	
$\hat{G}_2(s)$	Laplaceův obraz přenosu vedlejšího motoru a vrtule	
$\hat{G}_{azim.}(s)$	Laplaceův obraz přenosu v azimutu	
$\hat{G}_{elev.}(s)$	Laplaceův obraz přenosu v elevaci	
$\hat{G}_p(s)$	Laplaceův obraz přenosové funkce perturbované soustavy	
$\hat{G}_r(s)$	Laplaceův obraz přenosu reakční vazby elevace-azimut	
H_2	Hilbertův normovaný prostor, při $p = 2$	
H_∞	Banachův normovaný prostor, při $p = \infty$	
I_{MU}	interval škálovaných hodnot vstup. a výstup. signálů	$[MU]$
J	moment setrvačnosti	$[m^2 \cdot kg]$
J_H	moment setrvačnosti mech. části v horizontální ose	$[m^2 \cdot kg]$
J_M	moment setrvačnosti elevačního motoru	$[m^2 \cdot kg]$
J_T	moment setrvačnosti azimutového motoru	$[m^2 \cdot kg]$
J_V	moment setrvačnosti mech. části ve vertikální ose	$[m^2 \cdot kg]$
$J_{azim.}$	účelová funkce optimalizačního procesu v azimutu	
$J_{elev.}$	účelová funkce optimalizačního procesu v elevaci	
J_φ	celkový moment setrvačnosti těla vrtulníku v azimutu	$[m^2 \cdot kg]$
J_ψ	celkový moment setrvačnosti těla vrtulníku v elevaci	$[m^2 \cdot kg]$

Symboly		
Označení	**Význam**	**Jednotka**
K	zesílení výkonového zesilovače	$[V \cdot MU^{-1}]$
K_b	přístrojová konstanta stejnosměrného motoru	$[Wb]$
K_{gyro}	konstanta gyroskopické vazby elevace-azimut	$[N \cdot m \cdot s \cdot V^{-1}]$
$K_{gyro\ (emp.)}$	konstanta gyroskop. vazby v emp. modelu	$[N \cdot m \cdot s \cdot V^{-1}]$
K_i	proudová silová konstanta motoru	$[N \cdot m \cdot A^{-1}]$
K_{ω_1}	přístrojová vazba elevačního motoru	$[m^2 \cdot kg \cdot s]$
K_{ω_2}	přístrojová vazba azimutového motoru	$[m^2 \cdot kg \cdot s]$
$\widehat{K}(s)$	Laplaceův obraz přenosu H∞ robustního regulátoru	
L	moment hybnosti	$[m^2 \cdot kg \cdot s^{-1}]$
$\hat{L}(s)$	Laplaceův obraz přenosu otevřené smyčky	
M	moment síly	$[m^2 \cdot kg \cdot s^{-2}]$
M_1	moment síly vrtule hlavního (elevačního) motoru	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{1(emp.)}$	moment síly vrtule hlavního motoru v emp. modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_2	moment síly vrtule vedlejšího (azimutového) motoru	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{2(emp.)}$	moment síly vrtule vedlejšího motoru v emp. modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_H	celkový moment síly mech. části v horizontální ose	$[m^2 \cdot kg \cdot s^{-2}]$
M_{HM}	moment síly hlavní vrtule v horizontální ose	$[m^2 \cdot kg \cdot s^{-2}]$
M_{HT}	moment síly vedlejší vrtule v horizontální ose	$[m^2 \cdot kg \cdot s^{-2}]$
M_P	celkový moment síly způsobený rotací vrtule	$[m^2 \cdot kg \cdot s^{-2}]$
M_V	celkový moment síly mech. části ve vertikální ose	$[m^2 \cdot kg \cdot s^{-2}]$
M_{VM}	moment síly hlavní vrtule ve vertikální ose	$[m^2 \cdot kg \cdot s^{-2}]$
M_{VT}	moment síly vedlejší vrtule ve vertikální ose	$[m^2 \cdot kg \cdot s^{-2}]$
M_g	moment tíhové síly hmotného bodu těla vrtulníku	$[m^2 \cdot kg \cdot s^{-2}]$
M_{g_1}	amplituda momentu síly hmotného bodu	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{g(emp.)}$	moment tíhové síly hmotného bodu v emp. modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_{gyro}	moment gyroskopické síly	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{gyro\ (emp.)}$	moment gyroskopické síly v emp. modelu	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{gyro\ (lin.,emp.)}$	moment gyroskop. síly v emp. linearizovaném modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_{f_1}	moment třecí síly (vnitřní a Coulomb. tření) v elevaci	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{f_1(lin.)}$	moment třecí síly v linearizovaném elevačním modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_{f_2}	moment třecí síly (vnitřní a Coulomb. tření) v azimutu	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{f_2(lin.)}$	moment třecí síly v linearizovaném azimutovém modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_i	moment síly způsobený budicím napětím	$[m^2 \cdot kg \cdot s^{-2}]$
M_{ib}	moment síly způsobený protinapětím	$[m^2 \cdot kg \cdot s^{-2}]$
M_{m_1}	moment tíhové síly hmotného bodu vrtulníku	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{m_1(emp.)}$	moment tíhové síly hmotného bodu v emp. modelu	$[m^2 \cdot kg \cdot s^{-2}]$
M_{m_2}	moment tíhové síly ocasního závaží	$[m^2 \cdot kg \cdot s^{-2}]$
$M_n(x)$	Maclaurinův rozvoj n-tého řádu	
M_r	moment reakční síly hlavního motoru	$[m^2 \cdot kg \cdot s^{-2}]$
$M_{r(emp.)}$	moment reakční síly hlavního motoru v emp. modelu	$[m^2 \cdot kg \cdot s^{-2}]$
$\widehat{M}_{r(emp.)}(s)$	Laplaceův obraz momentu reakční síly v emp. modelu	
$M_{\dot{\varphi}}$	moment dostředivé síly	$[m^2 \cdot kg \cdot s^{-2}]$
M_ω	moment síly způsobený třením	$[m^2 \cdot kg \cdot s^{-2}]$
N	celočíselná konstanta udávající horní mez sumy	
R	elektrický odpor kotvy (rotoru) stejnosměrného motoru	$[\Omega]$
$\hat{R}(s)$	Laplaceův obraz citlivosti řízení	
$R_n(x)$	Langrangeův zbytek funkce $f(x)$ po n-tém členu rozvoje	
$\hat{S}(s)$	Laplaceův obraz citlivostní funkce (přenos odchylky)	

Symboly		
Označení	**Význam**	**Jednotka**
T_1	časová konstanta dynamiky hlavní motor-vrtule	$[s]$
$T_1(\psi)$	lineární aproximace funkce $g(\psi)$	
$T_1(\psi)_{\psi_0}$	lineární aproximace funkce $g(\psi)$ v pracovním bodě ψ_0	
T_2	časová konstanta dynamiky vedlejší motor-vrtule	$[s]$
$T_n(x)$	Taylorův rozvoj n-tého řádu	
T_{r_1}	první časová konstanta čitatele reakčního přenosu	$[s]$
T_{r_2}	druhá časová konstanta čitatele reakčního přenosu	$[s]$
$\hat{T}(s)$	Laplaceův obraz kompl. citlivostní funkce (přenos řízení)	
U	stejnosměrné elektrické napětí	$[V]$
$\hat{U}(s)$	Laplaceův obraz akční veličiny	
$U_{1(mot.)}$	výstupní napětí hlavního (elevačního) motoru	$[MU]$
$\hat{U}_{1(mot.)}(s)$	Laplaceův obraz výstupního napětí hlavního motoru	
$U_{1(reak.)}$	výstupní reakční napětí hlavního (elevačního) motoru	$[MU]$
$U_{2(mot.)}$	výstupní napětí vedlejšího (azimutového) motoru	$[MU]$
$\hat{U}_{2(mot.)}(s)$	Laplaceův obraz výstupního napětí vedlejšího motoru	
$U_B = u_3(t)$	stejnosměrné napětí pro nastavení pozice těžiště	$[V]$
U_{B0}	stejnoměrné napětí pro nastavení ref. pozice těžiště	$[V]$
$U_{B_{max}}$	napětí pro nastavení maximální vzdálenosti těžiště	$[V]$
$U_{B_{min}}$	napětí pro nastavení minimální vzdálenosti těžiště	$[V]$
$U_M = u_1(t)$	stejnosměrné napětí elevačního motoru	$[V]$
$\hat{U}_M(s)$	Laplaceův obraz napětí elevačního motoru	
$\boldsymbol{U}_M$	jednorozměrný vektor napětí hlavního motoru	$[V]$
$\hat{U}_r(s)$	Laplaceův obraz výstupního reakčního napětí	
$U_T = u_2(t)$	stejnosměrné napětí azimutového motoru	$[V]$
$\hat{U}_T(s)$	Laplaceův obraz napětí azimutového motoru	
$\boldsymbol{U}_T$	jednorozměrný vektor napětí vedlejšího motoru	$[V]$
$\hat{W}_{cmd}(s)$	Laplaceův obraz přenosu váhového filtru reference	
$\hat{W}_d(s)$	Laplaceův obraz přenosu váhového filtru poruchy	
$\hat{W}_e(s)$	Laplaceův obraz přenosu váhového filtru odchylky	
$\hat{W}_{noise}(s)$	Laplaceův obraz přenosu váhového filtru šumu senzorů	
$\hat{W}_u(s)$	Laplaceův obraz přenosu váhového filtru akční veličiny	
$X1$	detekce čela nebo týlu pouze jednoho kvadratur. signálu	
$X2$	detekce čela a týlu pouze jednoho kvadratur. signálu	
$X4$	detekce čela a týlu obou kvadraturních signálů	
$\hat{Y}(s)$	Laplaceův obraz výstupní veličiny	
$\hat{Y}_\varphi(s)$	Laplaceův obraz výstupní hodnoty úhlu azimutu	
$\hat{Y}_\psi(s)$	Laplaceův obraz výstupní hodnoty úhlu elevace	
a	šířka tělesa těžiště	$[m]$
a_1	přepočtová konstanta kvadratického členu (elevace)	$[N \cdot m \cdot MU^{-2}]$
a_2	přepočtová konstanta kvadratického členu (azimut)	$[N \cdot m \cdot MU^{-2}]$
a_r	reakční přepočtová konst. kvadr. členu (elevace-azimut)	$[N \cdot m \cdot MU^{-2}]$
b	délka tělesa těžiště	$[m]$
b_1	přepočtová konstanta lineárního členu (elevace)	$[N \cdot m \cdot MU^{-1}]$
b_2	přepočtová konstanta lineárního členu (azimut)	$[N \cdot m \cdot MU^{-1}]$
b_r	reakční přepočtová konst. lin. členu (elevace-azimut)	$[N \cdot m \cdot MU^{-1}]$
c	výška tělesa těžiště	$[m]$
e	Eulerovo číslo ($e = 2{,}718281828459\ldots$)	
$e(t)$	charakteristika regulační odchylky v čase	

Symboly		
Označení	**Označení**	**Jednotka**
$f_1(t)$	závislost počtu kroků IRC senzoru na U_M v ust. stavu	$[MP]$
$f_2(t)$	závislost počtu kroků IRC senzoru na U_T v ust. stavu	$[MP]$
$f(x)$	obecná matematická funkce nezávislé proměnné x	
$f^{(n)}(x = x_0)$	n-tá derivace funkce v pracovním bodě x_0	
g	tíhové zrychlení zemské ($g = 9{,}80665\ [m \cdot s^{-2}]$)	$[m \cdot s^{-2}]$
$g(\psi) = sin(\psi)$	nelineární funkce určená k linearizaci v elevaci	
$g^{(n)}(\psi)$	n-tá derivace nelineární funkce určené k linearizaci	
$g^{(n)}(\psi)$	n-tá derivace nelineární funkce určené k linearizaci	
h_{PC}	funkční hodnota výstupního signálu z PC	$[MU]$
$i = i(t)$	okamžitá hodnota elektrického proudu v čase	$[A]$
i	řádkový index v matici	
j	sloupcový indešx v matici	
j	imaginární jednotka $j = \sqrt{-1}$	
k	celočíselná konstanta; index sumačního členu	
k_1	přepočtová konstanta elevačního IRC senzoru	$[rad \cdot MP^{-1}]$
k_{1HM}	moment hybnosti hlavní vrtule v horizontální ose	$[m^2 \cdot kg \cdot s^{-1}]$
k_{1HT}	moment hybnosti vedlejší vrtule v horizonální ose	$[m^2 \cdot kg \cdot s^{-1}]$
k_{1VM}	moment hybnosti hlavní vrtule ve vertikální ose	$[m^2 \cdot kg \cdot s^{-1}]$
k_{1VT}	moment hybnosti vedlejší vrtule ve vertikální ose	$[m^2 \cdot kg \cdot s^{-1}]$
k_2	přepočtová konstanta azimutového IRC senzoru	$[rad \cdot MP^{-1}]$
k_{2HM}	moment setrvačnosti hlavní vrtule v horizontální ose	$[m^2 \cdot kg]$
k_{2HT}	moment setrvačnosti vedlejší vrtule v horizontální ose	$[m^2 \cdot kg]$
k_{2VM}	moment setrvačnosti hlavní vrtule ve vertikální ose	$[m^2 \cdot kg]$
k_{2VT}	moment setrvačnosti vedlejší vrtule ve vertikální ose	$[m^2 \cdot kg]$
k_B	přenosová konstanta servomechanismu	$[N \cdot m \cdot V^{-1}]$
k_M	přenosová konstanta hlavního (elevačního) motoru	$[N \cdot m \cdot V^{-1}]$
k_T	přenosová konstanta hlavního (azimutového) motoru	$[N \cdot m \cdot V^{-1}]$
k_φ	přepočtová konstanta azimutového IRC senzoru	$[rad \cdot MP^{-1}]$
k_ψ	přepočtová konstanta elevačního IRC senzoru	$[rad \cdot MP^{-1}]$
l_0	vzdálenost mezi středy těžiště a pomocného závaží	$[m]$
l_1	vzdálenost mezi středy těžiště a hmotného bodu	$[m]$
$l_2 = \Delta l_2$	délkový přírůstek od středu těžiště v závislosti na U_B	$[m]$
$l_{2(FS)}$	velikost intervalu posunu těžiště	$[m]$
$l_{2(ref.)}$	referenční vzdálenost středu těžiště v závislosti na $U_{B_{min}}$	$[m]$
m	dimenze normovaného vektorového prostoru	
m_1	hmotnost hmotného bodu těla vrtulníku	$[kg]$
m_2	hmotnost ocasního závaží	$[kg]$
m_t	hmotnost nastavitelného těžiště vrtulníku	$[kg]$
n	počet průhledných a neprůhledných rysek IRC kotouče	
n	exponent mocninné řady; argument faktoriálu	
p	mocnina vyjadřující příslušnou normu prostoru	
q	elektrický náboj	$[C]$
$r(t) = w(t)$	charakteristika řídicí veličiny neboli reference v čase	
t	spojitý čas (tempus)	$[s]$
u	vstupní řídicí (budicí) napětí motorů v elevaci a azimutu	$[V]$
$u(t)$	charakteristika akční veličiny v čase	
$u_{1(mot.)}(t)$	okamžitá hodnota výstupního napětí hlavního motoru	$[MU]$
$u_{2(mot.)}(t)$	okamžitá hodnota výstupního napětí vedlejšího motoru	$[MU]$
$u_{výst.}$	budicí napětí stejnosměrného motoru	$[V]$

Symboly		
Označení	**Označení**	**Jednotka**
u_b	elektromotorické napětí neboli zdroj protinapětí	$[V]$
x	nezávislá proměnná	
x_0	obecný pracovní bod neboli střed konvergence řady	
$\boldsymbol{x}$	obecný vektor	
y_{model}	odezva modelu reálného systému v elevaci nebo azimutu	$[rad]$
$y_{reál.}$	odezva reálného systému v elevaci nebo azimutu	$[rad]$
y_φ	funkční hodnota výstupního úhlu azimutu	$[rad]$
y_{φ_0}	ofset azimutového IRC senzoru při $y_\varphi = 0\ [rad]$	$[rad]$
y_ψ	funkční hodnota výstupního úhlu elevace	$[rad]$
y_{ψ_0}	ofset elevačního IRC senzoru při $y_\psi = 0\ [rad]$	$[rad]$
$y(t)$	charakteristika výstupní veličiny v čase	
Δ	obecný přírůstek; nestrukturovaná neurčitost (deltablok)	
Φ	magnetický indukční tok	$[Wb]$
α	úhlové zrychlení	$[rad \cdot s^{-2}]$
β	úhel posunutí prvního a druhého segmentu IRC senzoru	$[rad]$
δ	rovinný úhel	$[rad]$
ξ	obecný bod, v němž je splněna rovnice $F^{(1)}(\xi) = 0$	
π	Ludolfovo číslo ($\pi = 3{,}141598265359\ldots$)	
φ	úhel azimutu	$[rad]$
φ_{FS}	rozsah úhlu azimutu	$[rad]$
φ_{max}	maximální velikost úhlu azimutu	$[rad]$
φ_{min}	minimální velikost úhlu azimutu	$[rad]$
$\dot{\varphi}$	první derivace úhlu azimutu podle času	$[rad \cdot s^{-1}]$
$\ddot{\varphi}$	druhá derivace úhlu azimutu podle času	$[rad \cdot s^{-2}]$
ψ	úhel elevace	$[rad]$
ψ_{FS}	rozsah úhlu elevace	$[rad]$
ψ_{max}	maximální velikost úhlu elevace	$[rad]$
ψ_{min}	minimální velikost úhlu elevace	$[rad]$
ψ_0	významný úhel elevace coby pracovní bod	$[rad]$
ψ_1	pomocný úhel elevace	$[rad]$
ψ_2	pomocný úhel elevace	$[rad]$
$\dot{\psi}$	první derivace úhlu elevace podle času	$[rad \cdot s^{-1}]$
$\ddot{\psi}$	druhá derivace úhlu elevace podle času	$[rad \cdot s^{-2}]$
ω	úhlová rychlost, úhlový kmitočet	$[rad \cdot s^{-1}]$
$\dot{\omega}$	první derivace úhlové rychlosti podle času	$[rad \cdot s^{-2}]$
ω_1	úhlová rychlost elevačního motoru	$[rad \cdot s^{-1}]$
ω_2	úhlová rychlost azimutového motoru	$[rad \cdot s^{-1}]$
$\omega_H = \dot{\psi}$	úhlová rychlost v horizontální ose	$[rad \cdot s^{-1}]$
ω_{H0}	úhlová rychlost v horizontální ose v ustáleném stavu	$[rad \cdot s^{-1}]$
$\dot{\omega}_H = \ddot{\psi}$	první derivace úhlové rychlosti podle času v horizont. ose	$[rad \cdot s^{-2}]$
ω_M	úhlová rychlost elevačního motoru	$[rad \cdot s^{-1}]$
ω_{M0}	úhlová rychlost elevačního motoru v ustáleném stavu	$[rad \cdot s^{-1}]$
$\dot{\omega}_M$	první derivace úhlové rychlosti elevačního motoru	$[rad \cdot s^{-2}]$
$\omega_V = \dot{\varphi}$	úhlová rychlost ve vertikální ose	$[rad \cdot s^{-1}]$
ω_{V0}	úhlová rychlost ve vertikální ose v ustáleném stavu	$[rad \cdot s^{-1}]$
$\dot{\omega}_V = \ddot{\varphi}$	první derivace úhlové rychlosti podle času ve vertikál. ose	$[rad \cdot s^{-2}]$
ω_T	úhlová rychlost azimutového motoru	$[rad \cdot s^{-1}]$
ω_{T0}	úhlová rychlost azimutového motoru v ustáleném stavu	$[rad \cdot s^{-1}]$
$\dot{\omega}_T$	první derivace úhlové rychlosti azimutového motoru	$[rad \cdot s^{-2}]$

Symboly	
Označení	**Význam**
API	aplikační programovací rozhraní
ATL	druh pracovního rámce
BCL	základní knihovna tříd platformy .NET
CDS	společný datový systém platformy .NET
CIL	společný intermediální jazyk
CLR	společný běhový systém platformy .NET
CLS	společná specifikace jazyků platformy .NET
CTS	společný systém typů platformy .NET
COM	komponentní objektový model
CM	správa kódu platformy .NET
CUI	konzolové uživatelské rozhraní
C#	primární programovací jazyk platformy .NET
DE	ladicí nástroje platformy .NET
DLL	dynamicky linkovatelná knihovna
ECMA	nezisková organizace specifikující platformu .NET
GUI	grafické uživatelské rozhraní
IDL	jazyk identifikace rozhraní
IEC	Mezinárodní elektrotechnická komise
IL	intermediální jazyk
IRC	inkrementální rotační senzor
ISO	Mezinárodní organizace pro normalizaci
JIT	druh komplilace platformy .NET
LFT	lineární fraktální transformace – druh popisu neurčitosti
LQG	lineárně-kvadratické gaussovské řízení
LQR	lineárně-kvadratické řízení
LTR	metoda návrhu robustního řízení
MATLAB	komplexní programový nástroj určený návrh, analýzu, tvorbu, simulaci, modelování dynamických systémů, fyzikálních jevů, systémů reálného času, řídicích a komunikačních systémů
MCR	běhové prostředí pro aplikace spolupracující s MATLABem
MFC	druh pracovního rámce
MGF	prostředí pro práci s grafy v prostředí MATLAB
MIMO	systém s více vstupy a více výstupy
MM	správa paměti platformy .NET
.NET	platforma .NET Framework umožňuje nejen budování aplikací na rodině systému Windows, ale také na jiných operačních systémech
OLC	řízení života objektů platformy .NET
OOP	objektově orientované programování
PE	formát .NET binární jednotky operačního systému Windows
QFT	metoda návrhu robustního řízení
SISO	systém s jedním vstupem a jedním výstupem
STL	druh pracovního rámce
UI	uživatelské rozhraní platformy .NET
VB6	programovací jazyk Visual Basic 6.0 platformy .NET
VES	virtuální vykonávací systém
WinForms	formuláře operačního systému Windows
XML	rozšířitelný značkovací jazyk

Obsah

1 Úvod

Z fyzikálního hlediska je moderní helikoptéra velice složitý systém. Abychom pochopili jeho funkčnost, můžeme se jej pokusit namodelovat. Vycházíme z předpokladu, že helikoptéra a její model mají společné rysy, např. dynamika chování vrtulí nebo těla vrtulníku v prostoru, vyjádřené konkrétními fyzikálními veličinami (v tomto případě nás budou zajímat elevace a azimut, respektive úhel elevace a úhel azimutu).

Chceme-li, aby se model choval definovaným způsobem, musíme vytvořit řídicí systém, jímž za daných podmínek ovlivníme klíčové parametry modelu, neboli ovlivňujeme jeho chování. V teorii automatického řízení takový řídicí systém představuje vhodný typ regulátoru.

Určitě by bylo vhodné, kdyby nějaký uživatel coby pilot v kabině mohl takový model relevantně řídit. Výsledkem pak může být aplikace s uživatelským rozhraním, založené na určité platformě a určené pro počítačovou architekturu typu PC.

Kapitola 2 – Model vrtulníku CE 150

Úvod této kapitoly se věnuje obecné problematice reálných vrtulníků, tj. směrové kontrole při letu a principu stability vrtulníku v závislosti na jeho konstrukčním uspořádání. Druhou část kapitoly tvoří kompletní matematicko-fyzikální popis modelu vrtulníku CE 150 od firmy Humusoft s. r. o., zabývající se:

- **fyzikálním modelováním úlohy** – teoretický model (elevace, azimut, stejnosměrný motor, vrtule, fyzikální rozměry klíčových veličin)
- **identifikací modelu** – empirický model s ukázkami postupů identifikace některých významných parametrů (konverzní relace mezi fyzikálními veličinami a číslicově zpracovávanými signály, kalibrace čidel a jejich význam, typické konstanty v elevaci, azimutu, linearizované soustavě stejnosměrného motoru a vrtule, těle vrtulníku, význam těžiště)
- **linearizací dílčích částí modelu** – na základě empirického modelu odvození linearizovaných modelů a jejich přenosových funkcí (elevace, azimut), odvození přenosových funkcí statických lineárních systémů
- **srovnáním modelů –** komparace mezi matematickým modelem a reálným modelem

Cílem této kapitoly je nejen pochopení fyzikálního principu modelu, ale také detailní odvození příslušného matematického popisu, založeného převážně na rovnováze momentů sil, respektive momentových rovnicích.

Kapitola 3 – Robustní řízení

Tato kapitola se zabývá nejen teoretickou podstatou tohoto přístupu návrhu regulátoru z oblasti moderní teorie řízení, ale také praktickým návrhem. Protože oblast robustního řízení je velice obsáhlá, uvádíme pouze nezbytné matematické definice (teoretický význam norem signálů v časové oblasti a přenosových kmitočtových funkcí a jejich praktické využití v oblasti řízení, dále pak Riccatiho rovnice nebo Hamiltonovy matice), terminologii (význam robustnosti a výkonnosti, neurčitosti systému) a metodiku, v níž jsou uvedeny významné metody robustního návrhu a koncepce vyšetřování robustní stability (např. Charitonovovův teorém). Hlavní důraz je kladen na metodu robustního řízení typu H∞, konkrétně pak problémem smíšené citlivosti (Mixed Sensitivity Problem).

Druhá část kapitoly se věnuje samotnému návrhu robustního regulátoru, kdy v rámci zvolené koncepce diskutujeme otázku autonomnosti a celkové konfigurace systému s robustním regulátorem a rozšířenou soustavou (nominální soustava a váhové filtry) – oproti klasické smíšené citlivosti penalizujeme také vstupní signály. Návrh vychází z odvozených přenosových funkcí, kdy v případě matematického modelu vypočteme přenosovou funkci regulátoru, kterou následně aplikujeme na reálnou nominální soustavu a váhové filtry v odpovídajících tvarech. Dynamické odezvy obou systémů se posléze porovnají při shodných podmínkách, aby nedošlo ke zkreslení výsledků.

Cíl této kapitoly spočívá v seznámení se s problematikou robustního řízení, a to jak po stránce teoretické, tak rovněž po stránce praktické.

Kapitola 4 – Filozofie .NET

Úvod této kapitoly se věnuje obecnému nástinu problematiky softwarového inženýrství před příchodem platformy .NET Framework. Druhá část kapitoly se již plně věnuje filozofii této platformy, tj. popisuje její strukturu jako kompaktní celek, základní stavební kameny (CLR, CTS, CLS), vnitřní strukturu (distribuční jednotky, jmenné prostory a datové typy) a zmiňuje možnosti nezávislosti platformy .NET. Ve třetí části se krátce představuje hlavní programovací jazyk této platformy – C#. Závěrečná část kapitoly se grafickou stránkou klienta platformy .NET – formuláře Windows neboli WinForms a jejich standardní, ale také vlastní ovládací prvky, roztříděné do skupin podle příbuznosti.

Cíl této kapitoly spočívá v detailnějším se seznámení s platformou .NET Framework, tj. její filozofií, strukturou a terminologií, kterou s výhodou využíváme v následující kapitole.

Kapitola 5 – MATLAB Builder for. NET

Úvod této kapitoly se krátce zabývá propojením vývojového prostředí MathWorks MATLAB s platformou .NET Framework prostřednictvím nástroje MATLAB Builder for .NET, respektive v novějších verzích MATLAB Builder NE, přičemž výčet diskutovaných úloh ve zkratce demonstruje jeho aplikační možnosti, tj. od řešení různých jednodušších a složitějších matematických úloh až po implementaci v konzolových aplikacích a aplikacích s formuláři Windows.

Druhá část kapitoly se věnuje samotné WinForms aplikaci s názvem Helikoptéra CE 150, tj.:

- **vytvoření M-souborů** – obsahuje matlabovský kód pro výpočet H∞ norem klíčových přenosů citlivostních funkcí, regulátoru a soustavy
- **vytvoření .NET komponent** – zdrojové M-soubory jsou přetransformovány pomocí nástroje MATLAB Builder NE do uživatelské .NET komponenty
- **volání funkcí .NET komponenty** – detailní popis provázání datových typů MW z knihovny ***MWArray.dll*** a double platformy .NET s obsahem uživatelské .NET komponenty ve formě procedur jazyka C#
- **volání procedur na úrovni formuláře** – popis provázání procedur jazyka C# s prvky formulářů Windows pomocí událostí
- **graficko-uživatelské rozhraní aplikace** – vzhled formuláře a popis aplikační logiky s důrazem na ověřování validity vstupních dat pro relevantní výpočet H∞ norem

2 Model vrtulníku CE 150

2.1 Úvod do problematiky

Helikoptéry se mohou vznášet nad zemí i bez jakéhokoliv pohybu vpřed. Potřebné nadnášení způsobuje rotace vrtule. Když se helikoptéra vznáší, je velikost vztlakové síly rovna velikosti gravitační síly působící na helikoptéru. Aby se tělo vrtulníku mohlo vznést a stoupat, musí být vztlaková síla větší než síla gravitační, čehož se dosahuje zvětšením úhlu náběžných hran listů vrtule. Při klesání se úhel náběžných hran zmenší. [1]

2.1.1 Směrová kontrola

Má-li se helikoptéra pohybovat vpřed, rotor se natočí nepatrně směrem dopředu tak, aby vyvolal vztlak a zároveň ji poháněl v horizontálním směru. Aby mohlo dojít k natočení rotoru dopředu, úhel náběžných hran listů vrtule se při poloze zvyšuje, v opačné poloze zase snižuje. [1]

Při pohybu helikoptéry do stran nebo dozadu se uplatňuje mechanismus náklonu rotoru i v jiných směrech, například při pohybu vpravo by měl vztlak na levé straně vrtulníku vzrůstat a na pravé klesat tak, aby se rotor natočil směrem vpravo. [1]

2.1.2 Stabilita

Helikoptéra vybavená jedním motorem by se při vznášení ve vzduchu pomalu otáčela. Dle třetího Newtonova pohybového zákona, tj. zákon akce a reakce, odpovídá každé akci reakce stejné velikosti, ovšem opačného směru – tj. působí-li motor helikoptéry na rotor silou (akce), rotor zpětně působí na helikoptéru (reakce). Protože helikoptéra má podstatně vyšší hmotnost než rotor, bude se pohybovat jen pomalu, zato moment otáčení rotoru bude velikostně odpovídat protisměrně působícímu momentu otáčení vrtulníku. [1]

U většiny helikoptér se využívá jediného hlavního (elevačního) motoru a dalšího menšího vyvažovacího (azimutového) motoru, jehož listy leží ve svislé rovině kolmo na vodorovnou rovinu listů hlavního motoru. Rotory vytvářejí horizontálně působící sílu, která eliminuje stáčivý efekt hlavního rotoru, čímž dochází ke stabilizaci helikoptéry. [1]

Jiná konstrukce je založená na dvou rotorech otáčejících se opačnými směry, čímž vzniká jednak vztlak, ale zároveň se eliminují síly způsobující stáčení helikoptéry. Oproti klasickému uspořádání mají tyto vrtulníky jednu velikou výhodu, tj. dokáží transportovat náklad zavěšený na více místech helikoptéry – např. tzv. „létající banán“ (Flying Banana). To je způsobeno rovnoměrným rozmístěním rotorů, přičemž přední rotor bývá posazen o něco níže než zadní rotor. Nevýhoda spočívá v komplikovanější konstrukci ovládání obou rotorů. [1]

2.2 Popis modelu

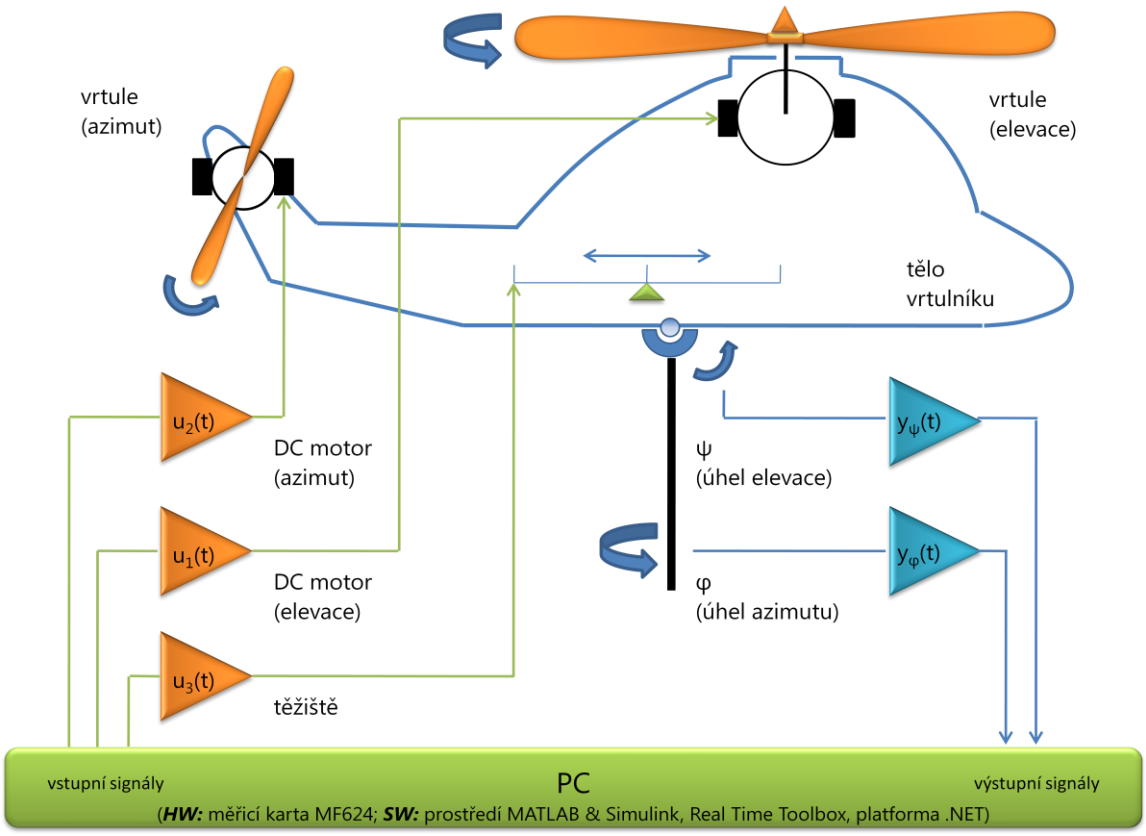
Model vrtulníku CE 150 je přípravek určený pro teoretické studium a praktický výzkum nejen základních, ale také složitějších principů v oblasti řízení, přičemž se lze zabývat následujícími problematikami, kupříkladu dle [1]:

- dynamické modelovaní
- identifikace parametrů modelu
- analýza naměřených charakteristik

- návrhy různých regulačních systémů pomocí klasických i moderních metod řízení

Popisovaný model vrtulníku je dynamický systém s více proměnnými s až třemi nastavitelnými vstupy a dvěmi měřícími výstupy. Všechny vstupy a výstupy jsou provázané. Systém je v podstatě nelineární a přinejmenším šestého řádu v závislosti na přesnosti modelování. Matematický model lze linearizovat v okolí pracovního bodu. [1] [5]

Model vrtulníku se skládá ze dvou stejnosměrných motorů, nesoucích tělo helikoptéry a ovládajících vrtule, tudíž tělo vrtulníku disponuje dvěma stupni volnosti. Osy rotace těla vrtulníku jsou kolmé stejně jako osy rotace motorů. Úhel azimutu (horizontální osa rotace) a úhel elevace (vertikální osa rotace) jsou současně ovlivňovány točením vrtulí. Stejnosměrné motory pro jejich ovládání jsou řízeny v závislosti na výstupním signále z PC, přičemž těžiště je ovládáno servomechanismem (viz obr. 2.2.1) dle [1] [5].



Obr. 2.2.1: blokové schéma modelu vrtulníku CE 150

2.2.1 Aplikační možnosti

Jak jsme již uvedli, model slouží a priori jako výuková pomůcka, tudíž možnosti jejího využití by měly postihnout danou problematiku v co možná největší míře, tedy:

- přímé odvození obecného matematického modelu helikoptéry pomocí Lagrangeových diferenciálních rovnic s následnou linearizací a zjednodušením
- přímá identifikace klíčových parametrů linearizovaného modelu
- popis systému pomocí přenosové matice a stavového prostoru
- porovnání metodiky návrhu spojitého a diskrétního regulátoru s optimální volbou vzorkovací frekvence

- návrhy zpětnovazebního regulátoru nebo zpětnovazebního regulátoru s pozorovatelem stavů
- návrh robustního a adaptivního řízení pro změnu polohy těžiště
- návrh regulátorů z oblasti moderní teorie řízení – LQR, LQG nebo H∞

2.2.2 Modelování

Pokus namodelovat detailní dynamiku systému vede na extrémně složitý, nečitelný a nepoužitelný model. Naším úkolem bude aplikace tzv. inženýrského přístupu, tj. schopnost najít důležité vlastnosti modelu a podmínky, za jakých může daný model správně fungovat. Zkoumaný systém pracuje v závislosti na určitých podmínkách, přičemž ne všechny dynamické vlastnosti jsou použity, což značně ulehčuje odvození matematického aparátu pro daný model. [1]

Při návrhu lze využít řadu, jak získat popis modelu, např. dle [1]:

- metody systematického modelování, založené na variačním přístupu (např. Langrangeovy rovnice)
- metoda přímého odvození modelu pomocí rovnováhy sil

Obě metody jsou popsány nelineárními diferenciálními rovnicemi. Při modelování chování modelu budeme dále využívat metodu založenou na rovnováze momentů sil. Pomocí modelování vytváříme teoretický model. [1]

Jednotky fyzikálních veličin

Pro lepší představu je vhodné uvádět hodnoty veličin ve fyzikálních jednotkách, tj. striktně používat základní i odvozené jednotky soustavy SI. Pro přehlednost a snadnou orientaci v následujícím výkladu je vhodné uvést některé konverzní vztahy mezi základními a odvozenými jednotkami mechanických, elektrických a magnetických veličin (viz tab. 2.2.1a) dle [2].

Mechanické veličiny			
Název veličiny	**Označení veličiny**	**Jednotka (odvozená)**	**Jednotka (SI)**
moment setrvačnosti	J	$m^2 \cdot kg$	$m^2 \cdot kg$
moment hybnosti	L	$N \cdot m \cdot s$	$m^2 \cdot kg \cdot s^{-1}$
moment síly	M	$N \cdot m$	$m^2 \cdot kg \cdot s^{-2}$
síla, tíha	F, F_g	N	$m \cdot kg \cdot s^{-2}$
úhel rovinný	δ	rad	1
úhlová rychlost	ω	$rad \cdot s^{-1}$	s^{-1}
úhlové zrychlení	α	$rad \cdot s^{-2}$	s^{-2}
zrychlení	a	$m \cdot s^{-2}$	$m \cdot s^{-2}$
Elektrické veličiny			
Název veličiny	**Označení veličiny**	**Jednotka (odvozená)**	**Jednotka (SI)**
elektrické napětí	U	V	$m^2 \cdot kg \cdot s^{-3} \cdot A^{-1}$
elektrický náboj	q	C	$s \cdot A$
Magnetické veličiny			
Název veličiny	**Označení veličiny**	**Jednotka (odvozená)**	**Jednotka (SI)**
mag. indukční tok	Φ	$Wb = V \cdot s$	$m^2 \cdot kg \cdot s^{-2} \cdot A^{-1}$

Tab. 2.2.1a: konverzní relace mezi základními a odvozenými jednotkami soustavy SI, využívanými v popisu modelu vrtulníku

Definice vstupních a výstupních veličin

Model vrtulníku odpovídá systému s více vstupy a více výstupy (MIMO – Multiple Input, Multiple Output), tedy:

$u_1(t) = U_M$ DC napětí hlavního (elevačního) motoru; vstupní parametr

$u_2(t) = U_T$ DC napětí ocasního (azimutového) motoru; vstupní parametr

$u_3(t) = U_B$ DC napětí servomechanismu pro nastavení pozice těžiště; vstupní parametr

$\psi(t) = \psi$ výsledný elevační úhel měřený na lokálním horizontu proti směru hodinových ručiček

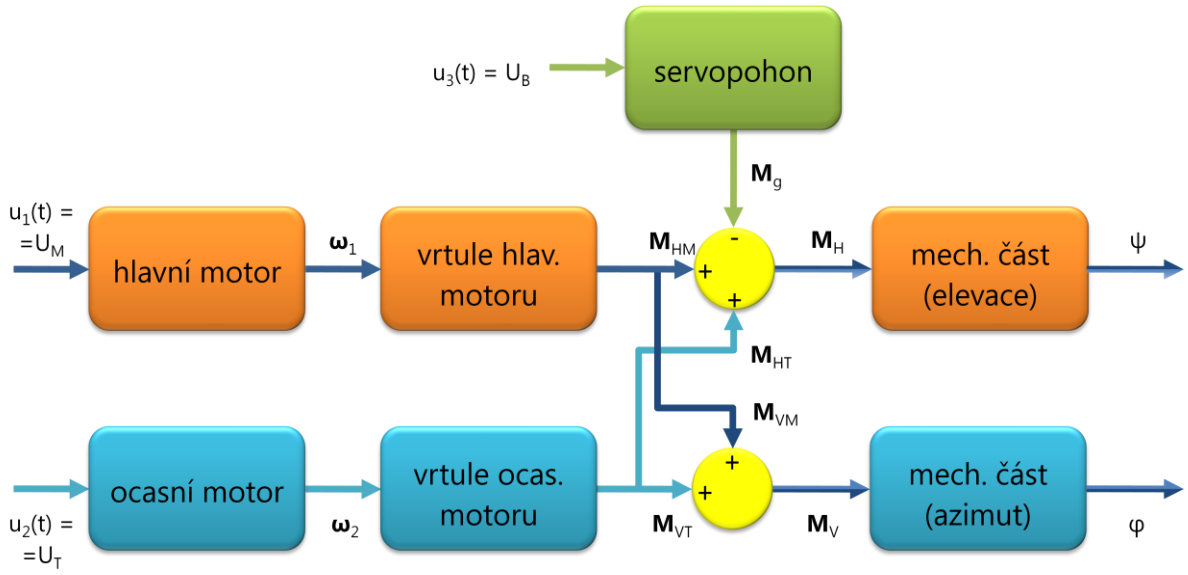
$\varphi(t) = \varphi$ výsledný azimutový úhel měřený v centrální poloze ve směru hodinových ručiček

V případě nastavení těžiště se nejedná o zcela autonomní vstupní parametr, neboť souvisí s chováním těla vrtulníku v elevaci – jedná se o nastavení momentů sil.

Struktura namodelovaného systému

Principiálně lze systém vrtulníku dekomponovat na konkrétní subsystémy, tedy dle [1] [3]:

- **motory** přeměna elektrického napětí na úhlový kmitočet
- **vrtule** přeměna úhlového kmitočtu na moment síly v horizontálním nebo vertikálním směru
- **mechanické části** sestava konstrukčních dílů a částí tvořících kompaktní celek



Obr. 2.2.2: blokové schéma vzájemného propojení jednotlivých subsystémů (motory, vrtule, mechanické části, servopohon) s vyznačením příslušných vstupních, stavových a výstupních veličin

Pro schéma platí tyto momentové relace dle [3]:

$$M_H = M_{HT} + M_{HM} - M_g \qquad (2-01a)$$

$$M_V = M_{VT} + M_{VM} \qquad (2-01b)$$

kde M_H celkový moment síly v horizontální ose

M_{HM} moment síly hlavní vrtule v horizontální ose
M_{HT} moment síly ocasní vrtule v horizontální ose (vazba elevace – azimut)
M_V celkový moment síly ve vertikální ose
M_{VM} moment síly hlavní vrtule ve vertikální ose (vazba azimut – elevace)
M_{VT} moment síly ocasní vrtule ve vertikální ose
M_g moment tíhové síly

Rovnice (2 – 01a) popisuje obecné rozložení silových momentů na horizontu (vertikální rovina) čili v elevaci, přičemž moment tíhové síly M_g lze měnit velikostí napětí $u_3(t) = U_B$ servomechanismu.

Rovnice (2 – 01b) popisuje obecné rozložení silových momentů ve vertikále (horizontální rovina) čili v azimutu.

Z obr. 2.2.2 je rovněž zřejmá křížová vazba mezi elevační a azimutovou částí schématu. Silový moment M_{HT} představuje působení ocasní vrtule na mechanickou část v elevaci, naopak silový moment M_{VM} představuje působení hlavní vrtule na mechanickou část v azimutu – tato vazba se projevuje velice výrazně; vliv azimutu na elevaci není tak patrný.

Modelování dynamiky mechanické části

Dynamika mechanické části je definována momentem setrvačnosti J a třením B, a to v obou osách. Výše uvedený moment setrvačnosti pochopitelně závisí na poloze těžiště, protože vliv se projeví primárně v elevaci a díky vzájemné vazbě elevace-azimut také v azimutu. [1]

Schématicky si lze mechanickou část představit jako blok, do něhož vstupuje moment síly v horizontální ose M_H (elevace) nebo ve vertikální ose M_V (azimut) a vystupuje úhel elevace ψ nebo úhel azimutu φ. Příslušné momentové rovnice jsou následující, tedy dle [3]:

$$M_H = J_H \cdot \dot{\omega}_H + B_H \cdot \omega_H = J_\psi \cdot \ddot{\psi} + B_\psi \cdot \dot{\psi} \qquad (2-02a)$$

$$M_V = J_V \cdot \dot{\omega}_V + B_V \cdot \omega_V = J_\varphi \cdot \ddot{\varphi} + B_\varphi \cdot \dot{\varphi} \qquad (2-02b)$$

z toho

$$\omega_H = \frac{d}{dt}\{\psi(t)\} = \dot{\psi} \qquad (2-03a)$$

$$\dot{\omega}_H = \frac{d}{dt}\{\omega_H(t)\} = \ddot{\psi} \qquad (2-03b)$$

$$\omega_V = \frac{d}{dt}\{\varphi(t)\} = \dot{\varphi} \qquad (2-03c)$$

$$\dot{\omega}_V = \frac{d}{dt}\{\omega_V(t)\} = \ddot{\varphi} \qquad (2-03d)$$

kde

M_H	celkový moment síly v horizontální ose	$[N \cdot m]$
J_H	moment setrvačnosti v horizontální ose	$[kg \cdot m^2]$
B_H	moment hybnosti v horizontální ose (vnitřní tření)	$[kg \cdot m^2 \cdot s^{-1}]$
ψ	úhel elevace	$[rad]$
ω_H	úhlová rychlost v horizontální ose	$[rad \cdot s^{-1}]$
$\dot{\omega}_H$	první derivace úhlové rychlosti v horizontální ose	$[rad \cdot s^{-2}]$
M_V	celkový moment ve vertikální ose	$[N \cdot m]$

J_V	moment setrvačnosti ve vertikální ose	$[kg \cdot m^2]$
B_V	moment hybnosti ve vertikální ose (vnitřní tření)	$[kg \cdot m^2 \cdot s^{-1}]$
φ	úhel elevace	$[rad]$
ω_V	úhlová rychlost ve vertikální ose	$[rad \cdot s^{-1}]$
$\dot{\omega}_V$	první derivace úhlové rychlosti ve vertikální ose	$[rad \cdot s^{-2}]$

Modelování dynamiky v elevaci

Rovnice rozložení momentů sil ve vertikální rovině, které působí na tělo vrtulníku, je dána relací dle [1]:

$$M_1 + M_{\dot{\varphi}} - M_{m_1} - M_{f_1} - M_{gyro} - J_\psi \cdot \ddot{\psi} = 0 \qquad (2-04a)$$

po ekvivalentních úpravách

$$J_\psi \cdot \ddot{\psi} = M_1 + M_{\dot{\varphi}} - M_{f_1} - M_m - M_{gyro} \qquad (2-04b)$$

nebo

$$\ddot{\psi} = \frac{1}{J_\psi} \cdot \left\{ M_1 + M_{\dot{\varphi}} - M_{f_1} - M_m - M_{gyro} \right\} \qquad (2-04c)$$

za předpokladu

$$J_\psi > 0 \qquad (2-04d)$$

z toho

$$J_\psi = m_1 \cdot {l_1}^2 \qquad (2-05a)$$

$$M_1 = K_{\omega_1} \cdot \omega_1^2 \cdot sign(\omega_1) \qquad (2-05b)$$

$$M_{\dot{\varphi}} = m_1 \cdot l_1 \cdot \dot{\varphi}^2 \cdot sin(\psi) \cdot cos(\psi) = \frac{1}{2} \cdot m_1 \cdot l_1 \cdot \dot{\varphi}^2 \cdot sin(2 \cdot \psi) \qquad (2-05c)$$

$$M_{f_1} = B_{\psi_1} \cdot \dot{\psi} + B_{\psi_2} \cdot sign(\dot{\psi}) \qquad (2-05d)$$

$$M_{m_1} = F_{m_1} \cdot l_1 \cdot sin(\psi) = m_1 \cdot g \cdot l_1 \cdot sin(\psi) = M_{g_1} \cdot sin(\psi) = M_g \qquad (2-05e)$$

$$M_{gyro} = K_{gyro} \cdot \omega_1 \cdot \dot{\varphi} \cdot cos(\psi) \; pro \; \dot{\varphi} \ll \omega_1 \qquad (2-05f)$$

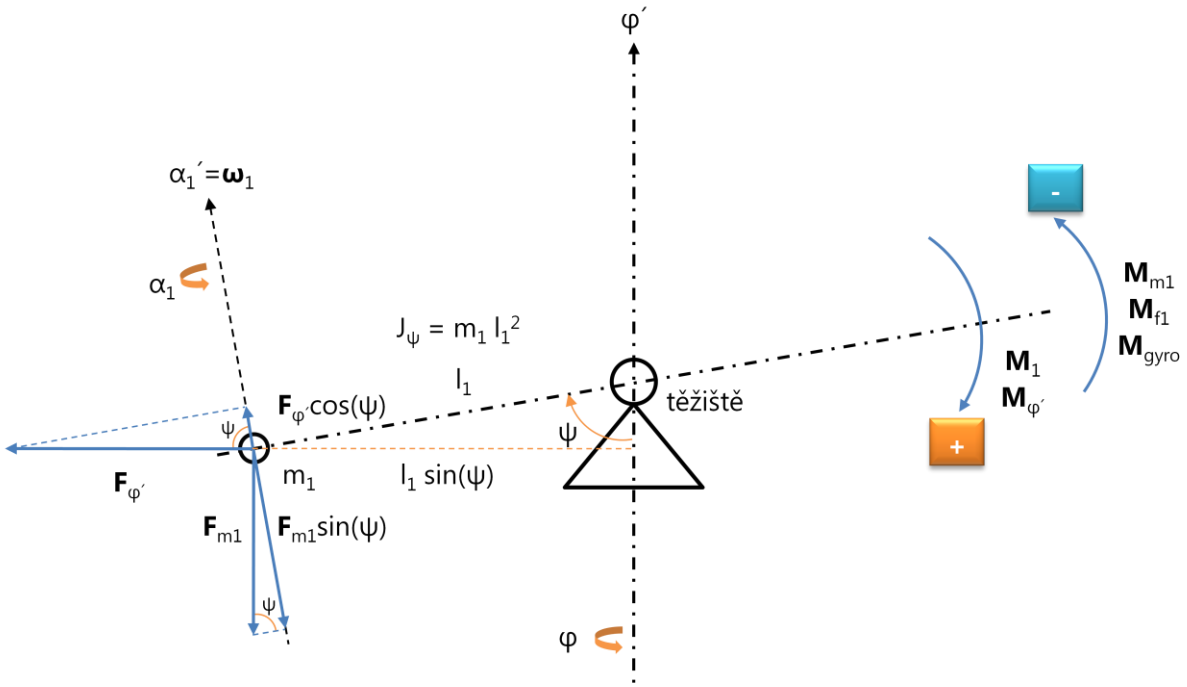
kde

M_1	moment síly vrtule elevačního (hlavního) motoru	$[N \cdot m]$
$M_{\dot{\varphi}}$	moment dostředivé síly	$[N \cdot m]$
M_{f_1}	moment třecí síly (vnitřní tření a Coulombovo tření)	$[N \cdot m]$
M_{m_1}	moment tíhové síly hmotného bodu vrtulníku	$[N \cdot m]$
M_{gyro}	moment gyroskopické síly	$[N \cdot m]$
M_{g_1}	amplituda momentu tíhové síly hmotného bodu	$[N \cdot m]$
J_ψ	moment setrvačnosti těla vrtulníku kolem horizontální osy	$[kg \cdot m^2]$
B_{ψ_1}	moment hybnosti těla vrtulníku (bez uvažování směru)	$[kg \cdot m^2 \cdot s^{-1}]$
B_{ψ_2}	moment hybnosti těla vrtulníku (s uvažováním směru)	$[kg \cdot m^2 \cdot s^{-1}]$
ψ	úhel elevace	$[rad]$
$\dot{\psi}$	první derivace úhlu elevace podle času	$[rad \cdot s^{-1}]$

$\ddot{\psi}$	druhá derivace úhlu elevace podle času	$[rad \cdot s^{-2}]$
φ	úhel azimutu	$[rad]$
$\dot{\varphi}$	první derivace úhlu azimutu podle času	$[rad \cdot s^{-1}]$
ω_1	úhlová rychlost vrtule elevačního motoru	$[rad \cdot s^{-1}]$
K_{gyro}	konstanta gyroskopické vazby elevace-azimut	$[kg \cdot m^2]$
K_{ω_1}	přístrojová konstanta elevačního motoru	$[kg \cdot m^2 \cdot s]$
m_1	hmotnost hmotného bodu	$[kg]$
l_1	vzdálenost mezi hmotným bodem a těžištěm	$[m]$
g	tíhové zrychlení	$[m \cdot s^{-2}]$

Některé vlivy jsou zanedbatelné, např. reakční síla stabilizace motoru a proměnlivost odporu vzduchu, závisející na otáčkách hlavní vrtule (elevace). Zatímco vlivy postranního motoru na elevační úhel jsou téměř zanedbatelné, proměnlivost tlumení oscilací těla vrtulníku je naopak zásadní. Vliv rychlosti otáčení hlavní vrtule na třecí sílu v elevaci se analyticky těžko modeluje, tudíž musí být vyhodnocen experimentálně. [1]

Výše uvedené matematické relace jsou graficky znázorněny ve schématu dynamiky v elevaci (viz obr. 2.2.3) dle [1].



Obr. 2.2.3: schéma dynamiky v elevaci – význam těžiště, silových momentů a analýza sil působících na tělo vrtulníku (hmotný bod) ve vertikální rovině čili horizontální ose rotace

Pracovní bod v rovnovážném stavu

Helikoptéra se nachází v rovnováze, jestliže elevační úhel a azimutový úhel jsou konstantní a jestliže se motory (hlavní elevační motor, ocasní azimutový motor) otáčejí konstantní úhlovou rychlostí. Velikosti vstupních signálů, nezbytných k nastavení rovnovážného stavu vrtulníku, definují pracovní body. [3]

Uvažujeme-li vzdálenost l_1 (mezi hmotným bodem těla vrtulníku a těžištěm; viz obr. 2.2.3), pak moment způsobený tíhovou silou je dán relací (2 – 05e).

Tento popis představuje nezbytnou část modelování dynamiky v elevaci. Budeme-li uvažovat malé změny elevačního úhlu, tj. do hodnoty přibližně 5 [°], pak moment tíhové síly hmotného bodu těla vrtulníku M_{m_1} je přímo úměrný jeho pozici, respektive stejnosměrnému napětí servomechanismu U_B pro nastavení pozice těžiště, tedy dle [1]:

$$M_{m_1} = k_B \cdot U_B = M_g \qquad (2-05g)$$

kde			
	k_B	přenosová konstanta servomechanismu	$[N \cdot m \cdot V^{-1}]$
	U_B	DC napětí servomechanismu pro nastavení pozice těžiště	$[V]$
	M_g	moment tíhové síly hmotného bodu vrtulníku	$[N \cdot m]$

Přenosová konstanta k_B se stará o přepočet velikosti přiloženého napětí U_B na vzdálenost l_1 při konstantní hmotnosti m_1.

Modelování dynamiky v azimutu

Následující rovnice ukazují rovnováhu sil v horizontální rovině vzhledem k hlavní síle působící na tělo vrtulníku ve směru úhlu azimutu, tedy dle [1]:

$$M_2 - M_{f_2} - M_r - J_\varphi \cdot \ddot{\varphi} = 0 \qquad (2-06a)$$

po ekvivalentních úpravách

$$J_\varphi \cdot \ddot{\varphi} = M_2 - M_{f_2} - M_r \qquad (2-06b)$$

nebo

$$\ddot{\varphi} = \frac{1}{J_\varphi} \cdot \left[M_2 - M_{f_2} - M_r\right] \qquad (2-06c)$$

za předpokladu

$$J_\varphi > 0 \qquad (2-06d)$$

z toho

$$J_\varphi = J_\psi \cdot sin(\psi) = m_1 \cdot {l_1}^2 \cdot sin(\psi) \qquad (2-07a)$$

$$M_2 = K_{\omega_2} \cdot \omega_2^2 \cdot sign(\omega_2) \qquad (2-07b)$$

$$M_{f_2} = B_{\varphi_1} \cdot \dot{\varphi} + B_{\varphi_2} \cdot sign(\dot{\varphi}) \qquad (2-07c)$$

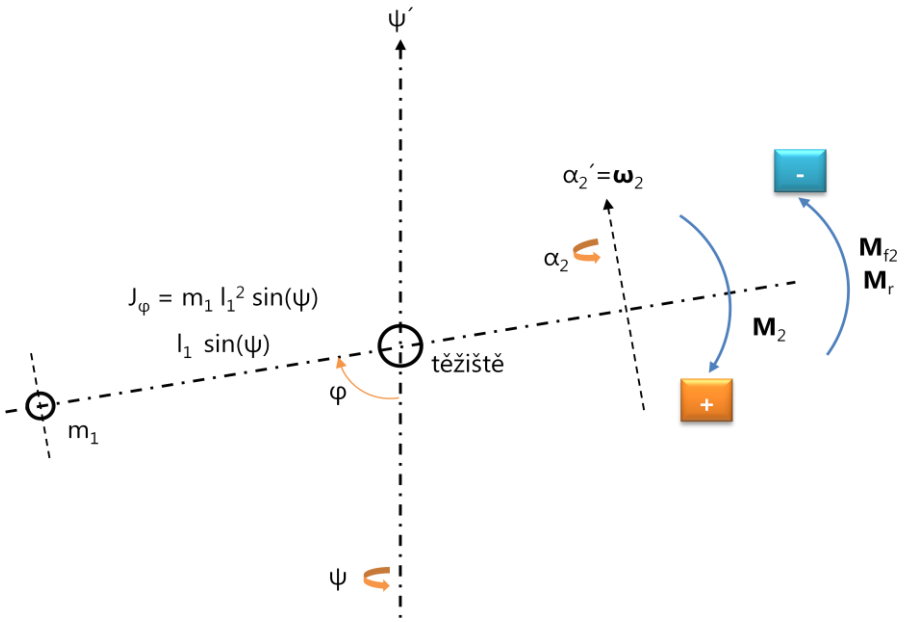
kde			
	M_2	moment síly vrtule ocasního (azimutového) motoru	$[N \cdot m]$
	M_{f_2}	moment třecí síly (vnitřní tření a Coulombovo tření)	$[N \cdot m]$
	M_r	moment reakční síly hlavního motoru	$[N \cdot m]$
	J_ψ	moment setrvačnosti těla vrtulníku kolem vertikální osy	$[kg \cdot m^2]$
	B_{φ_1}	moment hybnosti těla vrtulníku (bez uvažování směru)	$[kg \cdot m^2 \cdot s^{-1}]$
	B_{φ_2}	moment hybnosti těla vrtulníku (s uvažováním směru)	$[kg \cdot m^2 \cdot s^{-1}]$
	φ	úhel azimutu	$[rad]$
	$\dot{\varphi}$	první derivace úhlu azimutu podle času	$[rad \cdot s^{-1}]$
	$\ddot{\varphi}$	druhá derivace úhlu azimutu podle času	$[rad \cdot s^{-2}]$

ψ	úhel elevace	$[rad]$
$\dot{\psi}$	první derivace úhlu elevace podle času	$[rad \cdot s^{-1}]$
ω_2	úhlová rychlost vrtule azimutového motoru	$[rad \cdot s^{-1}]$
K_{ω_2}	přístrojová konstanta azimutového motoru	$[kg \cdot m^2 \cdot s]$
m_1	hmotnost hmotného bodu	$[kg]$
l_1	vzdálenost mezi hmotným bodem a těžištěm	$[m]$

Moment reakční síly M_r je důležitý, protože vzniká z momentu síly generovaného hlavním (elevačním) motorem při působení na rotující tělo vrtulníku. Moment setrvačnosti v azimutu J_φ je závislý na sinu úhlu elevace ψ (viz 2 – 07a), přičemž rovnost obou momentů setrvačnosti nastane, bude-li platit:

$$\psi = \frac{\pi}{2} \Rightarrow J_\varphi = m_1 \cdot {l_1}^2 \cdot sin\left(\frac{\pi}{2}\right) = m_1 \cdot {l_1}^2 \cdot 1 = m_1 \cdot {l_1}^2 = J_\psi \qquad (2-07d)$$

Výše uvedené matematické relace jsou graficky znázorněny ve schématu dynamiky v azimutu (viz obr. 2.2.4) dle [1].



Obr. 2.2.4: schéma dynamiky v azimutu – význam těžiště, silových momentů a azimutového momentu setrvačnosti v závislosti na úhlu elevace

Modelování dynamiky soustavy stejnosměrného motoru a vrtule

Rotor obecně charakterizujeme jako hmotu s určitým momentem setrvačnosti J, otáčející se úhlovou rychlostí ω a překonávající vnitřní tření B. Uvažujeme-li, že výsledný moment síly motoru M lineárně závisí na přiloženém vstupním napětí, pak platí obecné relace dle [3]:

$$k_M \cdot U_M = J_M \cdot \dot{\omega}_M + B_M \cdot \omega_M \qquad (2-08a)$$

$$k_T \cdot U_T = J_T \cdot \dot{\omega}_T + B_T \cdot \omega_T \qquad (2-08b)$$

kde			
	k_M	přenosová konstanta hlavního motoru	$[N \cdot m \cdot V^{-1}]$
	U_M	DC napětí hlavního motoru	$[V]$
	J_M	moment setrvačnosti hlavního motoru	$[kg \cdot m^2]$

B_M	moment hybnosti hlavního motoru (vnitřní tření)	$[kg \cdot m^2 \cdot s^{-1}]$
ω_M	úhlový kmitočet hlavního motoru	$[rad \cdot s^{-1}]$
$\dot{\omega}_M$	první derivace úhlového kmitočtu hlavního motoru	$[rad \cdot s^{-2}]$
k_T	přenosová konstanta ocasního motoru	$[N \cdot m \cdot V^{-1}]$
U_T	DC napětí ocasního motoru	$[V]$
J_T	moment setrvačnosti ocasního motoru	$[kg \cdot m^2]$
B_T	moment hybnosti ocasního motoru (vnitřní tření)	$[kg \cdot m^2 \cdot s^{-1}]$
ω_T	úhlový kmitočet ocasního motoru	$[rad \cdot s^{-1}]$
$\dot{\omega}_T$	první derivace úhlového kmitočtu ocasního motoru	$[rad \cdot s^{-2}]$

Přenosová charakteristika momentu síly vrtule v závislosti na úhlové rychlosti ω motoru závisí na geometrii vrtule, přičemž v tomto případě mají pevnou geometrii aproximovanou lineárně kvadratickou funkcí (momentové rovnice křížové vazby), tedy dle [3]:

$$M_{HM} = k_{2HM} \cdot \omega_M^2 + k_{1HM} \cdot \omega_M \qquad (2-09a)$$

$$M_{HT} = k_{2HT} \cdot \omega_T^2 + k_{1HT} \cdot \omega_T \qquad (2-09b)$$

$$M_{VM} = k_{2VM} \cdot \omega_M^2 + k_{1VM} \cdot \omega_M \qquad (2-09c)$$

$$M_{VT} = k_{2VT} \cdot \omega_T^2 + k_{1VT} \cdot \omega_T \qquad (2-09d)$$

kde

k_{1HM}	moment hybnosti hlavní vrtule v horizontální ose	$[kg \cdot m^2 \cdot s^{-1}]$
k_{1VM}	moment hybnosti hlavní vrtule ve vertikální ose	$[kg \cdot m^2 \cdot s^{-1}]$
k_{1HT}	moment hybnosti ocasní vrtule v horizontální ose	$[kg \cdot m^2 \cdot s^{-1}]$
k_{1VT}	moment hybnosti ocasní vrtule ve vertikální ose	$[kg \cdot m^2 \cdot s^{-1}]$
k_{2HM}	moment setrvačnosti hlavní vrtule v horizontální ose	$[kg \cdot m^2]$
k_{2VM}	moment setrvačnosti hlavní vrtule ve vertikální ose	$[kg \cdot m^2]$
k_{2HT}	moment setrvačnosti ocasní vrtule v horizontální ose	$[kg \cdot m^2]$
k_{2VT}	moment setrvačnosti ocasní vrtule ve vertikální ose	$[kg \cdot m^2]$

Dynamika soustavy motor-vrtule je popsána mechanickými a elektrickými veličinami s určitou mírou významnosti – např. vlastní indukčnost kotvy stejnosměrného motoru je velmi malá, kdežto Coulombovo tření a odporová síla, vznikající rotací vrtule ve vzduchu, jsou významné. Obecné rovnice vyšetřované soustavy jsou formálně shodné jak v elevaci, tak v azimutu, tedy dle [1]:

$$M_i - M_{ib} - M_\omega - M_P - J \cdot \dot{\omega} = 0 \qquad (2-10a)$$

po ekvivalentních úpravách

$$J \cdot \dot{\omega} = M_i - M_{ib} - M_\omega - M_P \qquad (2-10b)$$

nebo

$$\dot{\omega} = \frac{1}{J} \cdot [M_i - M_{ib} - M_\omega - M_P] \qquad (2-10c)$$

za předpokladu

$$J > 0 \qquad (2-10d)$$

z toho

$$M_i = K_i \cdot i(t) = \frac{K_i}{R} \cdot u(t) = K_i \cdot i = \frac{K_i}{R} \cdot u \qquad (2-11a)$$

$$M_{ib} = \frac{K_i}{R} \cdot u_b(t) = \frac{K_i}{R} \cdot u_b = \frac{K_i}{R} \cdot K_b \cdot \omega \qquad (2-11b)$$

$$M_\omega = B \cdot \omega + C \cdot sign(\omega) \qquad (2-11c)$$

$$M_P = B_P \cdot \omega + D_P \cdot \omega^2 \qquad (2-11d)$$

kde			
	u	vstupní řídicí (budící) elektrické napětí	$[V]$
	u_b	elektromotorické napětí (zdroj protinapětí)	$[V]$
	i	elektrický proud protékající kotvou DC motoru	$[A]$
	R	elektrický odpor kotvy DC motoru	$[\Omega]$
	ω	úhlová rychlost	$[rad \cdot s^{-1}]$
	K_b	konstanta DC motoru (magnetický indukční tok)	$[V \cdot s]$
	K_i	proudová silová konstanta	$[N \cdot m \cdot A^{-1}]$
	M_i	moment síly způsobený budícím napětím	$[N \cdot m]$
	M_{ib}	moment síly způsobený protinapětím	$[N \cdot m]$
	M_ω	moment síly způsobený třením	$[N \cdot m]$
	M_P	moment síly způsobený rotací vrtule	$[N \cdot m]$
	B	moment hybnosti způsobený viskózním třením	$[kg \cdot m^2 \cdot s^{-1}]$
	C	moment hybnosti způsobený Coulombovým třením	$[kg \cdot m^2 \cdot s^{-1}]$
	B_P	moment hybnosti odporu vzduchu (laminárním proudění)	$[kg \cdot m^2 \cdot s^{-1}]$
	D_P	moment setrvačnosti odporu vzduchu (turbulentním proudění)	$[kg \cdot m^2]$

Vrtule, přímo spojené hřídelí s motory, vytváří během rotace moment síly v elevaci M_1 (2 − 05b) a moment síly v azimutu M_2 (2 − 07b). Silové momenty M_i, M_{ib} a M_ω vytváří motor, kdežto silový moment M_P vzniká při rotaci vrtule.

Modelování senzoru a výkonového zesilovače

Uživatel s modelem vrtulníku komunikuje přes rozhraní nástroje MATLAB Real Time Toolbox, kdy všechny vstupní a výstupní signály jsou škálovány na uzavřený interval dle [1] [5]:

$$I_{MU} = \langle -1;\ +1 \rangle \qquad (2-12a)$$

kde			
	I_{MU}	interval škálovaných hodnot vstupních a výstupních signálů	$[MU]$

Jednotka [MU] představuje tzv. strojovou jednotku MU (Machine Unit), která nevyjadřuje fyzikální rozměr vstupních napěťových signálů. Tuto jednotku zavádíme proto, aby do modelu vstupovalo napětí ve voltech [V]. [1]

V případě výstupních signálů formálně definujeme jednotku [MP], abychom zajistili, že z modelu budou vystupovat úhly elevace a azimutu v radiánech [rad]. Pro přímé měření úhlu elevace a úhlu azimutu jsou použita inkrementální rotační čidla (IRC), tvořící statickou (statická převodní charakteristika) a v celém rozsahu měření úhlů lineární část měřicího řetězce. Pro přepočet elevace a azimutu platí tyto relace, tedy:

$$y_\psi(t) = y_\psi = k_\psi \cdot \psi + y_{\psi_0} \qquad (2-12b)$$

$$y_\varphi(t) = y_\varphi = k_\varphi \cdot \varphi + y_{\varphi_0} \qquad (2-12c)$$

kde			
	ψ	úhel elevace	$[MP]$
	y_ψ	funkční hodnota výstupního úhlu pro elevaci	$[rad]$
	k_ψ	přepočtová konstanta elevačního senzoru	$[rad \cdot MP^{-1}]$
	y_{ψ_0}	ofset elevačního senzoru při $y_\psi = 0\ [rad]$	$[rad]$
	φ	úhel azimutu	$[MP]$
	y_φ	funkční hodnota výstupního úhlu pro azimut	$[rad]$
	k_φ	přepočtová konstanta azimutového senzorů	$[rad \cdot MP^{-1}]$
	y_{ψ_0}	ofset elevačního senzoru při $y_\varphi = 0\ [rad]$	$[rad]$

Výkonové zesilovače, řízené přes měřicí kartu typu PnP (Plug-and-Play), ovládají stejnosměrné motory coby akční členy, které nastavují příslušné úhly elevace a azimutu. Mezi výstupní hodnotou z PC a fyzikální veličinou, tj. stejnosměrným napětím pro motory, platí lineární závislost dle relace dle [1]:

$$u(t) = u_{výst.} = K \cdot h_{PC} \qquad (2-12d)$$

kde			
	$u_{výst.}$	budící napětí pro stejnosměrný motor	$[V]$
	K	zesílení výkonového zesilovače	$[V \cdot MU^{-1}]$
	h_{PC}	hodnota výstupního signálu z PC	$[MU]$

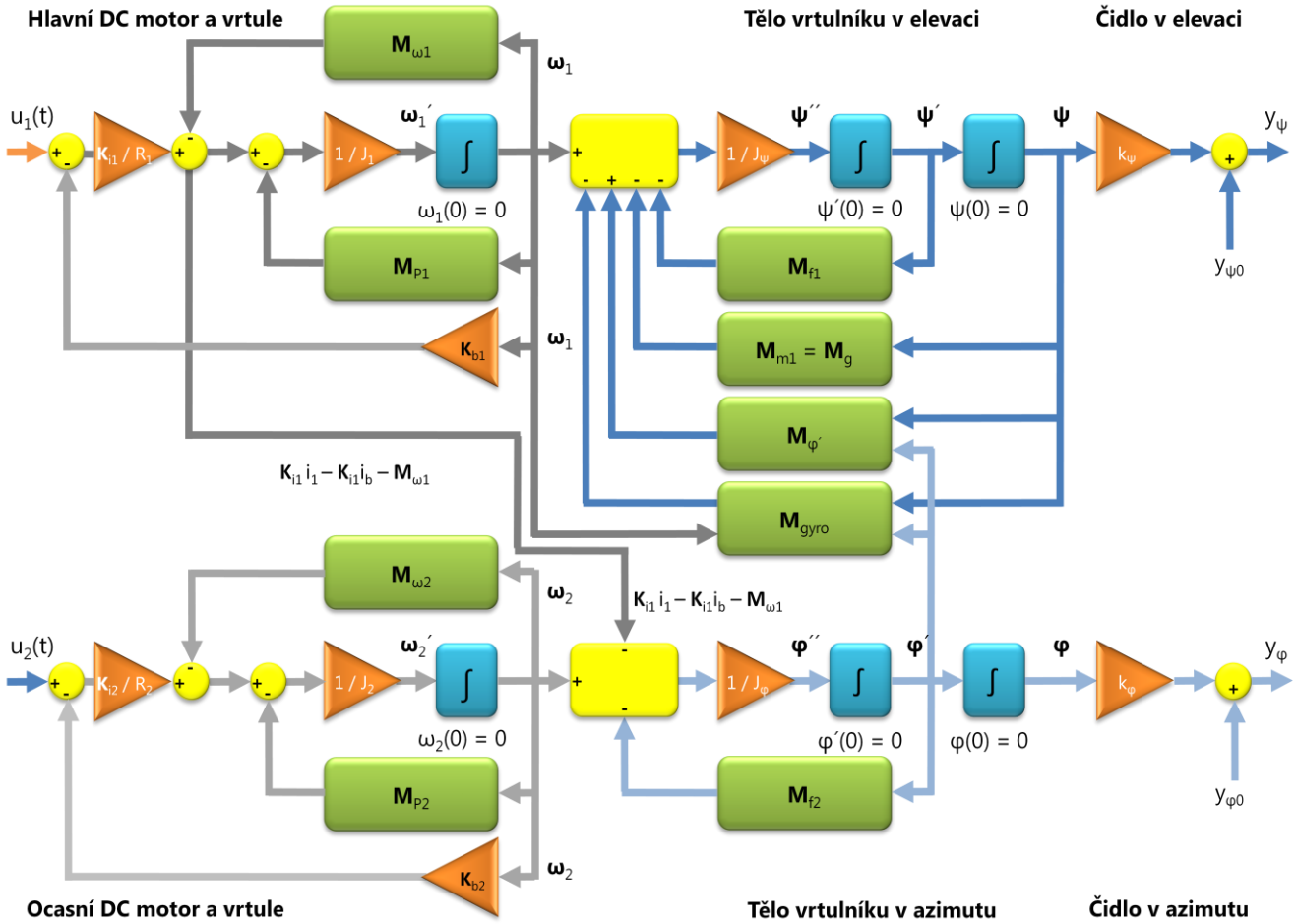
Z hlediska PC se jedná o výstupní signál, kdežto z hlediska modelu vrtulníku je jedná o vstupní řídicí signál.

Celková dynamika systému – teoretický model

Celkovou dynamiku těla vrtulníku jsme popsali jednak obecně nelineárními diferenciálními rovnicemi (platí pro dynamické subsystémy), tak rovněž lineárními algebraickými rovnicemi (platí pro statické subsystémy), tedy:

- soustava hlavního motoru a vrtule *dynamický subsystém*
- soustava ocasního motoru a vrtule *dynamický subsystém*
- tělo vrtulníku v elevaci *dynamický subsystém*
- tělo vrtulníku v azimutu *dynamický subsystém*
- čidlo v elevaci *statický subsystém*
- čidlo v azimutu *statický subsystém*

Sloučením všech těchto popisů lze vytvořit celkové teoretické blokové schéma čili teoretický model, tedy dle [1] [5]:



Obr. 2.2.5: blokové schéma kompletní dynamiky systému – teoretický model

2.2.3 Identifikace

Existují dva možné identifikační přístupy pro odhad parametrů systému, tedy dle [1]:

- identifikace jednotlivých subsystémů přímým měřením příslušných fyzikálních veličin
- analýza vstupních a výstupních systémových signálů – testování systému metodou černé skříňky (Black Box)

První metoda je sice časově náročnější, ale dobře vypovídá o chování systému z hlediska jeho fyzikální podstaty a funkcionality. Nevýhoda této metody spočívá v tom, že ne všechny fyzikální veličiny lze přímým měřením získat. [1]

Druhá metoda je sice elegantní, ovšem daný systém musí být lineární, aby ho bylo možno popsat obyčejnou lineární diferenciální rovnicí (OLDR), respektive přenosovou funkcí v Laplaceově transformaci. V případě nelineárních systémů je nutno takovýto model v okolí pracovního bodu linearizovat. [1]

Pro model vrtulníku byly identifikovány tyto subsystémy v elevaci (viz obr. 2.2.5), tedy dle [1]:

- soustava výkonového zesilovače a vedlejšího motoru
- tělo vrtulníku ve vertikální rovině
- senzor úhlu elevace

Pro model vrtulníku byly identifikovány tyto subsystémy v azimutu (viz obr. 2.2.5), tedy:

- soustava výkonového zesilovače a hlavního motoru
- tělo vrtulníku v horizontální rovině
- senzor úhlu azimutu

Pomocí identifikace vytváříme zjednodušený teoretický model neboli empirický model.

Rozsahy a přepočty jednotek

V empirickém modelu pracujeme nejen s fyzikálními jednotkami (elektrická napětí, momenty síly, úhlové rychlosti apod.), ale také s jednotkami bez fyzikálního významu. Konverzní konstanty se využívají pouze na vstupech a na výstupech ze zkoumaného procesu:

- **vstup:** budící napětí ve [V] se konvertují na číslicové signály v [MU]
- **výstup:** úhly v [MP] se konvertují na výstupní úhly elevace a azimutu v [rad]

Ve výpočtech si lze zvolit typ jednotek samozřejmě zvolit, ovšem je třeba striktně daný typ jednotek dodržet, aby nedocházelo ke zkreslení dosažených výsledků.

Přepočty vstupních veličin jsou následující dle [1] [3]:

Rozsahy a konverzní konstanty			
Vstupní veličiny	**Rozsah ve strojových jednotkách**		**Konverzní konstanta**
	$[MU]$	$[MU]$	$[V \cdot MU^{-1}]$
$\boldsymbol{U_M}$	-1	+1	12 / 1
$\boldsymbol{U_T}$	-1	+1	6 / 1
$\boldsymbol{U_B}$	-0,90	-0,80	1 / 1

Tab. 2.2.1b: rozsahy ve strojových jednotkách a konverzní konstanty pro vstupní napětí – elevační motor, azimutový motor a servomechanismus pro nastavení těžiště

Stanovení přepočtu výstupních veličin je náplní identifikace a kalibrace senzorů.

Identifikace a kalibrace senzorů

Kalibraci úhlových čidel lze provést přímo, neboť tento subsystém již vykazuje linearitu. Naším úkolem bude určit:

- hodnoty přepočtových konstant IRC senzorů – k_ψ a k_φ
- hodnoty posunutí funkčních hodnot IRC senzorů – y_{ψ_0} a y_{φ_0}

Model vrtulníku využívá k měření výstupních hodnot y_{ψ_0} a y_{φ_0} inkrementální rotační senzory (IRC) nebo též relativní čidla, které se typicky využívají ve zpětnovazebních systémech řízení polohy, rychlosti a popřípadě zrychlení v nejrůznějších aplikacích. Jsou charakteristické svou velkou rozlišovací schopností, malými rozměry a hmotností. Princip jejich činnosti je založen na otáčivém mezikruží s pravidelně se střídajícími transparentními a netransparentními ploškami (rysky), jež při rotaci přerušují emitovaný světelný paprsek svítivé LED diody, umístěné na jedné straně mezikruží. Průchod světla detekuje buď fototranzistor, nebo jiný světelný senzor, umístěný na druhé straně mezikruží – naproti LED diody. V této optické cestě mezi vysílačem a přijímačem bývá u většiny IRC senzorů vřazen nepohyblivý maskovací kotouček s ryskami o stejné rozteči, jakou má pohyblivý kotouček. [4]

Světlo ze zdroje prochází přes průhledné rysky průhledného kotoučku. Jsou-li v zákrytu průhledné rysky pohyblivého kotouče a průhledné rysky segmentu pevného maskovacího kotouče, dopadá na fotosenzor maximální velikost světelného toku od LED diody. V případě, že dojde k zákrytu průhledné rysky pohyblivého kotouče a segmentu nepohyblivého kotouče, dopad světla na fotosenzor je minimální. Mezi těmito pozicemi se světelný tok mění úměrně posunutí obou kotoučů. Výsledný kvazisinusový signál komparátor převádí na obdélníkový signál. [4]

Kromě výše uvedeného transmisního principu lze využít také princip reflexní, tj. zdroj světla i fotosenzor jsou umístěny na téže straně kotouče. [4]

Je-li potřeba navíc rozlišovat směr rotace, musí být pevný maskovací kotouč polohy opatřen druhým segmentem s ryskami posunutými vůči ryskám prvního segmentu o úhel daný relací, tedy dle [4]:

$$\beta = \left[k + \frac{1}{2}\right] \cdot \frac{2 \cdot \pi}{n} \qquad (2-13)$$

kde	β	úhel posunutí prvního a druhého segmentu	$[rad]$
	k	celočíselná konstanta	$[-]$
	n	počet průhledných a neprůhledných rysek na obvodu kotouče	$[-]$

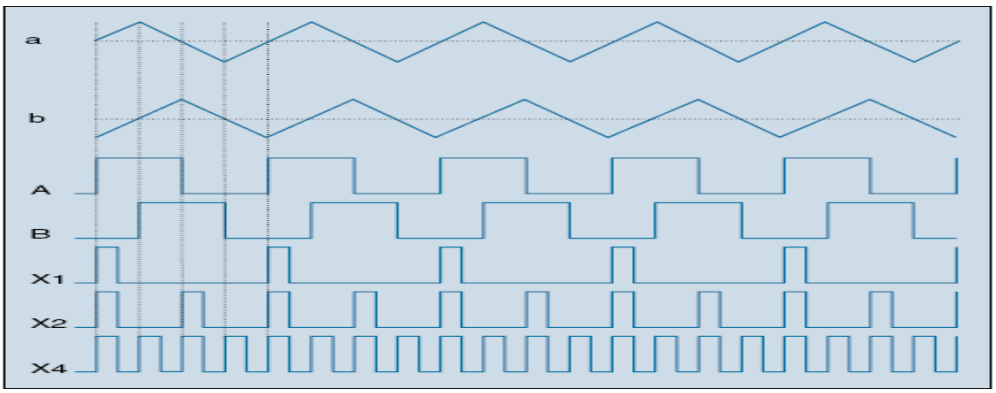
K tomuto segmentu přísluší druhý fotosenzor, snímající fázově posunutý světelný tok, přičemž signál z prvního fotosenzor označujeme jako A-signál, z druhého fotosenzoru jako B-signál. Detekováním změny fáze obou signálů získáme informaci o změně směru rotace. Tyto signály jsou někdy nazývány jako kvadraturní. [4]

Pro úplnost dodejme, že pohyblivý kotouček někdy obsahuje doplňkový otvor (průhledná ryska) a případně další soustavu světelného zdroje a fotosenzor, který detekuje tzv. referenční neboli nulovou pozici ve formě jednoho pulsu na otáčku. Takový se označuje jako referenční, nulový nebo také index-plus. [4]

U některých inkrementálních senzorů bývá jejich kvadraturní výstup doplněn tzv. komplementárními signály (signály v protifázi oproti kvadraturním signálům) z důvodu zvýšení odolnosti výstupních signálů proti rušení. Významnou vlastností inkrementálních čidel je rovněž možnost měnit rozlišení v závislosti na typu detekovaných hran dle [4]:

- **označení X1** detekce čela nebo týlu pouze jednoho kvadraturního signálu
- **označení X2** detekce čela a týlu pouze jednoho kvadraturního signálu
- **označení X4** detekce čela a týlu obou kvadraturních signálů

Situaci graficky reprezentuje obr. 2.2.6, tedy:



Obr. 2.2.6: detekce kvadraturních signálů (A, B) detekčními typy X1, X2 a X4

Množina hodnot úhlu elevace ψ je dána uzavřeným intervalem:

v radiánech

$$\psi \in \langle 0; \frac{\pi}{2} \rangle = \langle \psi_{min}\,; \psi_{max}\,\rangle \qquad (2-14a)$$

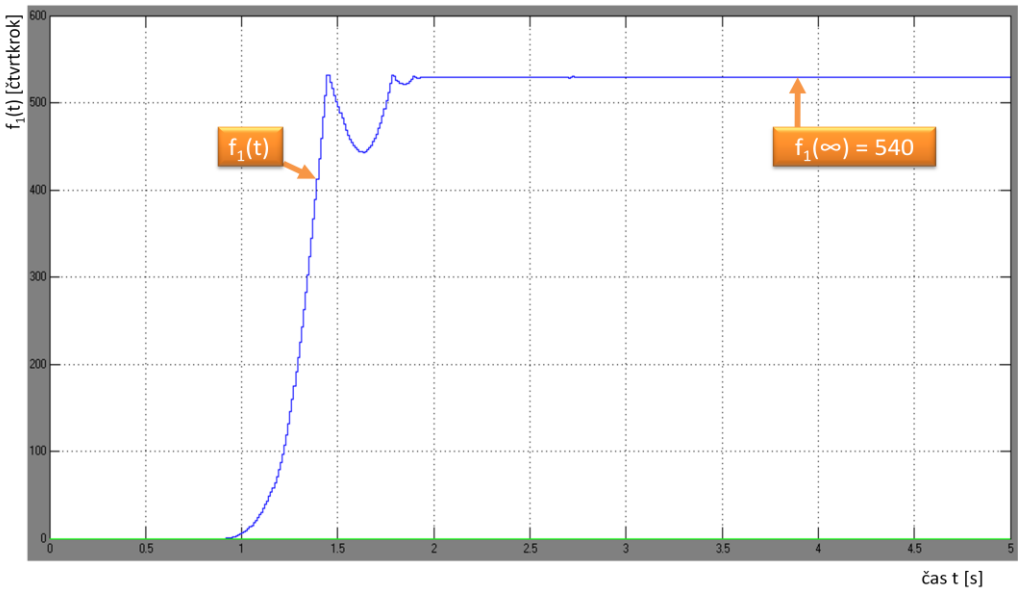
ve stupních

$$\psi \in \langle 0 \cdot \frac{180}{\pi}; \frac{\pi}{2} \cdot \frac{180}{\pi} \rangle = \langle 0; 90 \rangle \qquad (2-14b)$$

Rozsah elevace je dán relací (v radiánech):

$$\psi_{FS} = \psi_{max} - \psi_{min} = \frac{\pi}{2} - 0 = \frac{\pi}{2} \qquad (2-14c)$$

Nyní je třeba určit počet kroků elevačního inkrementálního senzoru v ustáleném stavu odezvy modelu na maximální velikost budícího napětí $U_M = 1{,}0\ [MU] = 12{,}0\ [V]$.



Obr. 2.2.7: odezva modelu na maximální velikost budícího napětí hlavního motoru $U_M = 1{,}0\ [MU] = 12{,}0\ [V]$, při $U_B = -0{,}85\ [V]$, $U_T = 0{,}0\ [MU] = 0{,}0\ [V]$ a X4 detekci kvadraturních signálů

Z odezvy na obr. 2.2.7 je zřejmé, že ustálený stav této charakteristiky odpovídá hodnotě (pro čtvrtinový krok a celý krok):

$$f_1(t = +\infty) = 540\ [čtvrtkroky] = \frac{540}{4}\ [kroky] = 135\ [kroky] = 135\ [MP] \qquad (2-14d)$$

Pro přepočet mezi jednotkami [rad] a [MP] lze určit konstantu $k_1 = k_\psi$ dle relace:

pro celý krok

$$k_1 = \frac{\psi_{FS}}{f_1(\infty)} = \frac{\pi}{2} \cdot \frac{1}{135} = \frac{\pi}{270}\ [rad \cdot MP^{-1}] \cong \frac{\pi}{256}[rad \cdot MP^{-1}] \qquad (2-14e)$$

pro čtvrtinový krok

$$k_1 = \frac{1}{4} \cdot \frac{\psi_{FS}}{f_1(\infty)} = \frac{1}{4} \cdot \frac{\pi}{2 \cdot 270} = \frac{\pi}{1080}\ [rad \cdot MP^{-1}] \cong \frac{\pi}{1024}[rad \cdot MP^{-1}] \qquad (2-14f)$$

Posunutí funkční hodnoty (ofset) posouvá nulový výstup senzoru z nulové vertikální pozice tak, aby byl nulový signál uprostřed výstupní hodnoty úhlu elevace y_ψ. Musíme tedy odečíst polovinu rozsahu elevace ψ_{FS}:

$$y_{\psi_0} = -\frac{\psi_{FS}}{2} = -\frac{\pi}{2} \cdot \frac{1}{2} = -\frac{\pi}{4}\ [rad] \qquad (2-14g)$$

Výsledná lineární algebraická rovnice elevačního senzoru je dána relací (pro čtvrtinový krok):

při naměřené odezvě

$$y_\psi = k_\psi \cdot \psi + y_{\psi_0} = \frac{\pi}{1080} \cdot \psi - \frac{\pi}{4} = \frac{\pi}{1080} \cdot (\psi - 270)\ [rad] \qquad (2-14h)$$

při aproximaci

$$y_\psi = k_\psi \cdot \psi + y_{\psi_0} = \frac{\pi}{1024} \cdot \psi - \frac{\pi}{4} = \frac{\pi \cdot \psi - 256 \cdot \pi}{1024} = \frac{\pi}{1024} \cdot (\psi - 256)\ [rad] \qquad (2-14ch)$$

Do rovnic (2 − 14h) a (2 − 14ch) lze za ψ dosadit funkční hodnotu z grafické závislosti zobrazené na obr. 2.2.7, neboli dané rovnice jsou přizpůsobeny pro detekci X4 (čtvrtinový krok).

Pokud bychom rovnici (2 -14h), respektive (2 – 14ch) analyzovali, dostaneme tyto výsledky:

$\boldsymbol{\psi}$	$\boldsymbol{y_\psi}$
$[čtvrtkroky]$	$[rad]$
0	$-\pi/4$
270	0
540	$+\pi/4$

Tab. 2.2.2a: závislost výstupní hodnoty úhlu elevace na elevačním úhlu při detekci X4

Množina hodnot úhlu azimutu φ je dána uzavřeným intervalem:

v radiánech

$$\varphi \in \langle -\frac{140 \cdot \pi}{180};\ +\frac{140 \cdot \pi}{180} \rangle = \langle \varphi_{min}\ ;\ \varphi_{max}\ \rangle \qquad (2-15a)$$

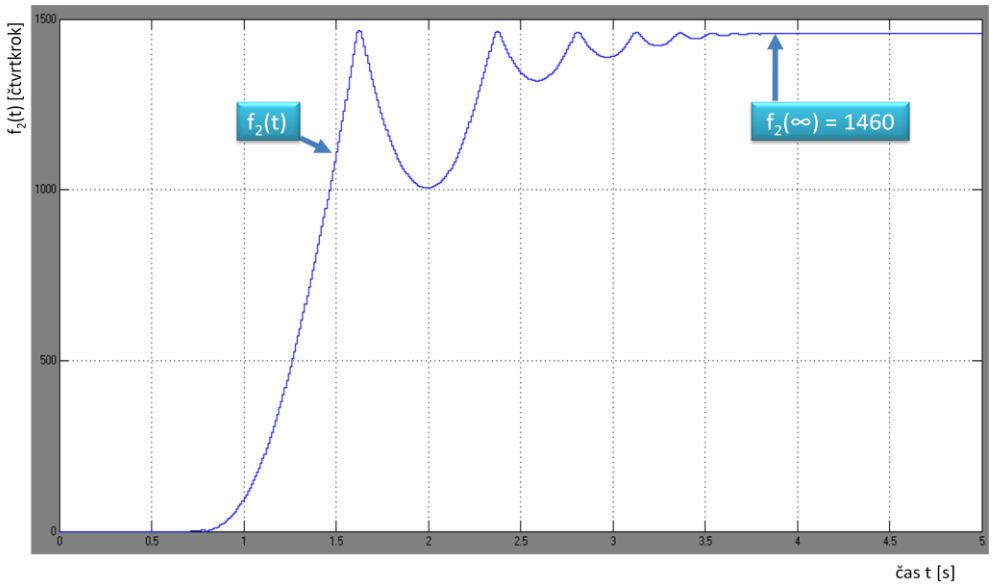
ve stupních

$$\varphi \in \langle -140; +140 \rangle \qquad (2-15b)$$

Rozsah azimutu je dán relací (v radiánech):

$$\varphi_{FS} = \varphi_{max} - \varphi_{min} = \frac{140 \cdot \pi}{180} - \left(-\frac{140 \cdot \pi}{180}\right) = 2 \cdot \frac{140 \cdot \pi}{180} = \frac{140 \cdot \pi}{90} = \frac{14}{9} \cdot \pi \qquad (2-15c)$$

Nyní je třeba určit počet kroků azimutového inkrementálního senzoru v ustáleném stavu odezvy modelu na maximální velikost budícího napětí $U_T = -1{,}0\ [MU] = -6{,}0\ [V]$.



Obr. 2.2.8: odezva modelu na maximální velikost budícího napětí ocasního motoru $U_T = -1{,}0\ [MU] = -6{,}0\ [V]$, při $U_B = -0{,}85\ [V]$, $U_M = 0{,}0\ [MU] = 0{,}0\ [V]$ a X4 detekci kvadraturních signálů

Z odezvy na obr. 2.2.8 je zřejmé, že ustálený stav této charakteristiky odpovídá hodnotě (pro čtvrtinový krok a celý krok):

$$f_2(t = +\infty) = 1460\ [čtvrtkroky] = \frac{1460}{4}\ [kroky] = 365\ [kroky] = 365\ [MP] \qquad (2-15d)$$

Pro přepočet mezi jednotkami [rad] a [MP] lze určit konstantu $k_2 = k_\varphi$ dle relace:

pro celý krok

$$k_2 = \frac{\varphi_{FS}}{f_2(\infty)} = \frac{14{\cdot}\pi}{9} \cdot \frac{1}{365} = \frac{14{\cdot}\pi}{3285}\ [rad \cdot MP^{-1}] \cong \frac{\pi}{256}\ [rad \cdot MP^{-1}] \qquad (2-15e)$$

pro čtvrtinový krok

$$k_2 = \frac{1}{4} \cdot \frac{\varphi_{FS}}{f_2(\infty)} = \frac{1}{4} \cdot \frac{14{\cdot}\pi}{3285} = \frac{14{\cdot}\pi}{13140}\ [rad \cdot MP^{-1}] \cong \frac{\pi}{1024}\ [rad \cdot MP^{-1}] \qquad (2-15f)$$

Posunutí funkční hodnoty (ofset) posouvá nulový výstup senzoru z nulové vertikální pozice tak, aby byl nulový signál uprostřed výstupní hodnoty úhlu azimutu y_φ. Musíme tedy odečíst polovinu rozsahu azimutu φ_{FS}:

$$y_{\varphi_0} = -\frac{\varphi_{FS}}{2} = -\frac{14{\cdot}\pi}{9} \cdot \frac{1}{2} = -\frac{14{\cdot}\pi}{18} = -\frac{7}{9} \cdot \pi\ [rad] \qquad (2-15g)$$

Výsledná lineární algebraická rovnice azimutového senzoru je dána relací (pro čtvrtinový krok):

při naměřené odezvě

$$y_\varphi = k_\varphi \cdot \varphi + y_{\varphi_0} = \frac{14{\cdot}\pi}{13140} \cdot \varphi - \frac{7{\cdot}\pi}{9} = \frac{\pi}{13140} \cdot \left(14 \cdot \varphi - \frac{91980}{9}\right)\ [rad] \qquad (2-15h)$$

při aproximaci

$$y_\varphi = k_\varphi \cdot \varphi + y_{\varphi_0} = \frac{\pi}{1024} \cdot \varphi - \frac{7 \cdot \pi}{9} = \frac{\pi \cdot \varphi - \frac{7168}{9} \pi}{1024} = \frac{\pi}{1024} \cdot \left(\varphi - \frac{7168}{9}\right) [rad] \qquad (2-15ch)$$

Do rovnic (2 – 15h) a (2 – 15ch) lze za φ dosadit funkční hodnotu z grafické závislosti zobrazené na obr. 2.2.8, neboli dané rovnice jsou přizpůsobeny pro detekci X4 (čtvrtinový krok).

Pokud bychom rovnici (2 -15h), respektive (2 – 15ch) analyzovali, dostaneme tyto výsledky:

$\boldsymbol{\varphi}$	$\boldsymbol{y_\varphi}$
$[čtvrtkroky]$	$[rad]$
0	$-7 \cdot \pi/9$
730	0
1460	$+7 \cdot \pi/9$

Tab. 2.2.2b: závislost výstupní hodnoty úhlu azimutu na azimutovém úhlu při detekci X4

Přepočty výstupních veličin pro celý krok (detekce X1), získané výše popsaným způsobem, jsou následující dle [1] [3] [5]:

Rozsahy a konverzní konstanty			
Výstupní veličiny	**Rozsah ve strojových jednotkách**		**Konverzní konstanta**
	$[MP]$	$[MP]$	$[rad \cdot MP^{-1}]$
$\boldsymbol{\psi}$	0	135	Π / 256
$\boldsymbol{\varphi}$	0	365	Π / 256
Ofsety IRC senzorů			
Označení	**Hodnota**		**Jednotka**
$\boldsymbol{y_{\psi_0}}$	- Π / 4		$[rad]$
$\boldsymbol{y_{\varphi_0}}$	- 7 · Π / 9		$[rad]$

Tab. 2.2.2c: rozsahy ve strojových jednotkách a konverzní konstanty pro výstupní úhly – elevace (z enkodéru), azimut (z enkodéru) při detekci X1; ofsety IRC senzorů

Přepočty výstupních veličin pro čtvrtinový krok (detekce X4), získané výše popsaným způsobem, jsou následující dle [1] [3] [5]:

Rozsahy a konverzní konstanty			
Výstupní veličiny	**Rozsah ve strojových jednotkách**		**Konverzní konstanta**
	$[čtvrtkroky]$	$[čtvrtkroky]$	$[rad \cdot čtvrtkrok^{-1}]$
$\boldsymbol{\psi}$	0	540	Π / 1024
$\boldsymbol{\varphi}$	0	1460	Π / 1024
Ofsety IRC senzorů			
Označení	**Hodnota**		**Jednotka**
$\boldsymbol{y_{\psi_0}}$	- Π / 4		$[rad]$
$\boldsymbol{y_{\varphi_0}}$	- 7 · Π / 9		$[rad]$

Tab. 2.2.2d: rozsahy ve strojových jednotkách a konverzní konstanty pro výstupní úhly – elevace (z enkodéru), azimut (z enkodéru) při detekci X4; ofsety IRC senzorů

Identifikace momentu tíhové síly

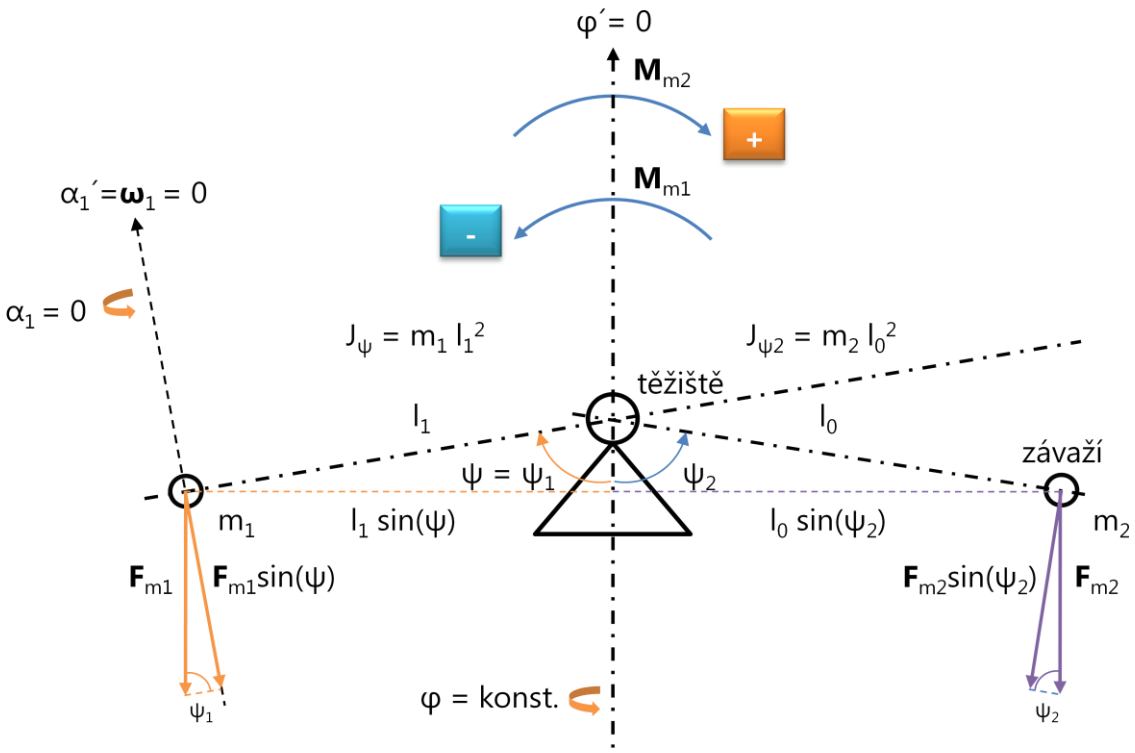
Cílem je identifikovat moment síly, vytvářeného hmotou umístěnou v těžišti těla vrtulníku dle rovnice pro rovnováhu sil – těžiště je umístěno ve své referenční pozici a tělo vrtulníku musí být fixně zajištěno

v azimutu, jehož úhel může být i nulový (viz obr. 2.2.9). Závažím o hmotnosti m_2 a vzdáleného od těžiště ve vzdálenosti l_0 se snažíme tuto soustavu vybalancovat, tedy dle [1]:

$$M_{m_2} - M_{m_1} = 0 \qquad (2-16a)$$

po úpravě

$$M_{m_1} = M_{m_2} \qquad (2-16b)$$



Obr. 2.2.9: schéma dynamiky v elevaci – význam těžiště, silových momentů a analýza sil působících na tělo vrtulníku (hmotný bod) a přídavné závaží při ustáleném stavu v azimutu

Z rovnice (2 – 16b) a obr. 2.2.9 je zřejmé, že obecně platí tyto matematické relace, tedy dle [1]:

$$m_1 \cdot g \cdot l_1 \cdot sin(\psi_1) = m_2 \cdot g \cdot l_0 \cdot sin(\psi_2) \qquad (2-16a)$$

po úpravě

$$M_{g_1} = m_2 \cdot g \cdot l_0 \cdot \frac{sin(\psi_2)}{sin(\psi_1)} = 9{,}80665 \cdot m_2 \cdot l_0 \cdot \frac{sin(\psi_2)}{sin(\psi_1)} \qquad (2-16b)$$

za předpokladu

$$sin(\psi_1) = sin(\psi) \neq 0 \Rightarrow \psi \in \mathbb{R} - \{k \cdot \pi\}; k \in \mathbb{Z} \qquad (2-16c)$$

Rovnice (2 – 16c) je v případě modelu splněna, protože platí relace (2 – 14a) definující platný rozsah úhlu elevace ψ.

Zajistíme-li vhodným zavěšením závaží následující relaci:

$$\psi_2 = \frac{\pi}{2} \Rightarrow sin(\psi_2) = sin\left(\frac{\pi}{2}\right) = 1 \qquad (2-16d)$$

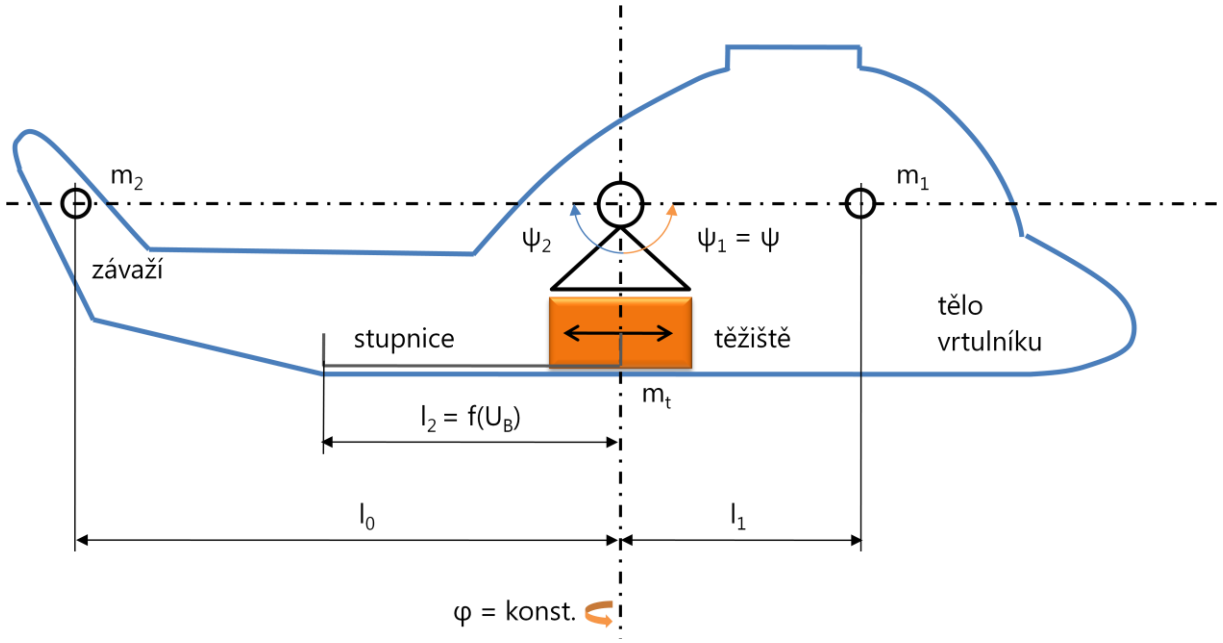
pak přirozeně platí

$$\psi_1 = \frac{\pi}{2} \Rightarrow sin(\psi_1) = sin\left(\frac{\pi}{2}\right) = 1 \qquad (2-16e)$$

Výše uvedeným postupem při splnění předpokladů jsme dosáhli redukce relace (2 – 16b) na tvar dle [1]:

$$M_{g_1} = m_2 \cdot g \cdot l_0 \cdot \frac{sin(\psi_2)}{sin(\psi_1)} = 9{,}80665 \cdot m_2 \cdot l_0 \qquad (2-16f)$$

Relace (2 – 16f) odpovídá úloze rovnosti momentů sil na páce. Nyní již zbývá empiricky určit hmotnost závaží m_2 a jeho vzdálenost od středu těžiště l_0.



Obr. 2.2.10: vyvážení těla vrtulníku v elevaci pomocí přídavného závaží a vyznačení příslušných veličin; těžiště se nachází v referenční poloze

Číselné hodnoty empiricky identifikovaných veličin jsou uvedeny následující tabulce:

Identifikace momentu tíhové síly		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{l_0}$	$14{,}25 \cdot 10^{-2}$	$[m]$
$\boldsymbol{l_1}$	$7{,}70 \cdot 10^{-2}$	$[m]$
$\boldsymbol{l_2 = f(U_B)}$	0 až $5{,}40 \cdot 10^{-2}$	$[m]$
$\boldsymbol{m_1}$	$10{,}00 \cdot 10^{-2}$	$[kg]$
$\boldsymbol{m_2}$	$6{,}00 \cdot 10^{-2}$	$[kg]$
$\boldsymbol{M_{g_1}}$	$8{,}39 \cdot 10^{-2}$	$[N \cdot m]$

Tab. 2.2.3: číselné hodnoty empiricky identifikovaných veličin

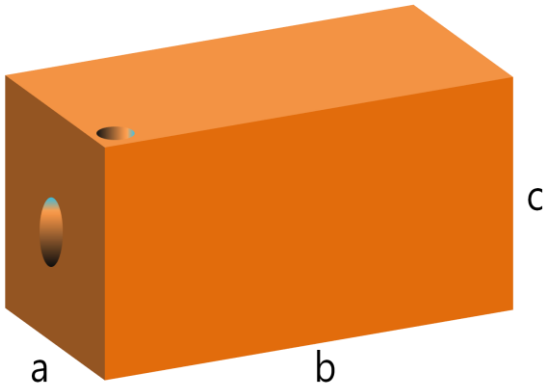
Výsledný moment tíhové síly je po dosazení do (2 – 16f) dán hodnotou:

$$M_{g_1} = m_2 \cdot g \cdot l_0 = 0{,}06 \cdot 9{,}80665 \cdot 0{,}1425 = 8{,}38847 \cdot 10^{-2}\ [N \cdot m] \qquad (2-16g)$$

Procedura identifikace je sice jednoduchá, ale nesprávná hodnota může ovlivnit určení ostatních parametrů hlavně elevaci. A protože vazba mezi elevací a azimutem je významná, bude ovlivněn i azimut. [1]

Identifikace relace mezi polohou těžiště a přiloženým napětím U_B

Těžiště se v modelu nachází v levé zadní části těla vrtulníku. Jedná se o kovový kvádr s otvorem pro vodící tyč a závitem pro přišroubování ramene spojeného se servomechanismem.



Obr. 2.2.11: 3D vyobrazení těžiště modelu vrtulníku s označením rozměrů tělesa – šířka (a), délka (b) a výška (c)

Údaje o rozměrech a hmotnosti jsou uvedeny v následující tabulce:

Identifikace relace mezi polohou těžiště a přiloženým napětím		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{a}$	$2{,}00 \cdot 10^{-2}$	$[m]$
$\boldsymbol{b}$	$3{,}50 \cdot 10^{-2}$	$[m]$
$\boldsymbol{c}$	$2{,}00 \cdot 10^{-2}$	$[m]$
$\boldsymbol{m_t}$	$10{,}00 \cdot 10^{-2}$	$[kg]$

Tab. 2.2.4: rozměry a hmotnost těžiště modelu vrtulníku

Pozici těžiště, respektive velikost l_2 (závislá proměnná), lze u modelu měnit DC napětím servomechanismu U_B (nezávislá proměnná). Obr. 2.2.10 ukazuje vyvážení těla vrtulníku v elevaci při:

- **vyvážené výstupní hodnotě elevace** $y_\psi = 0\ [rad]$
- **zafixované výstupní hodnotě azimutu** $y_\varphi = konst.\,[rad]$
  $y_\varphi = 0\ [rad]$
- **nulové (referenční) pozici těžiště** $l_{2\,(ref.)} = f\left(U_{B_{min}}\right) = 0\ [m]$

Charakteristika $l_2 = f(U_B)$ byla zjištěna empiricky, kdy při dané hodnotě napětí U_B se těžiště v horizontálním směru posunulo o přírůstek Δl_2 dle relace:

$$\Delta l_2 = l_2 - l_{2\,(ref.)} = f(U_B) - f\left(U_{B_{min}}\right) = l_2 - 0 = l_2 \qquad (2-17a)$$

velikost intervalu posunu těžiště s číselným vyjádřením

$$l_{2\,(FS)} = f\left(U_{B_{max}}\right) - f\left(U_{B_{min}}\right) = 0{,}054 - 0 = 0{,}054\ [m] \qquad (2-17b)$$

kde $\Delta l_2 = l_2 = f(U_B)$ délkový přírůstek od středu těžiště v závislosti na U_B $[m]$

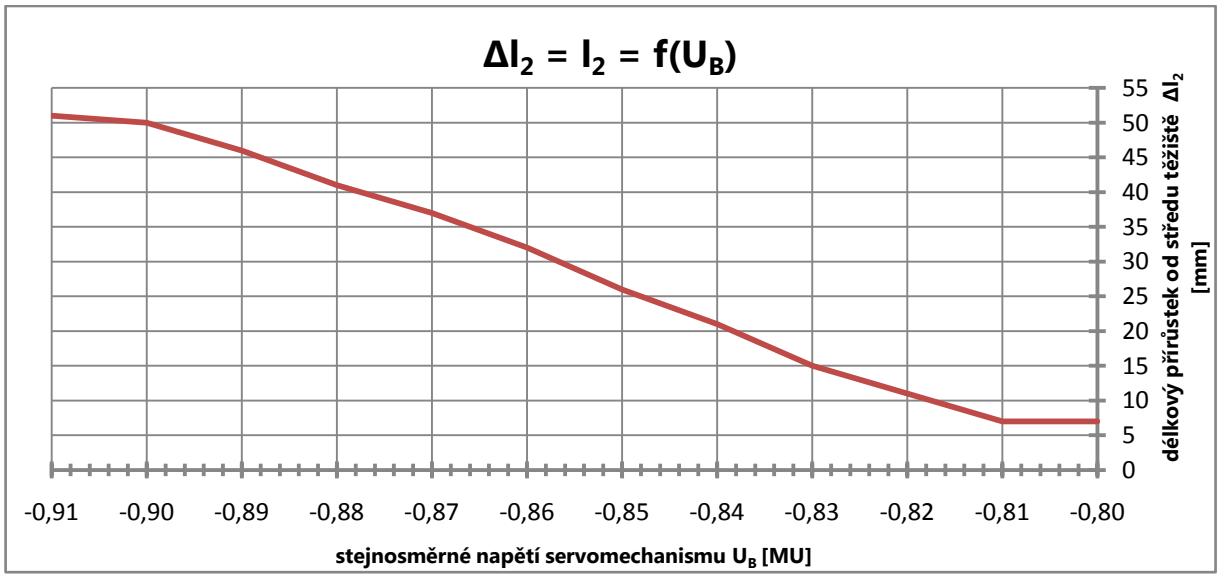
$l_{2\,(ref.)} = f(U_{B_{min}})$	referenční vzdálenost středu těžiště v závislosti na $U_{B_{min}}$	$[m]$
$l_{2\,(FS)}$	velikost intervalu posunu těžiště	$[m]$
U_B	obecná velikost DC napětí servomechanismu	$[MU]$
$U_{B_{min}}$	minimální velikost DC napětí servomechanismu	$[MU]$
$U_{B_{max}}$	maximální velikost DC napětí servomechanismu	$[MU]$

Z relace (2 – 17a) a obr. 2.2.10 je zřejmé, že délkový přírůstek Δl_2 je identický s l_2, neboť střed těžiště zároveň představuje jeho nulovou neboli referenční pozici $l_{2\,(ref.)}$ – jedná se o konstrukční uspořádání v modelu.

U_B	$\Delta l_2 = l_2$	$l_0 - l_2$	$l_1 + l_2$	$(l_1 + l_2)/U_B$	k_B
$[MU]$	$[mm]$	$[mm]$	$[mm]$	$[m \cdot MU^{-1}]$	$[N \cdot m \cdot V^{-1}]$
-0,80	7,0	135,5	84,0	-0,105	-0,103
-0,81	7,0	135,5	84,0	-0,104	-0,102
-0,82	11,0	131,5	88,0	-0,107	-0,105
-0,83	15,0	127,5	92,0	-0,111	-0,109
-0,84	21,0	121,5	98,0	-0,117	-0,115
-0,85	26,0	116,5	103,0	-0,121	-0,119
-0,86	32,0	110,5	109,0	-0,127	-0,125
-0,87	37,0	105,5	114,0	-0,131	-0,128
-0,88	41,0	101,5	118,0	-0,134	-0,131
-0,89	46,0	96,5	123,0	-0,138	-0,135
-0,90	50,0	92,5	127,0	-0,141	-0,138
-0,91	51,0	91,5	128,0	-0,141	-0,138
Arit. průměr	--------	--------	--------	--------	-0,121

Tab. 2.2.5: tabelované naměřené hodnoty délkových přírůstků Δl_2 při různých nastavených hodnotách napětí servomechanismu U_B; rozdíl vůči l_0, součet s l_1 a aritmetický (výběrový) průměr pro hodnotu přenosové konstanty k_B

Tabelované hodnoty délkových přírůstků od středu těžiště lze reprezentovat také graficky:



Obr. 2.2.12: grafická reprezentace závislosti $l_2 = f(U_B)$

Identifikace soustavy hlavního motoru a vrtule v empirickém modelu

Před identifikací dynamiky motoru a těla vrtulníku musí být známy statické charakteristiky odvozené z následující aproximační relace pro elevaci:

$$M_{1\,(emp.)} = a_1 \cdot u_{1\,(mot.)}{}^2(t) + b_1 \cdot u_{1\,(mot.)}(t) = a_1 \cdot U_{1\,(mot.)}^2 + b_1 \cdot U_{1\,(mot.)} \qquad (2-18a)$$

kde			
	$M_{1\,(emp.)}$	moment síly vrtule elevačního motoru v emp. modelu	$[N \cdot m]$
	$U_{1\,(mot.)}$	výstupní napětí hlavního motoru	$[MU]$
	a_1	přepočtová konstanta kvadratického členu	$[N \cdot m \cdot MU^{-2}]$
	b_1	přepočtová konstanta lineárního členu	$[N \cdot m \cdot MU^{-1}]$

Identifikace obou přepočtových konstant závisí na měření funkční hodnoty výstupního elevačního úhlu y_ψ v ustáleném stavu pro různé hodnoty vstupního napětí U_M. Stejně jako v případě identifikace momentu tíhové síly lze pro vyvážení těla vrtulníku v okolí vertikální pozice využít závaží o hmotnosti m_2 a umístěného ve vzdálenosti l_0, přičemž platí následující relace dle [1]:

$$M_{g\,(emp.)} = \left[M_{m_1} + M_{m_2}\right] \cdot sin\left\{\frac{y_\psi(U_M) - y_{\psi 0}}{k_\psi}\right\} = [m_1 \cdot g \cdot l_1 + m_2 \cdot g \cdot l_0] \cdot sin(\psi) \qquad (2-18b)$$

kde			
	$M_{g\,(emp.)}$	moment tíhové síly v empirickém modelu	$[N \cdot m]$
	M_{m_1}	moment tíhové síly hmotného bodu vrtulníku	$[N \cdot m]$
	M_{m_2}	moment tíhové síly závaží	$[N \cdot m]$
	$y_\psi(U_M)$	výstupní elevační úhel v závislosti na U_M	$[rad]$

Výsledná rovnice statické charakteristiky je po dosazení dána relací dle [1]:

$$M_{1\,(emp.)} = 0{,}1050 \cdot U_{1\,(mot.)}^2 + 0{,}00936 \cdot U_{1\,(mot.)}\ [N \cdot m] \qquad (2-18c)$$

Výše uvedená relace formálně odpovídá vztahu (2 − 05b) v teoretickém modelu. Dynamika těla vrtulníku je relativně pomalejší oproti dynamice soustavy hlavní motor-vrtule, kterou lze namodelovat například lineárním aperiodickým systémem druhého řádu ve tvaru přenosové funkce, tedy dle [1]:

$$\hat{G}_1(s) = \frac{\hat{L}\{u_{1\,(mot.)}(t)\}}{\hat{L}\{u_1(t)\}} = \frac{\hat{L}\{U_{1\,(mot.)}\}}{\hat{L}\{U_M\}} = \frac{\hat{U}_{1\,(mot.)}(s)}{\hat{U}_M(s)} = \frac{1}{(T_1 \cdot s+1)^2} \qquad (2-19a)$$

kde		
	$\hat{U}_M(s)$	Laplaceův obraz vstupního napětí hlavního motoru
	$\hat{U}_{1\,(mot.)}(s)$	Laplaceův obraz výstupního napětí hlavního motoru
	T_1	časová konstanta aproximovaného procesu $[s]$

Výsledná přenosová funkce dynamiky hlavní motor-vrtule je dána relací dle [1]:

$$\hat{G}_1(s) = \frac{1}{(T_1 \cdot s+1)^2} = \frac{1}{(0{,}2500 \cdot s+1)^2} = \frac{1}{0{,}0625 \cdot s^2 + 0{,}5000 \cdot s+1} \qquad (2-19b)$$

Identifikace hlavního motoru a vrtule v empirickém modelu		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{a_1}$	0,10500	$[N \cdot m \cdot MU^{-2}]$
$\boldsymbol{b_1}$	0,00936	$[N \cdot m \cdot MU^{-1}]$
$\boldsymbol{T_1}$	0,25000	$[s]$

Tab. 2.2.6: empiricky identifikované parametry soustavy hlavního motoru a vrtule

Identifikace soustavy ocasního motoru a vrtule v empirickém modelu

Před identifikací dynamiky motoru a těla vrtulníku musí být známy statické charakteristiky odvozené z následující aproximační relace pro azimut:

$$M_{2\ (emp.)} = a_2 \cdot u_{2\ (mot.)}{}^2(t) + b_2 \cdot u_{2\ (mot.)}(t) = a_2 \cdot U_{2\ (mot.)}^2 + b_2 \cdot U_{2\ (mot.)} \qquad (2-20a)$$

kde			
	$M_{2\ (emp.)}$	moment síly vrtule azimut. motoru v emp. modelu	$[N \cdot m]$
	$U_{2\ (mot.)}$	výstupní napětí ocasního motoru	$[MU]$
	a_2	přepočtová konstanta kvadratického členu	$[N \cdot m \cdot MU^{-2}]$
	b_2	přepočtová konstanta lineárního členu	$[N \cdot m \cdot MU^{-1}]$

Výsledná rovnice statické charakteristiky je po dosazení dána relací dle [1]:

$$M_{2\ (emp.)} = 0{,}0330 \cdot U_{2\ (mot.)}^2 + 0{,}0294 \cdot U_{2\ (mot.)}\ [N \cdot m] \qquad (2-20b)$$

Výše uvedená relace formálně odpovídá vztahu (2 − 07b) v teoretickém modelu. Dynamika těla vrtulníku je relativně pomalejší oproti dynamice soustavy ocasní motor-vrtule, kterou lze namodelovat například lineárním aperiodickým systémem druhého řádu ve tvaru přenosové funkce, tedy dle [1]:

$$\hat{G}_2(s) = \frac{\hat{L}\{u_{2\ (mot.)}(t)\}}{\hat{L}\{u_2(t)\}} = \frac{\hat{L}\{U_{2\ (mot.)}\}}{\hat{L}\{U_T\}} = \frac{\hat{U}_{2\ (mot.)}(s)}{\hat{U}_T(s)} = \frac{1}{(T_2 \cdot s+1)^2} \qquad (2-21a)$$

kde		
	$\hat{U}_T(s)$	Laplaceův obraz vstupního napětí ocasního motoru
	$\hat{U}_{2\ (mot.)}(s)$	Laplaceův obraz výstupního napětí ocasního motoru
	T_2	časová konstanta aproximovaného procesu $[s]$

Výsledná přenosová funkce dynamiky hlavní motor-vrtule je dána relací dle [1]:

$$\hat{G}_2(s) = \frac{1}{(T_2 \cdot s+1)^2} = \frac{1}{(0{,}25 \cdot s+1)^2} = \frac{1}{0{,}0625 \cdot s^2+0{,}2500 \cdot s+1} \qquad (2-21b)$$

Identifikace ocasního motoru a vrtule v empirickém modelu		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{a_2}$	0,0330	$[N \cdot m \cdot MU^{-2}]$
$\boldsymbol{b_2}$	0,0294	$[N \cdot m \cdot MU^{-1}]$
$\boldsymbol{T_2}$	0,2500	$[s]$

Tab. 2.2.7: empiricky identifikované parametry soustavy ocasního motoru a vrtule

Identifikace reakční křížové vazby hlavního motoru

Tato identifikace dynamiky křížové vazby závisí na reakčním momentu síly vrtule hlavního motoru dle relace podle [1] [5]:

$$M_{r\ (emp.)} = a_r \cdot u_{1\ (reak.)}{}^2(t) + b_r \cdot u_{1\ (reak.)}(t) = a_r \cdot U_{1\ (reak.)}^2 + b_r \cdot U_{1\ (reak.)} \qquad (2-22a)$$

kde			
	$M_{r\ (emp.)}$	moment síly reakční křížové vazby v emp. modelu	$[N \cdot m]$
	$U_{1\ (reak.)}$	výstupní reakční napětí hlavního motoru	$[MU]$
	a_r	reakční přepočtová konstanta kvadratického členu	$[N \cdot m \cdot MU^{-2}]$
	b_r	reakční přepočtová konstanta lineárního členu	$[N \cdot m \cdot MU^{-1}]$

Výsledná rovnice statické charakteristiky je po dosazení dána relací:

$$M_{r\ (emp.)} = 0{,}0148 \cdot U_{1\ (reak.)}^2 + 0{,}0108 \cdot U_{1\ (reak.)}\ [N \cdot m] \qquad (2-22b)$$

Lineární model lze vyjádřit následující přenosovou funkcí druhého řádu obsahující dvě nuly v čitateli a dva póly ve jmenovateli přenosové funkce, tedy:

$$\hat{G}_r(s) = \frac{\hat{L}\left\{u_{1(reak.)}(t)\right\}}{\hat{L}\{u_1(t)\}} = \frac{\hat{L}\left\{U_{1(reak.)}\right\}}{\hat{L}\{U_M\}} = \frac{\hat{U}_r(s)}{\hat{U}_M(s)} = \frac{T_{r_1}\cdot s^2 + T_{r_2}\cdot s + 1}{(T_1\cdot s+1)^2} \qquad (2-23a)$$

kde

$\hat{U}_r(s)$	Laplaceův obraz reakčního napětí	
$\hat{U}_M(s)$	Laplaceův obraz vstupního napětí hlavního motoru	
T_{r_1}	první reakční časová konstanta čitatele přenosu	$[s]$
T_{r_2}	druhá reakční časová konstanta čitatele přenosu	$[s]$
T_1	časová konstanta aproximovaného procesu	$[s]$

Výsledná přenosová funkce reakční křížové vazby je dána relací:

$$\hat{G}_r(s) = \frac{T_{r_1}\cdot s^2 + T_{r_2}\cdot s + 1}{(T_1\cdot s+1)^2} = \frac{0{,}0017\cdot s^2 + 0{,}1908\cdot s + 1}{(0{,}25\cdot s+1)^2} \qquad (2-23b)$$

Identifikace reakční křížové vazby hlavního motoru		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{a_r}$	0,0148	$[N \cdot m \cdot MU^{-2}]$
$\boldsymbol{b_r}$	0,0108	$[N \cdot m \cdot MU^{-1}]$
$\boldsymbol{T_{r_1}}$	0,0017	$[s]$
$\boldsymbol{T_{r_2}}$	0,1908	$[s]$
$\boldsymbol{T_1}$	0,2500	$[s]$

Tab. 2.2.8: empiricky identifikované parametry v reakční křížové vazby

Identifikace dynamiky těla vrtulníku v elevaci

Parametry v elevaci (vertikální rovina) se odhadují experimentálně při podmínkách dle [1]:

- oba motory jsou vypnuty
- azimut je zafixován
- elevační IRC senzor je zkalibrován

Samotná identifikace je založena na adaptační offline identifikaci, vhodné pro nelineární systémy. V tomto případě je využita statická minimalizace účelové funkce (Nelderova-Meadova simplexová metoda), uvažující sumu kvadrátů odchylek mezi skutečnou odezvou a namodelovanou odezvou dle relace dle [1]:

$$J_{elev.} = min_{\left(J_\psi\ B_{\psi_1} B_{\psi_2}\right)} \sum_{k=1}^{N} \left\{ y_{reál.}(k, \boldsymbol{U}_M) - y_{model}\left(k, \boldsymbol{U}_M, J_\psi, B_{\psi_1}, B_{\psi_2}\right)\right\}^2 \qquad (2-24a)$$

kde

$J_{elev.}$	účelová (ztrátová) funkce optimalizačního procesu v elevaci	
J_ψ	moment setrvačnosti těla vrtulníku kolem horizontální osy	$[kg \cdot m^2]$
B_{ψ_1}	moment hybnosti těla vrtulníku (bez uvažování směru)	$[kg \cdot m^2 \cdot s^{-1}]$
B_{ψ_2}	moment hybnosti těla vrtulníku (s uvažováním směru)	$[kg \cdot m^2 \cdot s^{-1}]$
k	index sčítance v sumě	
$\boldsymbol{U}_M$	jednorozměrný vektor DC napětí hlavního motoru	$[V]$

$y_{reál.}$ odezva reálného systému v elevaci
y_{model} odezva modelu systému v elevaci

Číselné hodnoty empiricky identifikovaných veličin jsou uvedeny následující tabulce dle [1]:

Identifikace dynamiky těla vrtulníku v elevaci		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{J_\psi}$	$4{,}370 \cdot 10^{-3}$	$[kg \cdot m^2]$
$\boldsymbol{B_{\psi_1}}$	$1{,}840 \cdot 10^{-3}$	$[kg \cdot m^2 \cdot s^{-1}]$
$\boldsymbol{B_{\psi_2}}$	$0{,}103 \cdot 10^{-3}$	$[kg \cdot m^2 \cdot s^{-1}]$

Tab. 2.2.9: empiricky identifikované parametry v elevaci

Identifikace dynamiky těla vrtulníku v azimutu

Parametry v azimutu (horizontální rovina) se odhadují experimentálně při podmínkách dle [1]:

- oba motory jsou vypnuty
- elevace je zafixována
- azimutový IRC senzor je zkalibrován

Samotná identifikace je založena na adaptační offline identifikaci, vhodné pro nelineární systémy. V tomto případě je využita statická minimalizace účelové funkce (Nelderova-Meadova simplexová metoda), uvažující sumu kvadrátů odchylek mezi skutečnou odezvou a namodelovanou odezvou dle relace dle [1]:

$$J_{azim.} = min_{\left(J_\varphi B_{\varphi_1} B_{\varphi_2}\right)} \sum_{k=1}^{N} \left\{ y_{reál.}(k, \boldsymbol{U}_T) - y_{model}\left(k, \boldsymbol{U}_T, J_\varphi, B_{\varphi_1}, B_{\varphi_2}\right) \right\}^2 \qquad (2-25a)$$

kde $J_{azim.}$ účelová (ztrátová) funkce optimalizačního procesu v azimutu
J_φ moment setrvačnosti těla vrtulníku kolem vertikální osy $[kg \cdot m^2]$
B_{φ_1} moment hybnosti těla vrtulníku (bez uvažování směru) $[kg \cdot m^2 \cdot s^{-1}]$
B_{φ_2} moment hybnosti těla vrtulníku (s uvažováním směru) $[kg \cdot m^2 \cdot s^{-1}]$
k index sčítance v sumě
$\boldsymbol{U}_T$ jednorozměrný vektor DC napětí ocasního motoru $[V]$
$y_{reál.}$ odezva reálného systému v azimutu
y_{model} odezva modelu systému v azimutu

Číselné hodnoty empiricky identifikovaných veličin jsou uvedeny následující tabulce dle [1]:

Identifikace dynamiky těla vrtulníku v azimutu		
Označení	**Hodnota**	**Jednotka**
$\boldsymbol{J_\varphi}$	$4{,}140 \cdot 10^{-3}$	$[kg \cdot m^2]$
$\boldsymbol{B_{\varphi_1}}$	$8{,}690 \cdot 10^{-6}$	$[kg \cdot m^2 \cdot s^{-1}]$
$\boldsymbol{B_{\varphi_2}}$	$0{,}596 \cdot 10^{-3}$	$[kg \cdot m^2 \cdot s^{-1}]$

Tab. 2.2.10: empiricky identifikované parametry v azimutu

Identifikace gyroskopické křížové vazby

Předpokládejme, že dynamika hlavního motoru (v elevaci) je rychlejší než dynamika těla vrtulníku v azimutu, tudíž jeden ze vstupů gyroskopické vazby – původně závislý na úhlové rychlosti hlavní vrtule ω_1 (viz obr. 2.2.5) – by mohl být přímo připojen na DC napětí hlavního motoru U_M, jehož velikost, stejně jako

velikost ω_1, se během testování nemění. Takto lze systém stabilizovat na určité hodnotě úhlu elevace ψ. Skoky momentu gyroskopické síly M_{gyro} se provádějí ručně, a to změnou úhlu azimutu φ. Po několika cyklech se sekvencí „STOP – pohyb ve směru hodinových ručiček – STOP – pohyb proti směru hodinových ručiček“ lze přibližně parametr K_{gyro} odhadnout ze změny elevačního úhlu ψ. [5]

Při vyjádření analytické relace pro odhad parametru K_{gyro} zanedbáváme v empirickém modelu elevace moment dostředivé síly $M_{\dot{\varphi}}$, moment třecí síly M_{f_1} a moment setrvačnosti těla vrtulníku kolem horizontální osy J_ψ. Platí tedy následující relace dle [1]:

v teoretickém modelu – relace (2 – 04a)

$$M_1 + M_{\dot{\varphi}} - M_{m_1} - M_{f_1} - M_{gyro} - J_\psi \cdot \ddot{\psi} = 0 \qquad (2-26a)$$

v empirickém modelu

$$M_{1(emp.)} - M_{m_1(emp.)} - M_{gyro\ (emp.)} = 0 \qquad (2-26b)$$

po dosazení do relace (2 – 26b)

$$a_1 \cdot U_{1\,(mot.)}^2 + b_1 \cdot U_{1\,(mot.)} - M_{g_1} \cdot sin(\psi) - K_{gyro\ (emp.)} \cdot \dot{\varphi} \cdot U_M \cdot cos(\psi) = 0 \qquad (2-26c)$$

kde

$M_{1\,(emp.)}$	moment síly vrtule elevačního motoru v emp. modelu	$[N \cdot m]$
$M_{m_1\,(emp.)}$	moment tíhové síly těla vrtulníku v emp. modelu	$[N \cdot m]$
$M_{gyro\ (emp.)}$	moment gyroskopické síly v emp. modelu	$[N \cdot m]$
$K_{gyro\ (emp.)}$	konstanta gyroskopické vazby v emp. modelu	$[N \cdot m \cdot s \cdot V^{-1}]$
U_M	DC napětí hlavního (elevačního) motoru	$[V]$
$U_{1\,(mot.)}$	výstupní napětí hlavního motoru	$[MU]$
a_1	přepočtová konstanta kvadratického členu	$[N \cdot m \cdot MU^{-2}]$
b_1	přepočtová konstanta lineárního členu	$[N \cdot m \cdot MU^{-1}]$
ψ	úhel elevace	$[rad]$
φ	úhel azimutu	$[rad]$
$\dot{\varphi}$	první derivace úhlu azimutu podle času	$[rad \cdot s^{-1}]$

Výsledná velikost konstanty gyroskopické vazby elevace-azimut je dána hodnotou:

$$K_{gyro\ (emp.)} = 1{,}50 \cdot 10^{-2}\ [N \cdot m \cdot s \cdot V^{-1}] \qquad (2-26d)$$

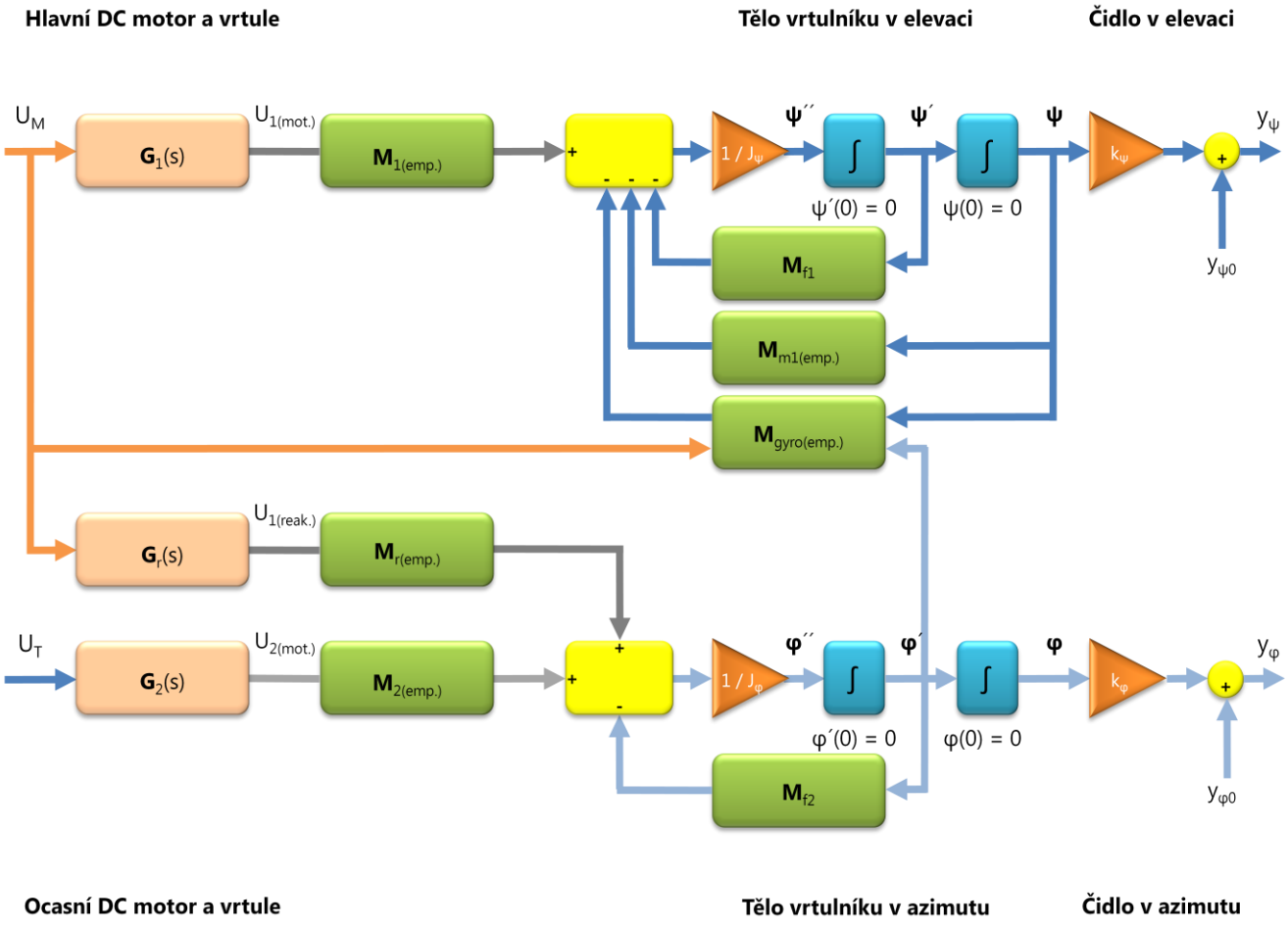
Celková dynamika systému – empirický model

Celková dynamika těla vrtulníku se v empirickém modelu výrazně zjednodušila oproti teoretickému modelu. Dynamiky motorů společně s reakční vazbou elevace-azimut jsou aproximovány lineárními přenosovými funkcemi druhého řádu. V elevaci se model zjednodušil díky předpokladům uvedeným v identifikaci gyroskopické křížové vazby. [1]

- hlavní (elevační) motor — *lineární přenosová funkce 2. řádu*
- reakční křížová vazba hlavního motoru — *lineární přenosová funkce 2. řádu; s 2 nulami*
- ocasní (azimutový) motor — *lineární přenosová funkce 2. řádu*
- vrtule — *lineárně-kvadratická aproximace*
- tělo vrtulníku v elevaci — *zjednodušený dynamický subsystém*

- tělo vrtulníku v azimutu *dynamický subsystém*
- čidlo v elevaci *statický subsystém*
- čidlo v azimutu *statický subsystém*

Sloučením všech těchto popisů lze vytvořit celkové empirické blokové schéma, tedy dle [1] [5]:



Obr. 2.2.13: blokové schéma kompletní dynamiky systému – empirický model

2.2.4 Linearizace

Z blokového schématu empirického modelu je zřejmé, že i přes značná zjednodušení v soustavě motor-vrtule obsahuje tyto elevační a azimutové nelinearity, tedy dle [1]:

v elevaci

$$M_{f_1} = B_{\psi_1} \cdot \dot{\psi} + B_{\psi_2} \cdot sign(\dot{\psi}) \qquad (2-27a)$$

$$M_{m_1\,(emp.)} = M_{g\,(emp.)} = M_{m_1} \cdot sin\left\{\frac{y_\psi(U_M) - y_{\psi 0}}{k_\psi}\right\} \qquad (2-27b)$$

$$M_{gyro\ (emp.)} = K_{gyro\ (emp.)} \cdot \dot{\varphi} \cdot U_M \cdot cos(\psi) \qquad (2-27c)$$

v azimutu

$$M_{f_2} = B_{\varphi_1} \cdot \dot{\varphi} + B_{\varphi_2} \cdot sign(\dot{\varphi}) \quad (2-27d)$$

Výběr pracovního bodu

Hlavním požadavkem pro výběr pracovního bodu je, aby elevační úhel a azimutový úhel byly konstantní, i když se motory otáčejí – je třeba dosáhnout ustáleného stavu. Budeme-li uvažovat fixní pozici těžiště vrtulníku dle [3]:

$$U_{B0} = konst. \quad (2-28a)$$

Pak budou všechny momenty setrvačnosti v obou osách rovněž konstantní, tedy budou platit následující podmínky, tedy dle [3]:

$$\omega_M = \omega_{M0} = \omega_1 = konst. \quad (2-28b)$$

$$\omega_T = \omega_{T0} = \omega_2 = konst. \quad (2-28c)$$

$$\psi = konst. \quad (2-28d)$$

$$\varphi = konst. \quad (2-28e)$$

$$\omega_H = \frac{d}{dt}\{\psi(t)\} = \omega_{H0} = \dot{\psi} = 0 \quad (2-28f)$$

$$\omega_V = \frac{d}{dt}\{\varphi(t)\} = \omega_{V0} = \dot{\varphi} = 0 \quad (2-28g)$$

Pro první derivace těchto stavových veličin pak přirozeně platí dle [3]:

$$\dot{\omega}_{M0} = \dot{\omega}_{T0} = \dot{\omega}_H = \dot{\omega}_V = \dot{\psi} = \dot{\varphi} = 0 \quad (2-28h)$$

Linearizace v elevaci

V relaci (2 – 27a) ignorujeme nelinearitu ve formě funkce signum (Coulombovo tření), tedy:

$$B_{\psi_2} \cdot sign(\dot{\psi}) = 0 \quad (2-29a)$$

z toho

$$M_{f_1(lin.)} = B_{\psi_1} \cdot \dot{\psi} + 0 = B_{\psi_1} \cdot \dot{\psi} \quad (2-29b)$$

V relaci (2 – 27b) představuje nelinearitu funkce sinus, kterou rozvineme pomocí Taylorova rozvoje dle relace dle [6]:

$$T_n(x) = \sum_{n=0}^{N} A_n \cdot (x - x_0)^n = \sum_{n=0}^{N} \frac{f^{(n)}(x=x_0)}{n!} \cdot (x - x_0)^n \quad (2-29c)$$

speciálně pro $x_0 = 0$ *(Maclaurinův polynom)*

$$M_n(x) = \sum_{n=0}^{N} A_n \cdot (x - 0)^n = \sum_{n=0}^{N} \frac{f^{(n)}(x=0)}{n!} \cdot x^n \quad (2-29d)$$

Pokud funkci $f(x)$ nahradíme Taylorovým polynem, je třeba si uvědomit, jak velké chyby se při aproximaci dopustíme. Jinými slovy odhadujeme rozdíl dle relace dle [6]:

$$R_n(x) = f(x) - T_n(x) = f(x) - \frac{f^{(n+1)}(\xi)}{(n+1)!} \cdot (x - x_0)^{n+1} \quad (2-29e)$$

z toho (Taylorův vzorec)

$$f(x) = T_n(x) + R_n(x) \quad (2-29f)$$

kde $T_n(x)$ Taylorův polynom n-tého řádu
$M_n(x)$ Maclaurinův polynom n-tého řádu
$f(x)$ obecná aproximovaná funkce
$R_n(x)$ Lagrangeův zbytek funkce $f(x)$ po n-tém členu polynomu
A_n n-tý člen Taylorova, respektive Maclaurinova polynomu
x_0 střed rozvoje čili střed konvergence řady, popř. zvolený pracovní bod
n stupeň derivace $f(x)$, exponent mocninné řady a argument faktoriálu

Předpokládáme, že daná funkce jedné proměnné $f(x)$ má v bodě x_0 derivace n-tého řádu. Pro další zkoumání takové funkce v blízkosti (pracovního) bodu provedeme aproximaci jednodušší funkcí ve formě polynomu, respektive konvergentní mocninné řady se středem konvergence řady v bodě x_0. Bude-li aproximační polynom prvého řádu, hovoříme o lineární aproximaci neboli linearizaci v daném pracovním bodě. [6]

Položme v relaci (2 – 27a):

$$sin\left\{\frac{y_\psi(U_M) - y_{\psi 0}}{k_\psi}\right\} = sin(\psi) = g(\psi) = g^{(0)}(\psi) \quad (2-29g)$$

z toho první derivace

$$g'(\psi) = g^{(1)}(\psi) = cos(\psi) \quad (2-29h)$$

Výsledná lineární aproximace se zanedbáním Lagrangeova zbytku je dána relací, tedy dle [6]:

$$T_1(\psi) = \frac{g^{(0)}(\psi_0)}{0!} \cdot (\psi - \psi_0)^0 + \frac{g^{(1)}(\psi_0)}{1!} \cdot (\psi - \psi_0)^1 = sin(\psi_0) + (\psi - \psi_0) \cdot cos(\psi_0) \quad (2-29ch)$$

Pro úplnost uveďme funkční závislost lineární aproximace ve významných elevačních bodech, tedy:

pro $\psi_0 = 0$

$$T_1(\psi)_{\psi_0=0} = sin(0) + (\psi - 0) \cdot cos(0) = 0 + \psi = \psi \quad (2-29i)$$

pro $\psi_0 = \frac{\pi}{4}$

$$T_1(\psi)_{\psi_0=\frac{\pi}{4}} = sin\left(\frac{\pi}{4}\right) + \left(\psi - \frac{\pi}{4}\right) \cdot cos\left(\frac{\pi}{4}\right) = \frac{\sqrt{2}}{2} \cdot \psi - \frac{\sqrt{2}}{2} \cdot \frac{4-\pi}{4} \quad (2-29j)$$

pro $\psi_0 = \frac{\pi}{2}$

$$T_1(\psi)_{\psi_0=\frac{\pi}{2}} = sin\left(\frac{\pi}{2}\right) + \left(\psi - \frac{\pi}{2}\right) \cdot cos\left(\frac{\pi}{2}\right) = 1 + 0 = 1 \quad (2-29k)$$

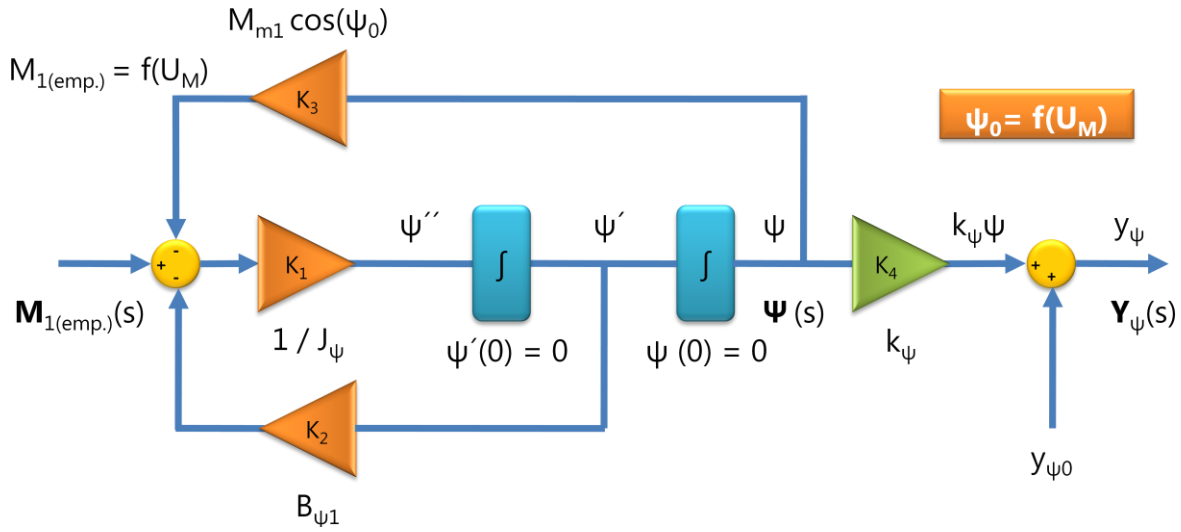
Výsledný linearizovaný moment tíhové síly v empirickém modelu je dán relací:

$$M_{g\ (lin.,emp.)} = M_{m_1} \cdot T_1(\psi) \cong M_{m_1} \cdot \psi \cdot cos(\psi_0) \qquad (2-29l)$$

V relaci (2 – 27c) představuje nelinearitu funkce kosinus, kterou opět můžeme rozvinout pomocí Taylorova rozvoje. Ovšem lze vyjít z předpokladů pro výběr pracovního bodu, konkrétně pak z relace (2 – 28g). Následně bude výsledný linearizovaný moment gyroskopické síly v empirickém modelu roven dle [1] [5]:

$$M_{gyro\ (lin.,emp.)} = K_{gyro\ (emp.)} \cdot 0 \cdot U_M \cdot cos(\psi) = 0 \qquad (2-29m)$$

Z výše uvedených linearizovaných vztahů lze sestavit celkové blokové schéma linearizace v elevaci (linearizovaná dynamika v elevaci a lineární senzorický subsystém), tedy dle [1]:



Obr. 2.2.14: celkové blokové schéma linearizované dynamiky okolo pracovního bodu v elevaci; bez závaží o hmotnosti m_2

Z obr. 2.2.14 lze rovněž sestavit obyčejnou lineární diferenciální rovnici druhého řádu pro linearizovanou dynamiku v elevaci, tedy:

$$\ddot{\psi} = \frac{1}{J_\psi} \cdot \left[M_{1(emp.)} - B_{\psi_1} \cdot \dot{\psi} - M_{m_1} \cdot cos(\psi_0) \cdot \psi\right] \qquad (2-30a)$$

po ekvivalentních úpravách

$$J_\psi \cdot \ddot{\psi} + B_{\psi_1} \cdot \dot{\psi} + M_{m_1} \cdot cos(\psi_0) \cdot \psi = M_{1(emp.)} = f(U_M) \qquad (2-30b)$$

z toho přenosová funkce (po Laplaceově transformaci a při nulových počátečních podmínkách)

$$\hat{G}_{elev.}(s) = \frac{\hat{\Psi}(s)}{\hat{M}_{1(emp.)}(s)} = \frac{1}{J_\psi \cdot s^2 + B_{\psi_1} \cdot s + M_{m_1} \cdot cos(\psi_0)} \qquad (2-30c)$$

Statický senzorický subsystém v elevaci je popsán lineární rovnicí – viz relace (2 – 12b):

$$y_\psi = k_\psi \cdot \psi + y_{\psi_0} \qquad (2-30d)$$

z toho Laplaceův obraz výstupního úhlu elevace (při nulových počátečních podmínkách)

$$\hat{Y}_\psi(s) = k_\psi \cdot \hat{\Psi}(s) + \frac{1}{s} \cdot y_{\psi_0} = k_\psi \cdot \hat{G}_{elev.}(s) \cdot \hat{M}_{1(emp.)}(s) + \frac{1}{s} \cdot y_{\psi_0} \qquad (2-30e)$$

po dosazení z relace (2 – 30c)

$$\hat{Y}_{\psi}(s) = k_{\psi} \cdot \frac{\hat{M}_{1\,(emp.)}(s)}{J_{\psi} \cdot s^2 + B_{\psi_1} \cdot s + M_{m_1} \cdot cos(\psi_0)} + \frac{1}{s} \cdot y_{\psi_0} \qquad (2-30f)$$

kde $\hat{M}_{1\,(emp.)}(s)$ Laplaceův obraz momentu síly vrtule elev. motoru v emp. modelu
$\hat{G}_{elev.}(s)$ přenosová funkce linearizované dynamiky v elevaci
$\hat{Y}_{\psi}(s)$ Laplaceův obraz výstupní hodnoty elevačního úhlu

Linearizace v azimutu

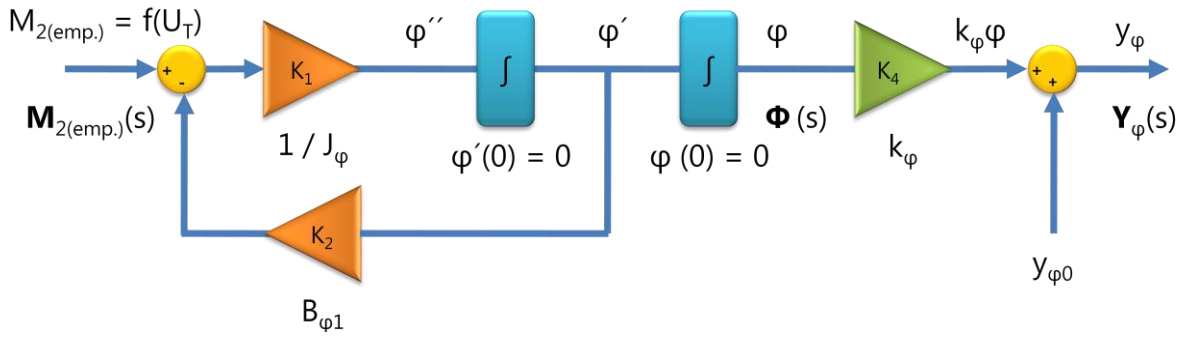
V relaci (2 – 27d) ignorujeme nelinearitu ve formě funkce signum (Coulombovo tření), tedy:

$$B_{\varphi_2} \cdot sign(\dot{\psi}) = 0 \qquad (2-31a)$$

z toho

$$M_{f_2(lin.)} = B_{\varphi_1} \cdot \dot{\varphi} + 0 = B_{\varphi_1} \cdot \dot{\varphi} \qquad (2-31b)$$

Z výše uvedené linearizovaného vztahu lze sestavit celkové blokové schéma linearizace v azimutu (linearizovaná dynamika v azimutu a lineární senzorický subsystém), tedy dle [1]:



Obr. 2.2.15a: celkové blokové schéma linearizované dynamiky v azimutu při zanedbání reakčního momentu síly hlavního motoru

Z obr. 2.2.15a lze rovněž sestavit obyčejnou lineární diferenciální rovnici druhého řádu pro linearizovanou dynamiku v azimutu (při zanedbání reakčního momentu síly hlavního motoru), tedy:

$$\ddot{\varphi} = \frac{1}{J_{\varphi}} \cdot \left[M_{2(emp.)} - B_{\varphi_1} \cdot \dot{\varphi} \right] \qquad (2-32a)$$

po ekvivalentních úpravách

$$J_{\varphi} \cdot \ddot{\varphi} + B_{\varphi_1} \cdot \dot{\varphi} = M_{2(emp.)} = f(U_T) \qquad (2-32b)$$

z toho přenosová funkce (po Laplaceově transformaci a při nulových počátečních podmínkách)

$$\hat{G}_{azim.}(s) = \frac{\hat{\Phi}(s)}{\hat{M}_{2(emp.)}(s)} = \frac{1}{J_{\varphi} \cdot s^2 + B_{\varphi_1} \cdot s} \qquad (2-32c)$$

Statický senzorický subsystém v azimutu je popsán lineární rovnici – viz relace (2 – 12c):

$$y_\varphi = k_\varphi \cdot \varphi + y_{\varphi_0} \qquad (2-32d)$$

z toho Laplaceův obraz výstupního úhlu azimutu (při nulových počátečních podmínkách)

$$\hat{Y}_\varphi(s) = k_\varphi \cdot \hat{\Phi}(s) + \frac{1}{s} \cdot y_{\varphi_0} = k_\varphi \cdot \hat{G}_{azim.}(s) \cdot \hat{M}_{2(emp.)}(s) + \frac{1}{s} \cdot y_{\varphi_0} \qquad (2-32e)$$

po dosazení z relace (2 – 32c)

$$\hat{Y}_\varphi(s) = k_\varphi \cdot \frac{\hat{M}_{2(emp.)}(s)}{J_\varphi \cdot s^2 + B_{\varphi_1} \cdot s} + \frac{1}{s} \cdot y_{\varphi_0} \qquad (2-32f)$$

kde $\hat{M}_{2\,(emp.)}(s)$ Laplaceův obraz momentu síly vrtule azim. motoru v emp. modelu

$\hat{G}_{azim.}(s)$ přenosová funkce linearizované dynamiky v azimutu

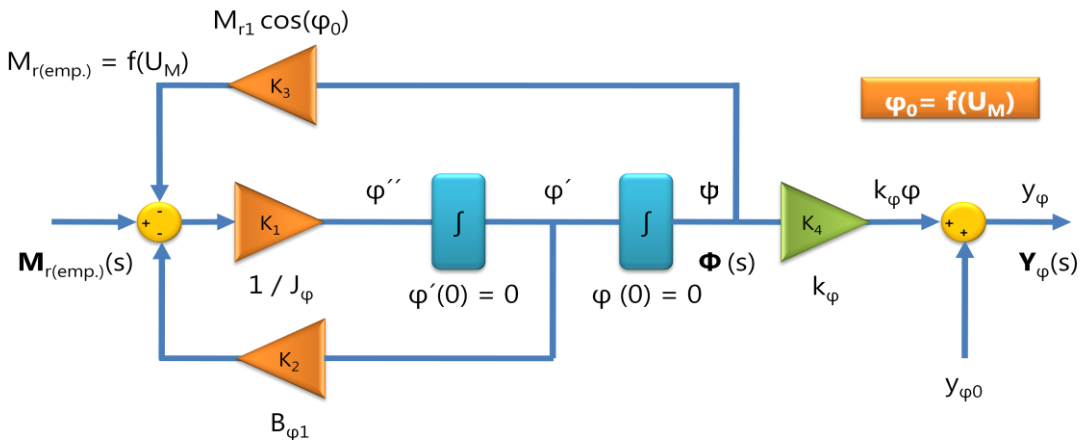
$\hat{Y}_\varphi(s)$ Laplaceův obraz výstupní hodnoty azimutového úhlu

Výše uvedené relace pro azimut platí za předpokladu zanedbání vlivu reakčního momentu síly hlavního motoru, což platí např. při fixním upevnění elevace nebo nulovém napětí hlavního motoru. Pokud však bychom uvažovali vliv výstupního reakčního napětí hlavního motoru $U_{1\,(reak.)}$, respektive vliv napětí U_M, a z něj následný reakční moment síly v empirickém modelu $M_{r(emp.)}$, odezvu na reakční moment M_{r_1} a úhel azimutu v ustáleném stavu φ_0, pak lze výstupní rovnici (2 – 32f) formálně přepsat do tvaru, tedy:

$$\hat{Y}_\varphi(s) = k_\varphi \cdot \frac{\hat{M}_{r(emp.)}(s)}{J_\varphi \cdot s^2 + B_{\varphi_1} \cdot s + M_{r_1} \cdot cos(\varphi_0)} + \frac{1}{s} \cdot y_{\varphi_0} \qquad (2-32g)$$

$\hat{M}_{r\,(emp.)}(s)$ Laplaceův obraz reakčního momentu síly v empirickém modelu

φ_0 úhel azimutu v ustáleném stavu



Obr. 2.2.15b: celkové blokové schéma linearizované dynamiky v azimutu při uvažování reakčního momentu síly hlavního motoru

Výše uvedená relace (2 – 32g) platí při volném azimutu a nulovém napětí ocasního motoru. Popisuje tedy situaci, kdy zkoumáme vliv elevace na azimut ve formě azimutového úhlu v ustáleném stavu. Po změření úhlů azimutu v ustáleném stavu při různých úrovních napětí U_M lze vliv odezvy, tedy momentu M_{r_1}, zanedbat.

2.2.5 Ověření namodelovaného systému

Poslední část modelování systému spočívá v jeho sestavení matematického a ověření validity, tj. stanovení platnosti, do jaké míry se shoduje s reálným systémem, popsaným v podkapitole 2.2.2.

K účelu porovnání obou modelů byly v prostředí MATLAB&Simulink vytvořeny následující soubory, tedy:

- konfigurační M-soubor obsahující hodnoty všech důležitých parametrů nejen reálného modelu, ale také namodelovaného linearizovaného modelu
- model v Simulinku obsahující schéma reálného modelu a schéma sestaveného linearizovaného modelu

Matematický model v elevaci

Přenosová funkce linearizovaného modelu v elevaci má při $U_B = -0{,}85\ [MU]$ následující tvar, tedy:

$$\hat{G}_\psi(s) = \frac{\Psi(s)}{U_M(s)} = 13 \cdot \frac{k_\psi \cdot [a_1 \cdot U_M \cdot s + b_1]}{(T_1 \cdot s + 1)^2 \cdot [J_\psi \cdot s^2 + B_{\psi_1} \cdot s + k_B \cdot U_B \cdot cos(\psi_0)]} \qquad (2-33a)$$

po dosazení

$$\hat{G}_\psi(s) = \frac{\Psi(s)}{U_M(s)} = \frac{13}{1024} \cdot \frac{0{,}105 \cdot 0{,}55 \cdot s + 0{,}00936}{(0{,}25 \cdot s + 1)^2 \cdot \left\{0{,}00437 \cdot s^2 + 0{,}00184 \cdot s + (-0{,}121) \cdot (-0{,}85) \cdot cos\left(\frac{77}{1024} \cdot \pi\right)\right\}} \qquad (2-33b)$$

při

$$k_\psi = \frac{1}{1024} \qquad (2-33c)$$

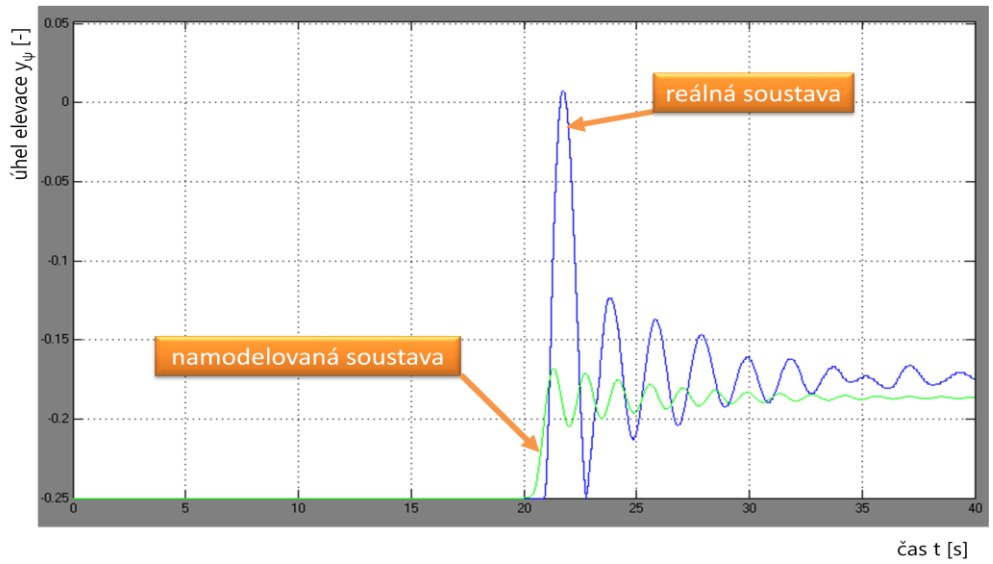
Z obr. 2.2.14 je zřejmé, že platí relace $\psi_0 = f(U_M)$ čili velikost vstupního napětí hlavního (elevačního) motoru v [MU], respektive ve [V], udává velikost úhlu elevace v ustáleném stavu v [rad], přičemž velikost vstupního napětí hlavního motoru byla stanovena experimentálně tak, aby se velikost elevačního úhlu ustálila na konkrétní hodnotě, tedy:

$$\psi_0 = f\{U_M = 0{,}55\ [MU] = 6{,}60\ [V]\} = p_\psi \cdot k_\psi = \frac{77 \cdot \pi}{1024}\,[rad] \qquad (2-33d)$$

kde p_ψ počet kroků elev. enkodéru v ustáleném stavu při detekci X4 $[MP]$

V případě relace (2 − 33c) se hodnota konstanty k_ψ zvolila správně, byť se v relaci (2 − 14f) liší o přenásobení Ludolfovým číslem π. Pokud bychom využili (2 − 14f), museli bychom také (2 − 33b) a ofset elevačního IRC čidla přenásobit π.

Naopak v případě relace (2 − 33d) musí být velikost elevačního úhlu v ustáleném stavu ψ_0 v [rad]. V tomto bodě je soustava v elevaci linearizována.



Obr. 2.2.16: odezvy namodelované soustavy v elevaci (žlutý průběh) a reálné soustavy v elevaci (modrý průběh) na velikost budícího napětí hlavního motoru $U_M = 0{,}55\ [MU] = 6{,}6\ [V]$, při $U_B = -0{,}85\ [V]$, $U_T = 0{,}0\ [MU] = 0{,}0\ [V]$ a X4 detekci kvadraturních signálů

Odezvy na obr. 2.2.16 charakterizují celkové chování v elevaci pro reálný a namodelovaný systém. Schéma časování experimentu je následující, tedy:

- **(0 až 5) [s]** inicializace reálného modelu – protočení obou vrtulí a kalibrace IRC čidel
- **(5 až 20) [s]** technická pauza – příprava reálného modelu
- **(20 až 40) [s]** skoková změna budicího napětí na $U_M = 0{,}55\ [MU] = 6{,}60\ [V]$

Ustálený stav odezvy reálného modelu je dán relací, tedy:

v radiánech

$$y_\psi = p_\psi \cdot k_\psi + y_{\psi_0} = \psi_0 + y_{\psi_0} = \frac{77 \cdot \pi}{1024} - \frac{\pi}{4} = \frac{77-256}{1024} \cdot \pi \cong -0{,}1748 \cdot \pi\ [rad] \qquad (2-33e)$$

v bezrozměrných jednotkách

$$y_\psi = p_\psi \cdot k_\psi + y_{\psi_0} = \psi_0 + y_{\psi_0} = \frac{77}{1024} - \frac{1}{4} = \frac{77-256}{1024} \cong -0{,}1748\ [-] \qquad (2-33f)$$

Výsledek relace (2 – 33f) přirozeně koresponduje s hodnotou ustáleného stavu odezvy reálné soustavy na obr. 2.2.16.

Matematický model v azimutu

Přenosová funkce linearizovaného modelu v azimutu má při $U_B = -0{,}85\ [MU]$ a $U_M = 0{,}0\ [MU]$ následující tvar podle relace (2 – 32c), tedy:

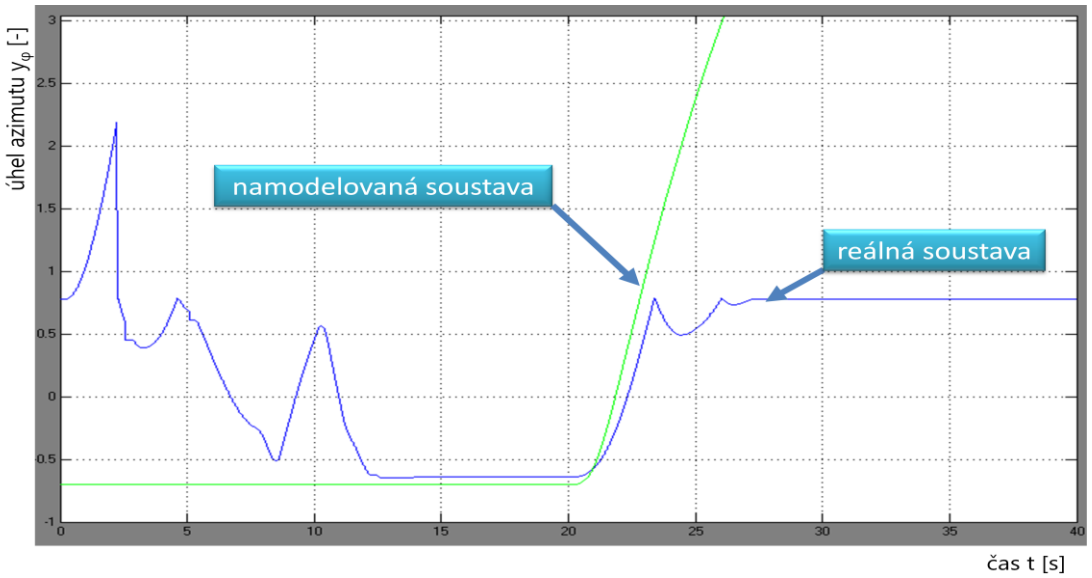
$$\hat{G}_\varphi(s) = \frac{\Phi(s)}{U_T(s)} = 78 \cdot \frac{k_\varphi \cdot [a_2 \cdot U_T \cdot s + b_2]}{(T_2 \cdot s + 1)^2 \cdot [J_\varphi \cdot s^2 + B_{\varphi 1} \cdot s]} \qquad (2-34a)$$

po dosazení

$$\hat{G}_\varphi(s) = \frac{\Phi(s)}{U_T(s)} = \frac{78}{1024} \cdot \frac{0{,}033 \cdot 0{,}20 \cdot s + 0{,}0294}{(0{,}25 \cdot s + 1)^2 \cdot \{0{,}00414 \cdot s^2 + 0{,}00869 \cdot s\}} \qquad (2-34b)$$

při

$$k_{\varphi} = \frac{1}{1024} \qquad (2-34c)$$



Obr. 2.2.17: odezvy namodelované soustavy v azimutu (žlutý průběh) a reálné soustavy v azimutu (modrý průběh) na velikost budícího napětí ocasního motoru $U_T = 0{,}20\ [MU] = 1{,}2\ [V]$, při $U_B = -0{,}85\ [V]$, $U_M = 0{,}0\ [MU] = 0{,}0\ [V]$ a X4 detekci kvadraturních signálů

Z obr. 2.2.15a je zřejmé, že dynamika v azimutu zanedbává reakční moment vazby elevace-azimut. S tímto předpokladem také pracuje matematický model v azimutu, protože koncepce regulátorů počítá s autonomním řízením elevace a azimutu.

Odezvy na obr. 2.2.17 charakterizují celkové chování v azimutu pro reálný a namodelovaný systém. Schéma časování experimentu je následující, tedy:

- **(0 až 5) [s]** inicializace reálného modelu – protočení obou vrtulí a kalibrace IRC čidel
- **(5 až 20) [s]** technická pauza – příprava reálného modelu
- **(20 až 40) [s]** skoková změna budicího napětí na $U_T = 0{,}20\ [MU] = 1{,}20\ [V]$

Z obr. 2.2.15a rovněž plyne, že ustálený stav azimutového úhlu je dán mechanickým dorazem, neboť při zvolené hodnotě budicího napětí $U_T = 0{,}20\ [MU] = 1{,}20\ [V]$ nenastalo přirozené ustálení; při $U_T \in (0{,}10\ [MU];\ 0{,}20\ [MU])$ se vrtule azimutu buď téměř netočí, nebo stejně nedojde k ustálení.

Kdyby nebyl úhel u reálné soustavy omezen mechanickým dorazem, rostl by nade všechny meze jako charakteristika namodelované soustavy.

3 Robustní řízení

3.1 Úvod do problematiky

3.1.1 Robustnost a výkonnost

Cílem robustního řízení je navrhnout dynamický řídicí systém pracující v reálném prostředí. Změny okolních podmínek mohou být kupříkladu způsobeny těmito faktory dle [7]:

- stárnutí komponent
- vliv teploty
- vliv pracovního prostředí

Řídicí systém musí být nejen odolný vůči výše uvedeným faktorům, ale také musí eliminovat nepřesnost modelu, tedy příslušná schopnost řídicího systému akceptování změn se nazývá robustnost. Žádaná hodnota výstupu bude dosažena i při omezených změnách vlastností řízené soustavy a při působení konstantních poruchových signálů. Z matematického hlediska to znamená, že robustní regulátor není vhodný pouze pro jednu konkrétní soustavu, ale pro množinu soustav. [7]

Jinými slovy robustnost hraje při návrhu řídicích systémů výraznou roli, protože reálné systémy jsou náchylné k vnějším poruchám a šumům měření. Navíc velice často existují rozdíly mezi navrženými matematickými modely a konkrétními reálnými systémy. Typickým příkladem může být navržení regulátoru, který bude stabilizovat soustavu, pokud není původně stabilní, a akceptovat určitou míru výkonnosti za přítomnosti poruchových signálů, šumů, těžko namodelovatelných procesních dynamických charakteristik nebo proměnných parametrů procesu. Takovéto typy úloh se nejlépe řeší mechanismem zpětnovazebního řízení, byť s sebou přinášejí řadu problémů dle [8]:

- vysoká cena (např. použití snímačů)
- složitost systému (např. možnosti implementace a spolehlivost)
- stabilita systému (např. požadavek interní stability a stabilizujících regulátorů)

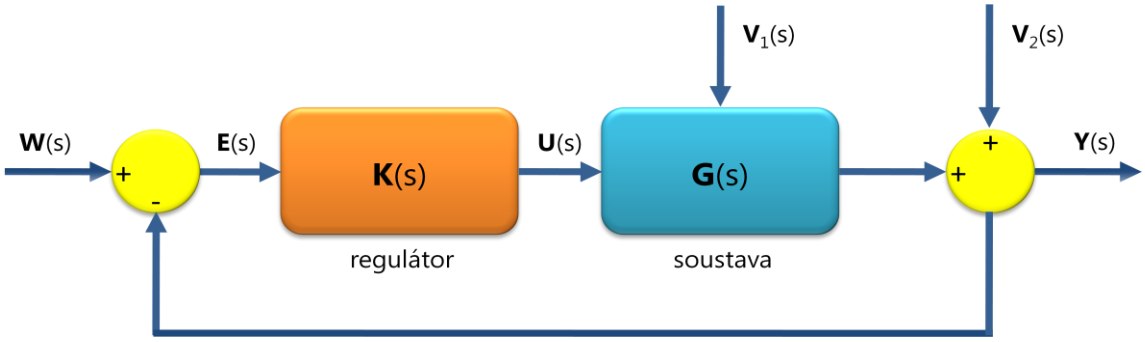
Potřeba a význam robustnosti jako součásti návrhu řídicích systémů se rozvíjí od 80. let 20. století. V klasickém SISO řízení je robustnosti zajištěno vhodným zesílením (amplitudová bezpečnost) a fází (fázová bezpečnost). Když se v 60. letech 20. století rozvinuly první návrhové techniky s více proměnnými, kladl se důraz na dosažení dobré výkonnosti, nikoliv však robustnosti. Tyto techniky s více proměnnými byly založeny na lineárně-kvadratickém kritériu a gaussovských poruchách. Ukázalo se, že mohou být úspěšně využity v řadě leteckých aplikací, kde lze vytvořit přesné matematické modely, včetně popisů vnějších poruchových signálů či šumů. Nicméně aplikace těchto metod – souhrnně nazývané jako LQG metody (lineárně-kvadratické gaussovské řízení) – v dalších průmyslových oborech zcela zjevně prokázaly jejich špatné vlastnosti z hlediska robustnosti, což vedlo ke snaze vyvinout teorii, která by explicitně zabývala otázkou robustnosti ve zpětnovazebním návrhu řízení. Průkopnická práce ve vývoji chystané teorie, dnes známé jako teorie optimálního řízení v H∞, byla představena na počátku 80. let 20. století Georgem Zamesem a Brucem A. Francisem. V H∞ přístupu se zpočátku specifikuje model systémové neurčitosti, tj. aditivní perturbace a/nebo výstupní poruchy. Ve většině případů stačí najít vhodný regulátor tak, aby uzavřená smyčka dosáhla určité robustní stability. Výkonnost je rovněž součástí optimalizační ztrátové (účelové) funkce. Elegantní formulace řešení jsou založeny na řešeních Riccatiho rovnic např. v MATLABu. [8]

Navrhneme-li regulátor pro konkrétní interval hodnot nějaké veličiny, pak je regulační obvod robustně stabilní. Dalším významným parametrem robustních regulátorů je jejich výkonnost, splňující požadavky na parametry dle [7]:

- řízení
- porucha
- rychlost odezvy

Problém návrhu robustního regulátoru je založen na zpětnovazebním obvodu (uzavřená smyčka), což umožňuje pracovat s citlivostí a eliminací poruchy. Zpětná vazba nestabilní soustavy na jedné straně stabilizuje, na druhé straně však může stabilní soustavu destabilizovat. [7]

Uvažujme klasické regulační schéma dle [7]:



Obr. 3.1.1: klasické regulační schéma s definicí obvodových veličin

Popis veličin v regulačním obvodě je následující:

$\hat{W}(s)$ Laplaceův obraz řídicí veličiny
$\hat{E}(s)$ Laplaceův obraz regulační odchylky
$\hat{U}(s)$ Laplaceův obraz akční veličiny
$\hat{V}_1(s)$ Laplaceův obraz poruchy – NF známé a neznámé poruchy; systém je musí eliminovat
$\hat{V}_2(s)$ Laplaceův obraz poruchy – senzory nebo měření, VF charakter a nepatrný vliv

Pomocí pravidel blokové algebry definujme matematické relace regulačního obvodu, tedy dle [7] [10]:

pro přenos otevřené smyčky

$$\hat{L}(s) = \hat{K}(s) \cdot \hat{G}(s) \qquad (3-01a)$$

pro přenos regulační odchylky neboli citlivostní funkce

$$\hat{G}_E(s) = \frac{\hat{E}(s)}{\hat{W}(s)} = \frac{1}{1+\hat{K}(s)\cdot\hat{G}(s)} = \frac{1}{1+\hat{L}(s)} = \hat{S}(s) \qquad (3-01b)$$

pro přenos řídicí veličiny neboli komplementární citlivostní funkce

$$\hat{G}_W(s) = \frac{\hat{Y}(s)}{\hat{W}(s)} = \frac{\hat{K}(s)\cdot\hat{G}(s)}{1+\hat{K}(s)\cdot\hat{G}(s)} = \frac{\hat{L}(s)}{1+\hat{L}(s)} = \hat{T}(s) \qquad (3-01c)$$

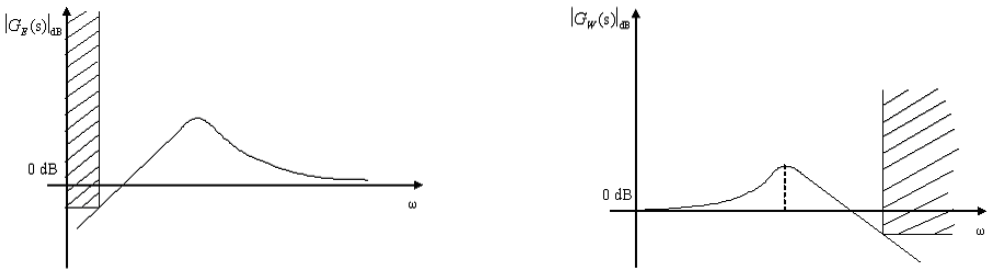
pro součet citlivostní funkce a komplementární citlivostní funkce platí omezující podmínka

$$\hat{S}(s) + \hat{T}(s) = \frac{1}{1+\hat{L}(s)} + \frac{\hat{L}(s)}{1+\hat{L}(s)} = \frac{1+\hat{L}(s)}{1+\hat{L}(s)} = 1 \qquad (3-01d)$$

Uvažujme požadavky na citlivostní funkci a komplementární funkci dle [7]:

- při $v_1(t) = v_2(t) = 0$ převažuje vliv řízení, tudíž citlivostní funkce $\hat{G}_E(s) = \hat{S}(s)$ musí být malá a komplementární citlivostní funkce $\hat{G}_W(s) = \hat{T}(s)$ bude velká
- eliminaci nízkofrekvenčního šumu $v_1(t)$ provádí celý regulační obvod, tudíž opět převažuje vliv řízení neboli citlivostní funkce $\hat{G}_E(s) = \hat{S}(s)$ musí být malá a komplementární citlivostní funkce $\hat{G}_W(s) = \hat{T}(s)$ bude velká
- eliminaci vysokofrekvenčního šumu $v_2(t)$ nesmí neovlivňovat celý regulační obvod, tudíž eliminujeme vlivy řízení a odchylky neboli citlivostní funkce $\hat{G}_E(s) = \hat{S}(s)$ musí být malá a komplementární citlivostní funkce $\hat{G}_W(s) = \hat{T}(s)$ bude rovněž malá

Jelikož nelze uvedené protichůdné požadavky splnit jedním regulátorem, je nutno nalézt kompromis mezi velikostmi citlivostní funkce a komplementární citlivostní funkce. [7]



Obr. 3.1.2: logaritmické amplitudové frekvenční charakteristiky pro citlivostní funkci (vlevo) a komplementární citlivostní funkci (vpravo)

3.1.2 Norma

Řídicí systém je s okolním prostředím provázán řídicími signály, poruchovými signály a šumovými signály, přičemž signály sledování odchylky a signály akční veličiny jsou rovněž pro návrh důležité. Pro účel analýzy a návrhu musí být normy definovány jako velikost takových signálů.

Definice matematické normy

Norma je funkce, která každému nenulovému vektoru přiřazuje kladné reálné číslo (tzv. délku nebo velikost), přičemž nulový vektor jako jediný má nulovou délku. V případě seminormy se naopak připouští, aby i nenulovým vektorům byla přiřazena nulová délka. [9]

Nechť V je vektorový prostor nad nějakým podtělesem $F \in \mathbb{R}$ určitého tělesa komplexních čísel (např. Riemannova koule neboli otevřená Gaussova rovina) a $\|.\|$ představuje reálnou funkci definovanou na prostoru V. [9]

Každá norma musí mít čtyři vlastnosti, tedy dle [9]:

$$\|u\| \geq 0 \qquad (3-02a)$$

$$\|u\| = 0 \Leftrightarrow u(t) = 0 \;\; pro \; \forall t \in \mathbb{R} \qquad (3-02b)$$

$$\|a \cdot u\| = |a| \cdot \|u\| \;\; pro \; \forall a \in F \qquad (3-02c)$$

$$\|u + v\| \leq \|u\| + \|v\| \;\; pro \; u, v \; \in V \qquad (3-02d)$$

Relace (3 – 01b) definuje tzv. seminormu.

Typy matematických norem

Euklidovská norma

Nechť na $\mathbb{R}^n$ existuje vektor $\boldsymbol{x} = \{x_1, x_2, \dots, x_n\}$, pak euklidovská norma tohoto vektoru je dána relací, tedy:

$$\|\boldsymbol{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} = \sqrt{\sum_{i=1}^{n} x_i^2} \qquad (3-03a)$$

Důsledkem této normy je Pythagorova věta. [9]

p-norma

Nechť na $\mathbb{R}^n$ existuje vektor $\boldsymbol{x} = \{x_1, x_2, \dots, x_n\}$ a nechť $p \geq 1$, kdy $p \in \mathbb{R}$, pak p-norma tohoto vektoru je dána relací, tedy dle [9]:

$$\|\boldsymbol{x}\|_p = [\sum_{i=1}^{n} |x_i|^p]^{\frac{1}{p}} \qquad (3-03b)$$

speciálně pro $p = 2$ *dostáváme euklidovskou normu*

$$\|\boldsymbol{x}\|_2 = [\sum_{i=1}^{n} |x_i|^2]^{\frac{1}{2}} = \sqrt{\sum_{i=1}^{n} |x_i|^2} = \sqrt{\sum_{i=1}^{n} x_i^2} \qquad (3-03c)$$

Maximová norma

Nechť na $\mathbb{R}^n$ existuje vektor $\boldsymbol{x} = \{x_1, x_2, \dots, x_n\}$, pak pro normu maxima platí relace, tedy dle [9]:

$$\|\boldsymbol{x}\|_\infty = max\{|x_1|, |x_2|, \dots, |x_n|\} = sup\{\boldsymbol{x}\} \qquad (3-03d)$$

kde $sup\{\boldsymbol{x}\}$ supremum množiny (vektoru) $\boldsymbol{x}$

Supremum v množině reálných čísel má každá shora omezená množina, přestože ne každá taková množina má maximum neboli největší prvek množiny. Jinými slovy supremum se zavádí k pojmu maximum. Rozdíl mezi maximem a supremem ovšem spočívá v tom, že supremum lze dohledat u více množin – např. omezené otevřené intervaly reálných čísel nemají maximum, ale lze u nich určit právě supremum coby nejmenší prvek (minimum) všech horních závor množiny. Pokud má množina maximum, má také supremum, které je zároveň maximem množiny. Dodejme, že naopak tato implikace platit nemusí. [9]

Pokud supremum existuje, je jednoznačně určeno, tj. daná množina nemůže mít dvě různá suprema, neboť v množině horních závor lze jednoznačně určit nejmenší prvek. [9]

Shora neomezené množiny reálných čísel supremum pochopitelně nemají.

Normy signálů a přenosových funkcí

Výše uvedené relace (3 − 03b), (3 − 03c) a (3 − 03d) nyní uplatníme pro vyjádření norem signálů spojitých v čase (záměna symbolu numerické sumace za spojitou integraci na definičním oboru spojitého času) a kmitočtových přenosových funkcí (za předpokladu integrace na definičním oboru kmitočtu).

Normy signálů

Nechť $t \in (-\infty; +\infty)$ přestavuje spojitý čas, nechť $\forall\, t \in (-\infty; +\infty)$: $f(t) = u(t)$ a nechť $p \geq 1$, kdy $p \in \mathbb{R}$, pak obecná p-norma takovéhoto skalárního spojitého nebo po částech spojitého signálu je dána relací, tedy dle [9]:

$$L^p_{m=1}(\mathbb{R}) \coloneqq \left\{ u(t) : \|\boldsymbol{u}\|_p = \left[\int_{t=-\infty}^{t=+\infty} |u(t)|^p \cdot dt \right]^{\frac{1}{p}} \right\} \quad (3-03e)$$

speciálně pro $p = 1$ *dostáváme*

$$\|\boldsymbol{u}\|_1 = \left[\int_{t=-\infty}^{t=+\infty} |u(t)|^1 \cdot dt \right]^{\frac{1}{1}} = \int_{t=-\infty}^{t=+\infty} |u(t)| \cdot dt \quad (3-03f)$$

pro $p = 2$ *platí (euklidovská norma)*

$$\|\boldsymbol{u}\|_2 = \left[\int_{t=-\infty}^{t=+\infty} |u(t)|^2 \cdot dt \right]^{\frac{1}{2}} = \sqrt{\int_{t=-\infty}^{t=+\infty} |u(t)|^2 \cdot dt} = \sqrt{\int_{t=-\infty}^{t=+\infty} u^2(t) \cdot dt}; \quad (3-03g)$$

pro $p = \infty$ *platí (maximová norma)*

$$\|\boldsymbol{u}\|_\infty = \left[\int_{t=-\infty}^{t=+\infty} |u(t)|^\infty \cdot dt \right]^{\frac{1}{\infty}} = sup_{t \in \mathbb{R}} \{|u(t)|\} \quad (3-03h)$$

Normované prostory, obsahující signály s konečnou normou, se označují jako $L^1(\mathbb{R})$, $L^p(\mathbb{R})$ a $L^\infty(\mathbb{R})$. Z hlediska signálu lze konstatovat, že relace (3 − 03f) představuje nevlastní integrál absolutní hodnoty signálu $u(t)$. Relace (3 − 03g) se často označuje jako energie signálu $u(t)$. Relace (3 − 03h) zase představuje amplitudu či špičkovou hodnotu signálu $u(t)$. [10]

Nechť $t \in (-\infty; +\infty)$ přestavuje spojitý čas, nechť $\forall\, t \in (-\infty; +\infty)$: $f(t) = [u_1(t), u_2(t), \dots, u_m(t)]^T$ a nechť $p \geq 1$, kdy $p \in \mathbb{R}$, pak obecná p-norma takovéhoto vektorového spojitého nebo po částech spojitého signálu je dána relací, tedy dle [9]:

$$L^p_m(\mathbb{R}) \coloneqq \left\{ u(t) : \|\boldsymbol{u}\|_p = \left[\int_{t=-\infty}^{t=+\infty} \sum_{i=1}^{m} |u(t)|^p \cdot dt \right]^{\frac{1}{p}} < \infty \right\} \quad (3-03ch)$$

kde $m \in \mathbb{Z}$ rozměr vektorového prostoru

Pro $p = \infty$ lze relaci (3 − 02ch) formálně upravit, tedy:

$$L^\infty_m(\mathbb{R}) \coloneqq \{ u(t) : \|\boldsymbol{u}\|_\infty = sup_{t \in \mathbb{R}} \{|u(t)|\} < \infty \} \quad (3-03k)$$

Z relace (3 − 03h) plyne, že se jedná o zobecnění relace (3 − 03a), která platí pro jednorozměrný signál.

Normy kmitočtových přenosových funkcí

Nechť $\omega \in (-\infty; +\infty)$ je kmitočet, nechť $\forall\, \omega \in (-\infty; +\infty)$: $\boldsymbol{F}(j \cdot \omega) = \boldsymbol{G}(j \cdot \omega)$ a nechť $p \geq 1$, kdy $p \in \mathbb{R}$, pak obecná p-norma takovéto přenosové funkce je dána relací, tedy dle [9]:

$$\|\boldsymbol{G}\|_p = \left[\frac{1}{2 \cdot \pi} \cdot \int_{\omega=-\infty}^{\omega=+\infty} |\boldsymbol{G}(j \cdot \omega)|^p \cdot d\omega \right]^{\frac{1}{p}} = \left[\frac{1}{2 \cdot \pi} \cdot \int_{\omega=-\infty}^{\omega=+\infty} \left| \hat{G}(j \cdot \omega) \right|^p \cdot d\omega \right]^{\frac{1}{p}} \quad (3-03ch)$$

pro $p = 1$ *platí*

$$\|\boldsymbol{G}\|_1 = \left[\frac{1}{2 \cdot \pi} \cdot \int_{\omega=-\infty}^{\omega=+\infty} |\boldsymbol{G}(j \cdot \omega)|^1 \cdot d\omega \right]^{\frac{1}{1}} = \frac{1}{2 \cdot \pi} \cdot \int_{\omega=-\infty}^{\omega=+\infty} \left| \hat{G}(j \cdot \omega) \right| \cdot d\omega \quad (3-03i)$$

pro $p = 2$ *platí (euklidovská norma)*

$$\|\boldsymbol{G}\|_2 = \left[\frac{1}{2\cdot\pi}\cdot\int_{\omega=-\infty}^{\omega=+\infty}|\boldsymbol{G}(j\cdot\omega)|^2\cdot d\omega\right]^{\frac{1}{2}} = \frac{1}{\sqrt{2\cdot\pi}}\cdot\sqrt{\int_{\omega=-\infty}^{\omega=+\infty}\left|\hat{G}(j\cdot\omega)\right|^2\cdot d\omega} \qquad (3-03j)$$

pro $p \to \infty$ platí:

$$\|\boldsymbol{G}\|_\infty = \left[\frac{1}{2\cdot\pi}\cdot\int_{\omega=-\infty}^{\omega=+\infty}|\boldsymbol{G}(j\cdot\omega)|^\infty\cdot d\omega\right]^{\frac{1}{\infty}} = sup_\omega\{\left|\hat{G}(j\cdot\omega)\right|\} \qquad (3-03k)$$

Aplikace norem v řízení

Nejčastějšími normami, využívanými v robustním řízení, jsou následující dle [10]:

- **norma H_2 (konečná)** – *přenosová funkce je striktně ryzí (stupeň čitatele přenosu je menší než stupeň jmenovatele přenosu) bez pólu na imaginární ose; metoda využívá euklidovskou normu. Matematicky H_2 označuje Hilbertův prostor všech regulárních (analytických) komplexních funkcí.*
- **norma H∞ (konečná)** – *přenosová funkce je ryzí (stupeň čitatele přenosu je menší nebo roven stupni jmenovatele přenosu) bez pólu na imaginární ose; metoda využívá maximovou normu. Matematicky H_∞ označuje Banachův prostor všech regulárních (analytických) komplexních funkcí. V tomto úplně normovaném prostoru každá cauchyovská posloupnost konverguje.*

Z matematického hlediska je možné počítat normy i nestabilních přenosů (póly v pravé polorovině Gaussovy roviny), ovšem z hlediska řízení to nemá význam. [7]

3.1.3 Neurčitosti systému

Neurčitost může být obecně rozdělena do dvou kategorií, tedy dle [8]:

- poruchové signály
- dynamické perturbace

Poruchové signály se projevují jako vstupní a výstupní poruchové veličiny, šum senzorů nebo šum aktuátoru apod. Dynamické perturbace zase představují míru nesrovnalosti mezi matematickým modelem systému (např. teoretický nebo empirický model vrtulníku) a dynamikou reálného systému (např. reálný model vrtulníku). Matematický model jakéhokoliv systému je vždy pouhou aproximací fyzikální reality dynamiky systému. Typické zdroje takovýchto nesrovnalostí reprezentují nenamodelovatelné (obvykle vysokofrekvenční) dynamiky, zanedbání nelinearit systému při jeho modelování, provedení úmyslného snižování řádu systému a změny parametrů systému způsobené změnami okolí modelu. Tyto chyby v modelování mohou nepříznivě ovlivnit stabilitu a chování řízeného systému. [8]

Nestrukturované neurčitosti

Mnoho dynamických perturbací, které mohou nastat v různých částech systému, lze soustředit do jediného perturbačního bloku Δ (tzv. deltablok), např. některé těžko modelovatelné vysokofrekvenční dynamiky. Této reprezentaci neurčitosti říkáme nestrukturovaná neurčitost. [8]

V případě lineárního, time-invariantního systému představuje deltablok matici neznámých přenosových funkcí. Nestrukturovanou neurčitost dynamiky, tj. relace mezi nominální soustavou $\hat{G}_0(s)$ a perturbovanou soustavou $\hat{G}_p(s)$, lze popsat několika způsoby, tedy dle [8]:

aditivní perturbace

$$\hat{G}_p(s) = \hat{G}_0(s) + \Delta(s) \qquad (3-04a)$$

inverzní aditivní perturbace

$$\left[\hat{G}_p(s)\right]^{-1} = \left[\hat{G}_0(s)\right]^{-1} + \Delta(s) \qquad (3-04b)$$

vstupní multiplikativní perturbace

$$\hat{G}_p(s) = \hat{G}_0(s) \cdot \left[\hat{I} + \Delta(s)\right] \qquad (3-04c)$$

výstupní multiplikativní perturbace

$$\hat{G}_p(s) = \left[\hat{I} + \Delta(s)\right] \cdot \hat{G}_0(s) \qquad (3-04d)$$

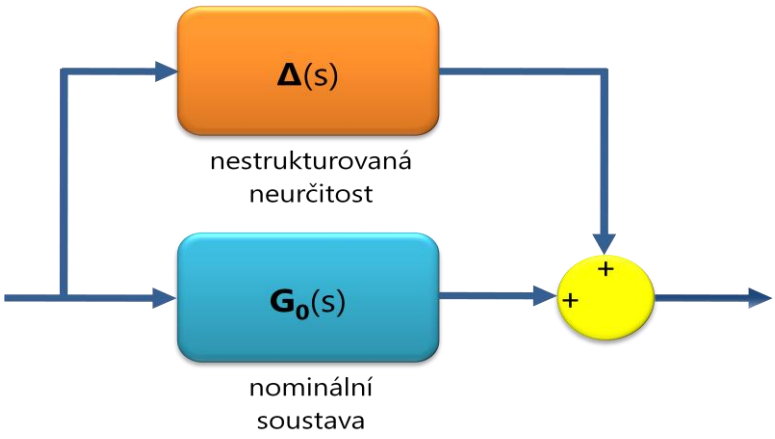
inverzní vstupní multiplikativní perturbace

$$\left[\hat{G}_p(s)\right]^{-1} = \left[\hat{I} + \Delta(s)\right] \cdot \left[\hat{G}_0(s)\right]^{-1} \qquad (3-04e)$$

inverzní výstupní multiplikativní perturbace

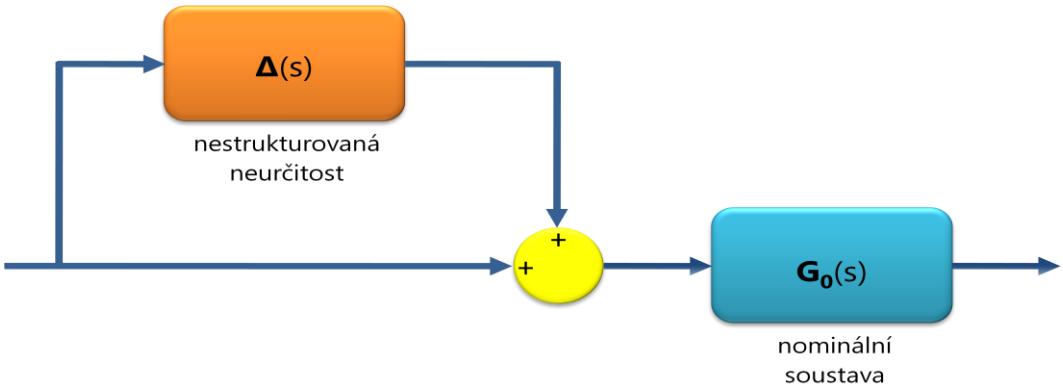
$$\left[\hat{G}_p(s)\right]^{-1} = \left[\hat{G}_0(s)\right]^{-1} \cdot \left[\hat{I} + \Delta(s)\right] \qquad (3-04f)$$

Aditivní vyjádření neurčitosti popisují absolutní chybu mezi aktuální dynamikou a dynamikou nominálního modelu. [8]



Obr. 3.1.3a: základní blokové schéma aditivní perturbace

Multiplikativní vyjádření zase popisují relativní chybu. [8]



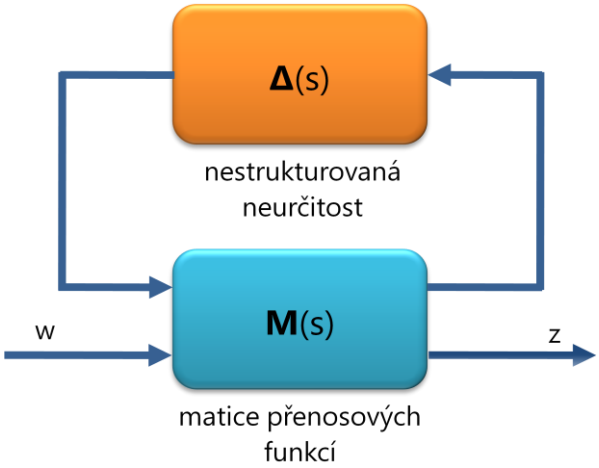
Obr. 3.1.3b: základní blokové schéma multiplikativní perturbace (konfigurace vstupní multiplikativní perturbace)

Parametrické neurčitosti

Jak již bylo zmíněno, nestrukturované neurčitosti jsou vhodné pro popis těžko namodelovatelných nebo zanedbaných systémových dynamik. Tyto komplexní neurčitosti obvykle nastávají v rozsahu vysokých frekvencí a mohou obsahovat dopravní zpoždění, parazitní vazby, hysterezi nebo jiné nelinearity. Nicméně dynamické perturbace v mnoha průmyslových řídicích systémech mohou být rovněž způsobeny nepřesným popisem vlastností komponenty nebo posunem pracovních bodů apod. Takové perturbace jsou představovány výkyvy určitých systémových parametrů přes rozsah možných hodnot (komplexní nebo reálné). Ovlivňují systém v rozsahu nízkých frekvencí. Obvykle jsou tyto neurčitosti reálné. [8] [10]

Lineární fraktální transformace (LFT)

Tento typ reprezentace neurčitosti se poprvé objevil v 50. letech 20. století; později byl adaptován na problematiku robustního řízení, konkrétně pak při modelování neurčitostí. [8]



Obr. 3.1.4: standardní blokové schéma při M-Δ konfiguraci

Matici přenosových funkcí $\widehat{\boldsymbol{M}}(s)$ lze rozepsat dle relace podle [8]:

$$\widehat{\boldsymbol{M}}(s) = \begin{bmatrix} \widehat{M}_{11}(s) & \widehat{M}_{12}(s) \\ \widehat{M}_{21}(s) & \widehat{M}_{22}(s) \end{bmatrix} \qquad (3-05a)$$

Rozměry prvku $\widehat{M}_{11}(s)$ souhlasí s $\Delta(s)$. Pro výstupní veličinu z v interakci se vstupní veličinou w platí následující relaci dle [8]:

$$z = \left\{\widehat{M}_{11}(s) + \widehat{M}_{21}(s)\cdot\Delta(s)\cdot\left[\hat{I} - \widehat{M}_{11}(s)\cdot\Delta(s)\right]^{-1}\cdot\widehat{M}_{12}(s)\right\}\cdot w \qquad (3-05b)$$

Za předpokladu, že k matici $\left[\hat{I} - \widehat{M}_{11}(s)\cdot\Delta(s)\right]$ existuje matice inverzní, čili musí platit relace o nenulovém determinantu této matice – matice je regulární:

$$det\left[\hat{I} - \widehat{M}_{11}(s)\cdot\Delta(s)\right] = \left|\hat{I} - \widehat{M}_{11}(s)\cdot\Delta(s)\right| > 0 \qquad (3-05c)$$

Pak lze definovat lineární fraktální transformaci dle [8]:

$$F\left[\widehat{\boldsymbol{M}}(s),\Delta(s)\right] = \widehat{M}_{22}(s) + \widehat{M}_{21}(s)\cdot\Delta(s)\cdot\left[\hat{I} - \widehat{M}_{11}(s)\cdot\Delta(s)\right]^{-1}\cdot\widehat{M}_{12}(s) \qquad (3-05d)$$

Protože se horní smyčka matice $\widehat{\boldsymbol{M}}(s)$ uzavírá přes deltablok, hovoříme o U lineární fraktální transformaci (ULFT) označenou jako $F_u\left[\widehat{\boldsymbol{M}}(s),\Delta(s)\right]$. Pokud se uzavírá spodní smyčka matice přes deltablok, hovoříme o L lineární fraktální transformaci (LLFT) označenou jako $F_l\left[\widehat{\boldsymbol{M}}(s),\Delta(s)\right]$. Zaměníme-li formálně deltablok za přenos robustního regulátoru, pak lze relaci (3 – 05d) vyjádřit dle [8]:

$$F_l\left[\widehat{\boldsymbol{M}}(s),\widehat{K}(s)\right] = \widehat{M}_{11}(s) + \widehat{M}_{12}(s)\cdot\widehat{K}(s)\cdot\left[\hat{I} - \widehat{M}_{22}(s)\cdot\widehat{K}(s)\right]^{-1}\cdot\widehat{M}_{21}(s) \qquad (3-05e)$$

Přenosová matice $\widehat{\boldsymbol{M}}(s)$ bude mít pro každý typ nestrukturované neurčitosti (aditivní, multiplikativní) jiný tvar, tedy dle [8]:

aditivní perturbace

$$\widehat{M}(s) = \begin{bmatrix} 0 & \hat{I} \\ \hat{I} & \hat{G}_0(s) \end{bmatrix} \qquad (3-05f)$$

inverzní aditivní perturbace

$$\widehat{M}(s) = \begin{bmatrix} -\hat{G}_0(s) & \hat{G}_0(s) \\ -\hat{G}_0(s) & \hat{G}_0(s) \end{bmatrix} \qquad (3-05g)$$

vstupní multiplikativní perturbace

$$\widehat{M}(s) = \begin{bmatrix} 0 & \hat{I} \\ \hat{G}_0(s) & \hat{G}_0(s) \end{bmatrix} \qquad (3-05h)$$

výstupní multiplikativní perturbace

$$\widehat{M}(s) = \begin{bmatrix} 0 & \hat{G}_0(s) \\ \hat{I} & \hat{G}_0(s) \end{bmatrix} \qquad (3-05ch)$$

inverzní vstupní multiplikativní perturbace

$$\widehat{M}(s) = \begin{bmatrix} -\hat{I} & \hat{I} \\ -\hat{G}_0(s) & \hat{G}_0(s) \end{bmatrix} \qquad (3-05i)$$

inverzní výstupní multiplikativní perturbace

$$\widehat{M}(s) = \begin{bmatrix} -\hat{I} & \hat{G}_0(s) \\ -\hat{I} & \hat{G}_0(s) \end{bmatrix} \qquad (3-05j)$$

Strukturované neurčitosti

V mnoha návrzích robustního řízení se pro modelování neurčitosti využívá nestrukturovaná neurčitost v kombinaci s neurčitostí parametrickou. Deltablok má následující obecný tvar dle [8]:

$$\Delta = diag\{\delta_1 \cdot \hat{I}_{r_1}, \dots, \delta_M \cdot \hat{I}_{r_M}, \Delta_1, \dots, \Delta_N\} : \delta_i \in \mathbb{R}, \Delta_j \in \mathbb{R}^{m_j \times m_j} \qquad (3-06a)$$

Dimenze deltabloku je dána relací dle [8]:

$$n = \sum_{i=1}^{M} r_i + \sum_{j=1}^{N} m_j \qquad (3-06b)$$

Deltablok obsahuje dva typy bloků neurčitosti – M opakujících se skalárních bloků a N tzv. plných bloků, přičemž skalární bloky náleží do množiny reálných čísel a plné bloky nemusí být nutně čtvercové (v případě čtvercového tvaru se notace značně zjednoduší). Pokud je perturbovaný systém popsán lineární fraktální transformací (LFT) s výše uvedeným diagonálním tvarem deltabloku, viz relace (3 − 06b), má deltablok určitou strukturu, a tudíž hovoříme o strukturované neurčitosti. [8]

3.2 Metody robustního návrhu

Techniky stavového prostoru v časové doméně umožnily vyhnout se nejen problémům s maticemi přenosových funkcí, ale rovněž poskytly nástroje pro analýzu a návrh systémů s více vstupy a výstupy MIMO. V přibližně stejné době, kdy se vyvíjely metody optimálního řízení, se rovněž vyvíjel výzkum zaměřený na rozšíření nástrojů pro klasické řízení MIMO systémů. Robustní návrh spočívá v nalezení takového regulátoru, aby výsledný systém v uzavřené smyčce byl rovněž robustní. Robustnost se stala hlavním hlediskem v oblasti řízení, tudíž brzy následovaly specifikace a metody, tedy dle [7]:

- metoda H∞
- metoda H_2
- metoda LTR (Loop Transfer Recovery)
- μ – syntéza
- metoda QFT (Quantitative Feedback Theory)
- Charitonovovův teorém pro vyšetření robustní stability
- specifikace teorému o malém zesílení (Small-Gain Theorem)
- specifikace strukturovaných singulárních hodnot (Structured Singular Values)

V dalším výkladu se již zaměříme pouze na metodu H∞.

3.2.1 Metoda H∞

Řídicí systém je robustní, zůstane-li stabilní a splňuje-li určitá kritéria chování za přítomnosti možných neurčitostí. Optimalizační metoda H∞, vyvíjená od 80. let 20. století, se ukázala velice efektivní a účinnou návrhovou metodu robustního řízení v oblasti lineárních, time-invariantních řídicích systémů. [7]

Regulátory typu H∞ mají svoji vlastní terminologii, notaci a koncepci. Tato metoda vede na množinu vhodných stabilních přenosových funkcí, fyzikálně realizovatelných. Obdobně jako u LQR a LQG

regulátorů předpokládáme optimalizaci účelové funkce, která porovná různé přenosové funkce, a vybere z dané množiny tu nejvhodnější. Požadavky pro uzavřenou smyčku jsou následující, tedy dle [7]:

- **fyzikální realizovatelnost**:
  Stupeň jmenovatele přenosové funkce musí být větší nebo roven stupni čitatele přenosu.
- **stabilita**:
  Póly přenosu musí ležet v levé polorovině Gaussovy roviny neboli v oblasti konvergence Laplaceovy transformace (za předpokladu totožnosti řídicí přímky a imaginární osy).

Základním předpokladem metody H∞ je znalost přenosové funkce dané soustavy, přičemž porovnáváme v ∞-normě dle relace (3 – 03k) podle [7] [8]:

$$\|\boldsymbol{G}\|_\infty = \left\|\hat{G}\right\|_\infty = sup_\omega\{|\hat{G}(j\cdot\omega)|\} \qquad (3-07a)$$

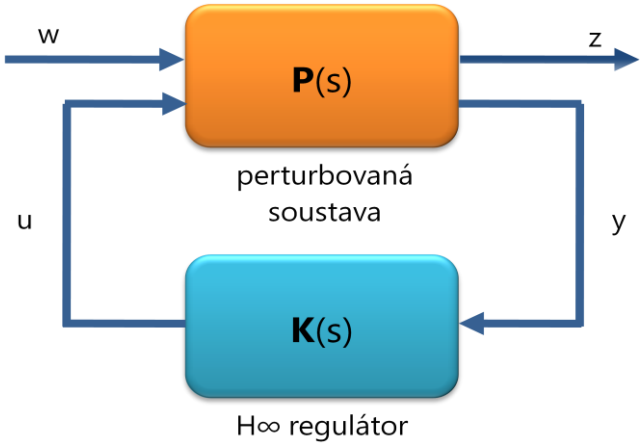
Normu lze graficky reprezentovat jako maximum Bodeho diagramu za předpokladu, že přenosová funkce je konečná a nemá žádné imaginární póly, přičemž cíl spočívá v její minimalizaci v ∞-normě. Snižuje se vrchol Bodeho diagramu, čímž se zvyšuje zásoba robustní stability. [7]

Problém smíšené citlivosti

V praktických průmyslových aplikacích se zpravidla nevyužívá pouze jediná účelová funkce, nýbrž kombinace více takových funkcí, např. splnění požadavku dobrého sledování při omezené energii řídicího signálu. Pak řešíme úlohu smíšené citlivosti neboli tzv. „S over KS“ problém, definovaný obecnou relací (pro SISO systém) dle [8]:

$$\min_{K_{stabilizuj\ ící}} \left\| \begin{matrix} \hat{S}(s) \\ \hat{K}(s)\cdot\hat{S}(s) \end{matrix} \right\|_\infty = \min_{K_{stabilizuj\ ící}} \left\| \begin{matrix} \left[1+\hat{G}(s)\cdot\hat{K}(s)\right]^{-1} \\ \hat{K}(s)\cdot\left[1+\hat{G}(s)\cdot\hat{K}(s)\right]^{-1} \end{matrix} \right\|_\infty \qquad (3-07b)$$

Relace (3 – 07b) coby účelová funkce může rovněž být vyjádřena požadavky návrhu týkající se aditivní perturbace, např. nominální chování, dobré sledování řídicí veličiny (reference) nebo eliminace poruchových veličin a robustní stabilita. [8]



Obr. 3.2.1a: standardní blokové schéma při H∞ konfiguraci

Na obr. 3.2.1 je znázorněno standardní blokové schéma H∞ konfigurace využívající lineární fraktální transformaci (LFT) se specifikací jednotlivých externích vstupů, externích výstupů, vstupů do regulátoru a

výstupů z regulátoru. Regulační obvod obsahuje robustní regulátor s přenosem $\hat{K}(s)$ a perturbovanou (také rozšířenou nebo zobecněnou) soustavu s přenosem $\hat{P}(s)$, která má dva vstupy a dva výstupy dle [7]:

- $w \equiv \boldsymbol{w}(t)$ *vektor vstupní referenční veličiny; externí vstupní signály*
- $u \equiv \boldsymbol{u}(t)$ *vektor výstupní akční veličiny; výstupní řídicí signály z regulátoru*

Hlavní rozdíl mezi těmito vektory spočívá v tom, že regulátor neovlivňuje vstupy. Vektor vstupních řídicích veličin $\boldsymbol{w}(t)$ v sobě zahrnuje externí (vnější) rušení, šum ze senzorů a sledovací (referenční) signály. Naproti tomu výstupy ze soustavy jsou rozděleny do dvou skupin dle [7]:

- $y \equiv \boldsymbol{y}(t)$ *vektor výstupní veličiny; měřené výstupy*
- $z \equiv \boldsymbol{z}(t)$ *regulované výstupy; minimalizované či penalizované výstupy*

Úloha je pak definována tak, že v regulačním obvodě robustního řízení se hledá vnitřně stabilizující regulátor $\hat{K}(s)$ pro danou zobecněnou soustavu $\hat{P}(s)$, který minimalizuje, respektive penalizuje vektor regulovaných výstupů $\boldsymbol{z}(t)$. Jinými slovy minimalizujeme maximovou normu přenosové funkce mezi $\boldsymbol{w}(t)$ a $\boldsymbol{z}(t)$, danou relací dle [8]:

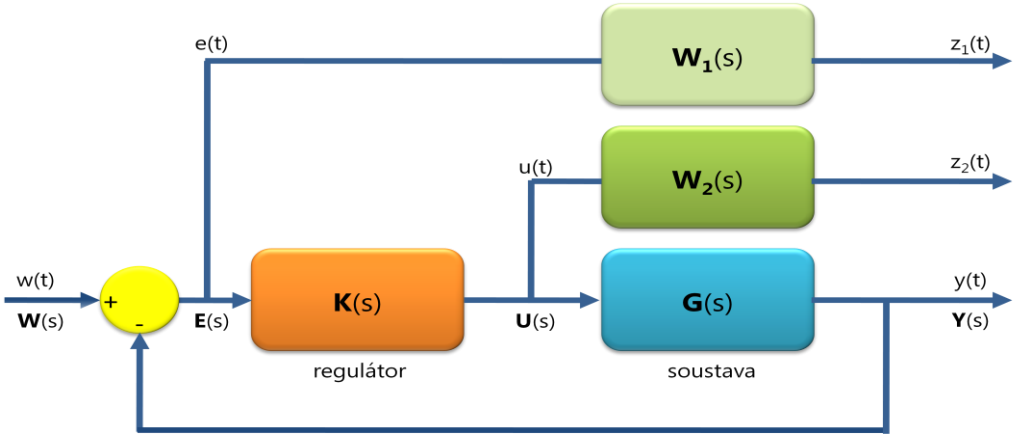
$$z = \left\{\hat{P}_{11}(s) + \hat{P}_{12}(s) \cdot \hat{K}(s) \cdot \left[\hat{I} - \hat{P}_{22}(s) \cdot \hat{K}(s)\right]^{-1} \cdot \hat{P}_{21}(s)\right\} \cdot w \qquad (3-07c)$$

po úpravě dostáváme lineární fraktální transformací

$$z = F_l\left[\hat{P}(s), \hat{K}(s)\right] \cdot w \qquad (3-07d)$$

Pak lze H∞ optimalizační problém vyjádřit vztahem, tedy dle [8]:

$$\min_{K_{stabilizuj\ ící}} \left\| F_l\left[\hat{P}(s), \hat{K}(s)\right] \right\|_\infty \qquad (3-07e)$$



Obr. 3.2.1b: standardní blokové schéma problému smíšené citlivosti – regulátor a perturbovaná soustava obsahující nominální soustavu a váhové filtry regulační odchylky a akční veličiny

Na obr. 3.2.1b je znázorněno standardní blokové schéma problému smíšené citlivosti a jedná se vlastně o podrobnější rozkreslení obr. 3.2.1a, přičemž lze snadno odvodit následující vztahy, tedy:

pro externí vstupní signály

$$w \equiv \boldsymbol{w}(t) = r(t) \qquad (3-07f)$$

pro výstupní řídicí signály z regulátoru

$$u \equiv \boldsymbol{u}(t) = u(t) \tag{3 – 07g}$$

pro měřené výstupy

$$y \equiv \boldsymbol{y}(t) = e(t) \tag{3 – 07h}$$

pro minimalizované či penalizované výstupy

$$z \equiv \boldsymbol{z}(t) = [z_1(t), z_2(t)]^T = \left[\widehat{W}_1(s) \cdot e(t), \widehat{W}_2(s) \cdot u(t)\right]^T \tag{3 – 07ch}$$

Pro zobecněnou soustavu obsahující váhové filtry platí dle [8]:

$$\hat{P}(s) = \begin{bmatrix} \hat{P}_{11}(s) & \hat{P}_{12}(s) \\ \hat{P}_{21}(s) & \hat{P}_{22}(s) \end{bmatrix} = \begin{bmatrix} \widehat{W}_1(s) & -\widehat{W}_1(s) \cdot \hat{G}(s) \\ 0 & \widehat{W}_2(s) \\ \hat{I} & -\hat{G}(s) \end{bmatrix} \tag{3 – 07i}$$

při

$$\hat{P}_{11}(s) = \widehat{W}_1(s) \cdot \left[\hat{I}, 0\right]^T = \left[\widehat{W}_1(s) \cdot \hat{I}, 0\right]^T = \left[\widehat{W}_1(s), 0\right]^T \tag{3 – 07j}$$

$$P_{12}(s) = \left[-\widehat{W}_1(s) \cdot \hat{G}(s), \widehat{W}_2(s) \cdot \hat{I}\right]^T = \left[-\widehat{W}_1(s) \cdot \hat{G}(s), \widehat{W}_2(s)\right]^T \tag{3 – 07k}$$

$$P_{21}(s) = \left[\hat{I}\right]^T = \hat{I} \tag{3 – 07l}$$

$$P_{22}(s) = \left[-\hat{G}(s)\right]^T = -\hat{G}(s) \tag{3 – 07m}$$

Váhové filtry $\widehat{W}_1(s)$ a $\widehat{W}_2(s)$ se v praxi velice často využívají; v takovém případě lze relaci (3 – 07b) formálně upravit do tvaru popisujícího účelovou funkci, tedy (pro SISO systém) dle [8]:

$$\min_{K_{stabilizuj\ ící}} \left\| \begin{matrix} \widehat{W}_1(s) \cdot \hat{S}(s) \\ \widehat{W}_2(s) \cdot \hat{K}(s) \cdot \hat{S}(s) \end{matrix} \right\|_\infty = \min_{K_{stabil\ izuj\ ící}} \left\| \begin{matrix} \widehat{W}_1(s) \cdot \left[1 + \hat{G}(s) \cdot \hat{K}(s)\right]^{-1} \\ \widehat{W}_2(s) \cdot \hat{K}(s) \cdot \left[1 + \hat{G}(s) \cdot \hat{K}(s)\right]^{-1} \end{matrix} \right\|_\infty \tag{3 – 07n}$$

Problém suboptimálního řešení

V obecném (neskalárním) tvaru relace (3 – 07b), tj. formální záměna skaláru jedna za jednotkovou matici příslušného rozměru, neexistuje jednoznačné řešení. Všeobecně řečeno, neexistuje žádný vzorec analytického řešení. Při praktickém návrhu obvykle stačí najít stabilizující regulátor $\hat{K}(s)$ takový, aby maximová norma (H∞ norma) přenosová funkce uzavřené smyčky odpovídala relaci, tedy dle [8]:

$$\left\|F_l\left[\hat{P}(s), \hat{K}(s)\right]\right\|_\infty < \gamma \tag{3 – 08a}$$

při

$$\gamma > \gamma_0 = \left\|F_l\left[\hat{P}(s), \hat{K}(s)\right]\right\|_\infty \tag{3 – 08b}$$

Relace (3 – 08b) představuje H∞ suboptimální problém. Při řešení takovéto optimalizační úlohy se začíná s počáteční hladinou kladného skaláru γ, jehož hodnota se snižuje tak dlouho, dokud nebude mít úloha

řešení. Pro získání uvedené počáteční hodnoty je nutno vyřešit LQG úlohu, obsahující vrchol přenosové funkce v uzavřené smyčce. Lze si představit, že pokud úspěšně redukujeme hodnotu γ, můžeme najít také optimální řešení úlohy. [7] [8]

Řešení pro normalizované systémy

Stavový (vnitřní) popis rozšířené soustavy na obr. 3.2.1a je dán relacemi, tedy dle [8]:

$$\dot{x}(t) = \hat{A} \cdot x(t) + \hat{B}_1 \cdot w(t) + \hat{B}_2 \cdot u(t) \qquad (3-09a)$$

$$z(t) = \hat{C}_1 \cdot x(t) + \hat{D}_{11} \cdot w(t) + \hat{D}_{12} \cdot u(t) \qquad (3-09b)$$

$$y(t) = \hat{C}_2 \cdot x(t) + \hat{D}_{21} \cdot w(t) \qquad (3-09c)$$

kde

$x(t) \in \mathbb{R}^n$	vektor stavů
$w(t) \in \mathbb{R}^{m_1}$	vektor vstupní referenční veličiny při $m_1 \geq p_2$
$u(t) \in \mathbb{R}^{m_2}$	vektor výstupní akční veličiny při $m_2 \leq p_1$
$z(t) \in \mathbb{R}^{p_1}$	vektor minimalizovaných či penalizovaných výstupů
$y(t) \in \mathbb{R}^{p_2}$	vektor měřených výstupů

Popřípadě lze rozšířenou matici zapsat ve tvaru tzv. pakované matice, tedy dle [8]:

$$\hat{P}(s) = \begin{bmatrix} \hat{P}_{11}(s) & \hat{P}_{12}(s) \\ \hat{P}_{21}(s) & \hat{P}_{22}(s) \end{bmatrix} = \begin{bmatrix} \hat{A} & \hat{B}_1 & \hat{B}_2 \\ \hat{C}_1 & \hat{D}_{11} & \hat{D}_{12} \\ \hat{C}_2 & \hat{D}_{21} & \hat{D}_{22} \end{bmatrix} \qquad (3-09d)$$

Pokud neexistuje přímá vazba mezi vstupem $u(t)$ a výstupem $y(t)$, platí $\hat{D}_{22} = 0$. Tento předpoklad je odůvodněný ve většině reálných průmyslových řídicích systémů. [8]

H∞ řešení je dáno řešením dvou algebraických Riccatiho rovnic. Riccatiho rovnice ve tvaru dle [8]:

$$\hat{A}^T \cdot \hat{X} + \hat{X} \cdot \hat{A} - \hat{X} \cdot \hat{R} \cdot \hat{X} + \hat{Q} = \hat{A}^T \cdot \hat{X} + \hat{X} \cdot \hat{A} - \hat{X} \cdot \hat{R}^T \cdot \hat{X} + \hat{Q}^T = 0 \qquad (3-09e)$$

Relace (3 – 09e) jednoznačně koresponduje Hamiltonově matici ve tvaru dle [8]:

$$\hat{H} = \begin{bmatrix} \hat{A} & -\hat{R} \\ -\hat{Q} & \hat{A}^T \end{bmatrix} \qquad (3-09f)$$

Stabilizační řešení $\hat{X}$ (pokud existuje) je reprezentován symetrickou maticí řešící Riccatiho rovnici a je takové, že $\left[\hat{A} - \hat{R} \cdot \hat{X}\right]$ je stabilní matice. Matematicky lze stabilizující řešení uvést ve tvaru dle [8]:

$$\hat{X} = Ric\left[\hat{H}\right] \qquad (3-09g)$$

Definujme relace dle [8]:

$$\hat{R}_n = [\hat{D}_{11} \quad \hat{D}_{12}]^T \cdot [\hat{D}_{11} \quad \hat{D}_{12}] - \begin{bmatrix} \gamma^2 \cdot \hat{I}_{m_1} & 0 \\ 0 & 0 \end{bmatrix} \qquad (3-09h)$$

a

$$\widetilde{\hat{R}_n} = [\hat{D}_{11} \quad \hat{D}_{21}]^T \cdot [\hat{D}_{11} \quad \hat{D}_{21}] - \begin{bmatrix} \gamma^2 \cdot \hat{I}_{p_1} & 0 \\ 0 & 0 \end{bmatrix} \qquad (3-09ch)$$

K výše uvedeným relacím (3 – 09h) a (3 – 09ch) lze rovněž najít stabilizující řešení využívající dvě Hamiltonovy matice dle [8]:

$$\hat{X} = Ric\left\{\begin{bmatrix} \hat{A} & 0 \\ -\hat{C}_1^T \cdot \hat{C}_1 & -\hat{A}^T \end{bmatrix} - \begin{bmatrix} \hat{B} \\ -\hat{C}_1^T \cdot [\hat{D}_{11} \quad \hat{D}_{12}]^T \end{bmatrix} \cdot \hat{R}_n^{-1} \cdot [\hat{D}_{11} \quad \hat{D}_{12}] \cdot \hat{C}_1 \cdot \hat{B}^T\right\} \qquad (3-09i)$$

a

$$\hat{Y} = Ric\left\{\begin{bmatrix} \hat{A}^T & 0 \\ -\hat{B}_1 \cdot \hat{B}_1^T & -\hat{A} \end{bmatrix} - \begin{bmatrix} \hat{C}^T \\ -\hat{B}_1 \cdot [\hat{D}_{11} \quad \hat{D}_{21}] \end{bmatrix} \cdot \widetilde{\hat{R}_n^{-1}} \cdot [\hat{D}_{11} \quad \hat{D}_{21}]^T \cdot \hat{B}_1^T \cdot \hat{C}\right\} \qquad (3-09j)$$

Aby vůbec mohlo suboptimální řešení existovat, musí být pro něj stanoveny nutné a postačující podmínky:

- $[\hat{A}, \hat{B}_2]$ *je stabilizovatelná a* $[\hat{C}_2, \hat{A}]$ *je detekovatelná*
- $\hat{D}_{12} = [0 \quad I_{m_2}]^T$ a $\hat{D}_{21} = [0 \quad I_{p_2}]$
- $\forall\, \omega \in \mathbb{R}: rank \begin{bmatrix} \hat{A} - j \cdot \omega \cdot \hat{I} & \hat{B}_2 \\ \hat{C}_1 & \hat{D}_{12} \end{bmatrix} = n + m_2$
- $\forall\, \omega \in \mathbb{R}: rank \begin{bmatrix} \hat{A} - j \cdot \omega \cdot \hat{I} & \hat{B}_1 \\ \hat{C}_2 & \hat{D}_{21} \end{bmatrix} = n + p_2$

3.3 Návrh regulátorů

Na základě výše uvedených teoretických poznatků z oblasti H∞ robustního řízení lze navrhnout robustní regulátor dle zvolené koncepce, kdy se zaobíráme otázkou autonomnosti řízení a problémem smíšené citlivosti, přičemž vážíme také vstupní signály. Pro návrh regulátorů jsou klíčové odvozené matematické modely v elevaci a v azimutu, kdy navrhujeme regulátor jak pro matematický model, tak rovněž pro model reálný. Nakonec jsou obě odezvy vzájemně porovnány.

3.3.1 Požadavek autonomnosti

Hlavní problém tohoto MIMO systému spočívá především eliminací vazby elevace-azimut. Celkově samozřejmě chceme eliminovat obě vazby, ovšem víme, že vazba azimut-elevace není tak významná. Koncepce tudíž vychází z předpokladu navržení dvou samostatných regulátorů, tj. elevačního regulátoru a azimutového regulátoru.

Definice autonomnosti praví, že žádaná veličina $w(t)$ musí ovlivňovat pouze jedinou odpovídající výstupní veličinu $y(t)$. [11]

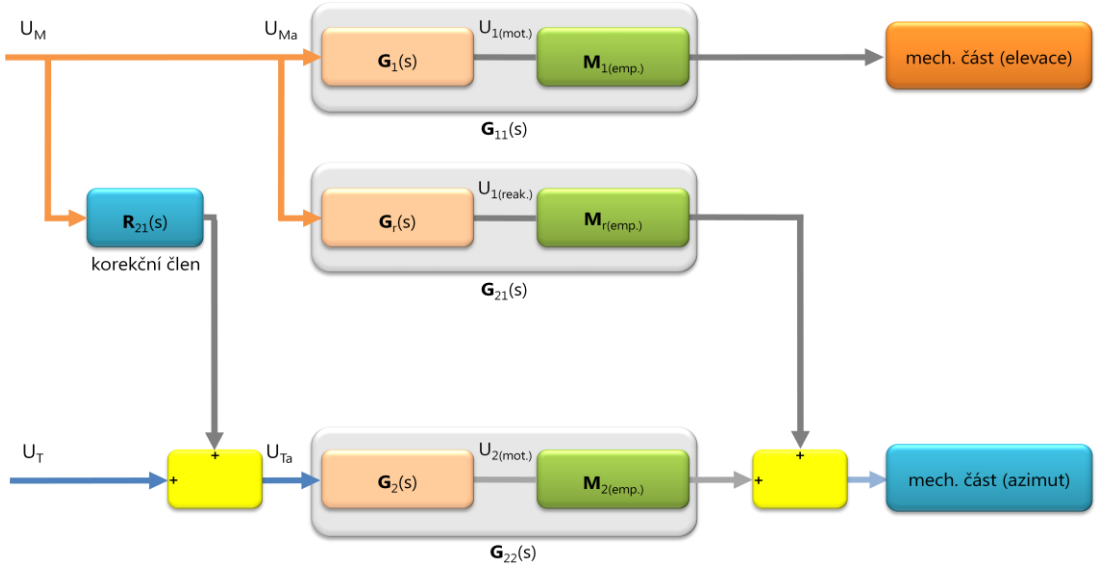
Autonomnost rovněž požaduje, aby přenosová matice otevřené smyčky byla diagonální – prvky matice se nachází pouze na hlavní diagonále. A priori diagonalitu vyžadujeme u matice řízení. S ohledem na explicitně zapsané znaménko je přenosová funkce korekčního členu dána obecnou relací vycházející z teorie determinantů matic (Laplaceův doplněk) dle [12]:

$$\hat{R}_{i,j}(s) = (-1)^{j+1} \cdot \hat{R}_{j,j}(s) \cdot \left|\frac{\hat{G}^*_{j,i}(s)}{\hat{G}^*_{j,j}(s)}\right| \qquad (3-10a)$$

kde i index řádku matice
j index sloupce matice

$(-1)^{j+1}$ Laplaceův (algebraický) doplněk
$|\hat{G}^*_{j,i}|$ determinant matice $\hat{G}^*_{j,i}(s)$
$|\hat{G}^*_{j,j}|$ determinant matice $\hat{G}^*_{j,j}(s)$

Relace (3 – 10a) má spíše teoretický charakter. Než si přesně pamatovat přesně jeho tvar a neudělat chybu algebraického doplňku, vyplatí se při malém počtu regulovaných veličin (v našem případě to jsou dvě veličiny) přímé odvozené podmínek autonomnosti. Situaci dokládá obr. 3.3.1, přímo modifikovaný pro model vrtulníku.



Obr. 3.3.1: blokové schéma pro eliminaci vazby elevace-azimut; vazba azimut-elevace je zanedbána

Jestliže akční zásah vyjádřený napětím U_M vyvolá nežádoucí reakci na vstupu azimutové mechanické části popsanou Laplaceovým obrazem $\hat{G}_{21}(s) \cdot \hat{U}_M(s)$, pak korekčním členem $\hat{R}_{21}(s)$ ji lze zcela vykompenzovat za podmínky dle [12]:

$$\hat{G}_{21}(s) \cdot \hat{U}_M(s) + \hat{G}_{22}(s) \cdot \hat{R}_{21}(s) \cdot \hat{U}_M(s) = 0 \Rightarrow \hat{R}_{21}(s) = -\frac{\hat{G}_{21}(s)}{\hat{G}_{22}(s)} \qquad (3-10b)$$

Přenos druhého korekčního členu lze buď napsat přímo využitím pravidla o cyklické záměně indexů, nebo opět sestavit podmínkovou rovnici dle relace (3 – 10b), tedy dle [12]:

$$\hat{G}_{12}(s) \cdot \hat{U}_T(s) + \hat{G}_{11}(s) \cdot \hat{R}_{12}(s) \cdot \hat{U}_T(s) = 0 \Rightarrow \hat{R}_{12}(s) = -\frac{\hat{G}_{12}(s)}{\hat{G}_{11}(s)} \qquad (3-10c)$$

Relace (3 – 10c) popisuje vazbu azimut-elevace, která však není významná, tudíž lze psát dle [12]:

$$\hat{R}_{12}(s) = \hat{G}_{12}(s) = 0 \qquad (3-10d)$$

Schéma na obr. 3.3.1 je možno interpretovat také tak, že bez principiálního (konstrukčního) zásahu do samotné soustavy nelze sice odstranit interní fyzikální vazbu mezi elevací a azimutem ve formě přenosu $\hat{G}_{21}(s)$. Ovšem lze relativně snadno realizovat externí propojení mezi vstupy soustavy vrtulníku pomocí korekčního přenosu $\hat{R}_{21}(s)$, což zabezpečí totéž, co neproveditelné odstranění křížové vazby uvnitř přenosové matice MIMO systému – rozložení modelu na dva, zdánlivě se neovlivňující modely. Jejich regulaci pak zabezpečí dvě na sobě nezávislé regulační smyčky. [12]

Dosadíme-li do relace (3 – 10c), dostáváme přenos korekčního členu, tedy:

$$\hat{R}_{21}(s) = -\frac{T_{r_2}\cdot s^2+T_{r_1}\cdot s+1}{(T_1\cdot s^2+1)^2}\cdot\frac{a_r\cdot U_M\cdot s+b_r}{a_2\cdot U_T\cdot s+b_2}\cdot(T_2\cdot s^2+1)^2 \qquad (3-10e)$$

Z relace (3 – 10e) je zřejmé, že přenosová funkce v takovémto tvaru nesplňuje podmínku fyzikátní realizovatelnosti, neboť stupeň čitatele je o dva řády vyšší než stupeň jmenovatele. Proto definujme dvě setrvačnosti s podmínkami $\xi_1 \ll T_{r_1}$ a $\xi_2 \ll T_{r_2}$. Fyzikálně realizovatelný korekční přenos eliminující vazbu elevace-azimut je dán relací, tedy:

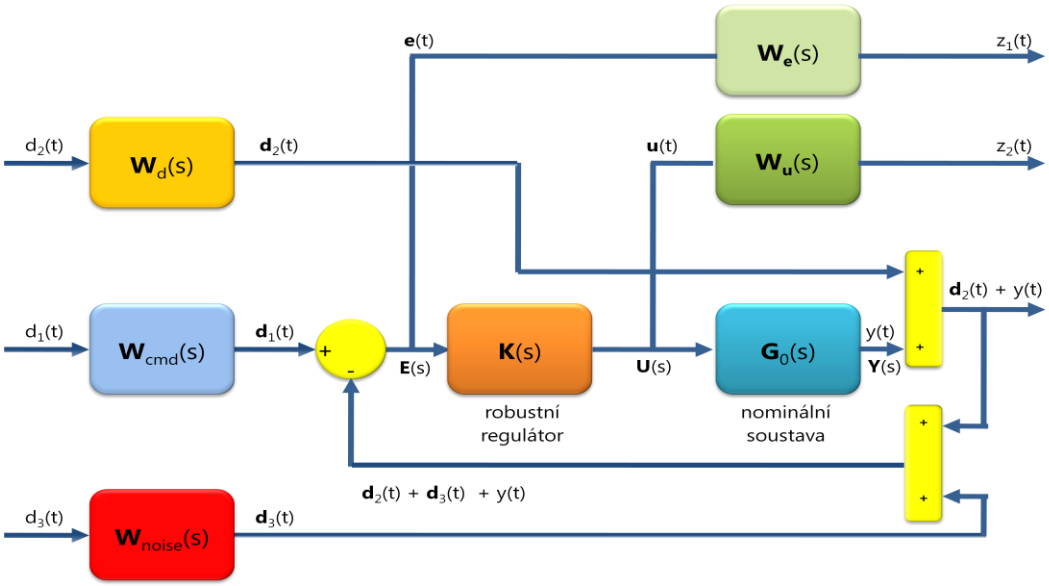
$$\hat{R}_{21}(s) = -\frac{T_{r_2}\cdot s^2+T_{r_1}\cdot s+1}{\xi_2\cdot s^2+\xi_1\cdot s+1}\cdot\frac{a_r\cdot 0{,}55\cdot s+b_r}{a_2\cdot 0{,}2\cdot s+b_2}\cdot\frac{(T_2\cdot s+1)^2}{(T_1\cdot s+1)^2} \qquad (3-10f)$$

3.3.2 Koncepce regulátorů

Částečně byla koncepce návrhu robustní regulace modelu vrtulníku objasněna v požadavku na autonomnost řízení elevace a azimutu. Druhá část se již týká H∞ robustní regulace, konkrétně pak modifikace problému smíšené citlivosti (MSP), kdy navíc vedle regulační odchylky $e(t)$ a akční veličiny $u(t)$ penalizujeme také skupinu externích vstupních signálů:

- $d_1(t) = r(t) = w(t)$ reference čili externí řídicí veličina
- $d_2(t)$ nízkofrekvenční poruchová veličina (porucha)
- $d_3(t)$ vysokofrekvenční poruchová veličina (šum)

Pro penalizaci signálů vstupujících a vystupujících z rozšířené soustavy v elevaci či v azimutu využíváme váhové přenosy nebo též váhové filtry. Celkové blokové schéma regulátoru a rozšířené soustavy je následující, tedy:



Obr. 3.3.2: blokové schéma propojení H∞ robustního regulátoru, nominální soustavy a váhových filtrů pro model vrtulníku

Existuje mnoho různých způsobů, jak lze nominální soustavu rozšířit. Nicméně čím více externích vstupů a penalizovaných (chybových) výstupů máme k dispozici, tím náročnější bude volba váhových filtrů. Protože váhové filtry jsou obecně stabilní přenosy (nemusí se nutně jednat o ryzí racionální lomenou funkci) určitého řádu. Tudíž čím více jich přidáme, tím vyšší řád bude výsledný systém mít. Takto propojený systém posléze využíváme k vyjádření stavového popisu, respektive přenosové funkce H∞ optimálního, respektive suboptimálního robustního regulátoru $\widehat{K}(s)$ s jedním stupněm volnosti (konfigurace 1DOF).

Význam jednotlivých bloků je následující dle [13]:

- $\widehat{W}_{cmd}(s)$
  Tento váhový přenos se stará o sledování reference. Na vstupu se objevuje normalizovaný signál, na výstupu pak signál v příslušných fyzikálních jednotkách.
- $\widehat{W}_{d}(s)$
  Tento váhový přenos upravuje frekvenční a amplitudové charakteristiky vnějších nízkofrekvenčních poruchových signálů, působících na nominální soustavu.
- $\widehat{W}_{noise}(s)$
  Tento váhový přenos představuje modely šumů senzorů ve frekvenční doméně. Snaží se zachytit určitou informaci, odvozenou z laboratorních pokusů nebo výrobních měření, v řízení. Šum samozřejmě vykazuje vysokofrekvenční charakter.
- $\widehat{W}_{e}(s)$
  Tento váhový přenos penalizuje řídicí signál z robustního regulátoru, tudíž penalizuje regulační odchylku. Stanovuje převrácenou hodnotu očekávaného tvaru výstupní veličiny. Na vstupu filtru se objevuje signál v příslušných fyzikálních jednotkách a na výstup je normalizovaný.
- $\widehat{W}_{u}(s)$
  Tento váhový přenos penalizuje řídicí signál z robustního regulátoru, tudíž penalizuje akční veličinu. Stanovuje převrácenou hodnotu očekávaného tvaru výstupní veličiny. Na vstupu filtru se objevuje signál v příslušných fyzikálních jednotkách a na výstup je normalizovaný.

Kompletní výpočet nejen přenosové funkce H∞ robustního regulátoru, ale také klíčové H∞ normy se provádí v MATLABu. Základ výpočtového algoritmu tvoří správné propojení nominální soustavy a váhových filtrů v korespondenci s obr. 3.3.2. Programové řešení v M-souboru vypadá následovně, tedy:

```
systemnames = 'G Wcmd Wd Wnoise We Wu';
inputvar = '[d1; d2; d3; u]';
outputvar = '[We; Wu; Wcmd-G-Wnoise]';
input_to_Wcmd = '[d1]';
input_to_Wd = '[d2]';
input_to_Wnoise = '[d3]';
input_to_G = '[u]';
input_to_We = '[Wcmd-G-Wnoise]';
input_to_Wu = '[u]';
cleanupsysic = 'yes';
P = sysic
NControl = 1;
NMeasure = 1;
r = [NControl NMeasure];
[K, CL, gopt] = hinfsyn(P, NMeasure, NControl);
```

Napřed definujeme, jaké systémy budeme propojovat – proměnná `systemnames`. Pak definujeme vektor vstupních signálů (externí řídicí signály a řídicí signál z regulátoru) – proměnná `inputvar`. Výstup je

reprezentován proměnnou outputvar obsahující penalizaci regulační odchylky (`We`), akční veličiny (`Wu`) a měřený výstup (`Wcmd-G-Wnoise`). Následně propojíme všechny vstupy do váhových filtrů. Příkazem `cleanupsysic` s nastavenou hodnotou atributu na `yes` říkáme, že chceme, aby proměnné `systemnames`, `inputvar` a `outputvar` byly ihned po vytvoření propojení systému odstraněny z pracovního prostředí (Workspace) MATLABu.

Proměnná `P` představuje rozšířenou soustavu či systémové propojení (`sysic`, System Interconnection). Aby byl výčet parametrů pro výpočet kompletní, musíme definovat počet řídicích výstupů z regulátoru (`NControl`) a počet měřených výstupů (`NMeasure`). Výpočet H∞ regulátoru (`K`), přenosu uzavřené smyčky (`CL`) a maximovou normu přenosu uzavřené smyčky (`gopt`) získáme zavoláním funkce `hinfsyn` s parametry `P`, `NMeasure` a `NControl`.

Volání funkce `hinfsyn` se dá samozřejmě rozšířit o více vstupních a výstupních parametrů, přičemž v tomto konkrétním případě dle [13]:

- jsou řešeny dvě algebraické Riccatiho rovnice
- $\gamma \in (0, +\infty)$
- přenos uzavřené smyčky se počítá pomocí lin. fraktální transformace $CL = F\{\hat{P}(s), \hat{K}(s)\}$
- $\gamma_0 = \|CL\|_\infty = \left\|F\{\hat{P}(s), \hat{K}(s)\}\right\|_\infty$

MATLAB, konkrétně pak Robust Control Toolbox, obsahuje další funkce, s nimiž lze problematiku návrhu spojitého nebo diskrétního H∞ robustního regulátoru řešit. Pro úplnost pouze uveďme prototyp funkce zaměřené na problematiku klasické smíšené citlivosti:

- `[K, CL, gopt, INFO] = mixsyn(G, W1, W2, W3)`

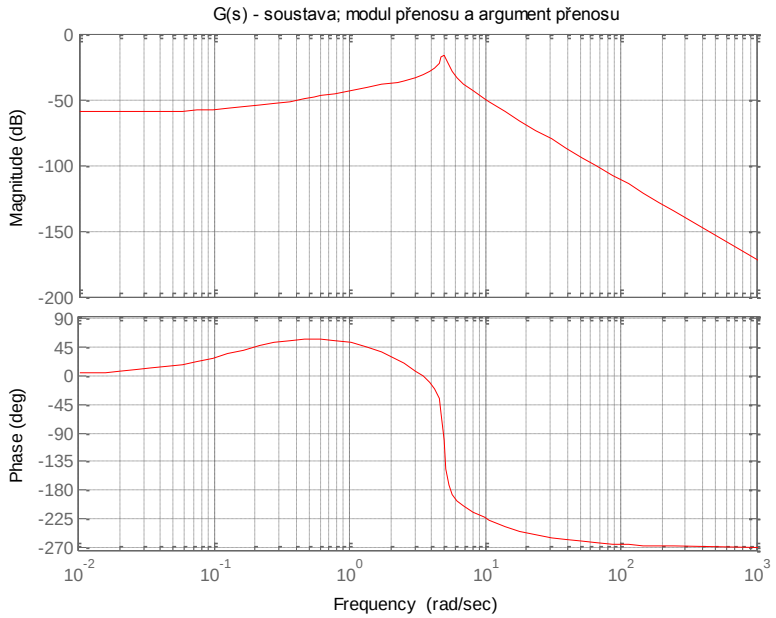
Problémem funkce `mixsyn` je jednak počet váhových filtrů a jednak jejich charakter, protože penalizují pouze regulační odchylku (`W1`), akční veličinu (`W2`) a měřený výstup (`W3`). Vzhledem ke zvolené koncepci návrhu by nebylo možno pomocí této funkce penalizovat vstupy.

Elevační regulátor pro matematický model

Přenosová funkce dynamiky matematického modelu v elevaci je dle relace (2 – 33b) dána relací, tedy:

$$\hat{G}_\psi(s) = \frac{\hat{\Psi}(s)}{\hat{U}_M(s)} = \frac{13}{1024} \cdot \frac{0{,}105 \cdot 0{,}55 \cdot s + 0{,}00936}{(0{,}25 \cdot s + 1)^2 \cdot \left\{0{,}00437 \cdot s^2 + 0{,}00184 \cdot s + (-0{,}121) \cdot (-0{,}85) \cdot cos\left(\frac{77}{1024} \cdot \pi\right)\right\}} \qquad (3-11a)$$

Amplitudová frekvenční charakteristika (modul přenosu) a fázová frekvenční charakteristika (argument přenosu) vypadají následovně:



Obr. 3.3.3a: amplitudová frekvenční charakteristika a fázová frekvenční charakteristika zadané přenosové funkce matematického modelu v elevaci

Z amplitudové frekvenční charakteristiky na obr. 3.3.3a je zřejmé, že obsahuje nejvyšší hodnotu při:

- **frekvence:** $\omega_{MAX} = 4{,}9448\ [rad\ \cdot s^{-1}]$
- **modul přenosu:** $\left|\hat{G}_{\psi}(j\cdot\omega)\right|_{MAX} = -16{,}05\ [dB] = 10^{-\frac{16{,}05}{20}}\ [-] \cong 0{,}1576\ [-]$

Soustava je čtvrtého řádu a obsahuje čtyři stabilní póly – z toho dva komplexně sdružené a jeden dvojnásobný pól:

$p_1 = -0{,}2105 + j\cdot 4{,}9448$

$p_2 = \overline{p_1} = -0{,}2105 - j\cdot 4{,}9448$

$p_3 = p_4 = -4$

Maximová norma zadané soustavy činí (v souladu s maximální hodnotou modulu přenosu):

$\left\|\hat{G}_{\psi}(s)\right\|_{\infty} = 0{,}1577\ [-]$

Tvary váhových filtrů pro zadanou soustavu jsou následující, tedy:

váhový přenos pro referenci

$$\hat{W}_{cmd}(s) = \frac{1}{0{,}25\cdot s+1} \qquad (3-11b)$$

váhový přenos pro nízkofrekvenční poruchovou veličinu

$$\hat{W}_{d}(s) = \frac{0{,}5}{0{,}1\cdot s+1} \qquad (3-11c)$$

váhový přenos pro vysokofrekvenční poruchovou veličinu

$$\widehat{W}_{noise}(s) = \frac{0{,}01 \cdot s+1}{s+1} \qquad (3-11d)$$

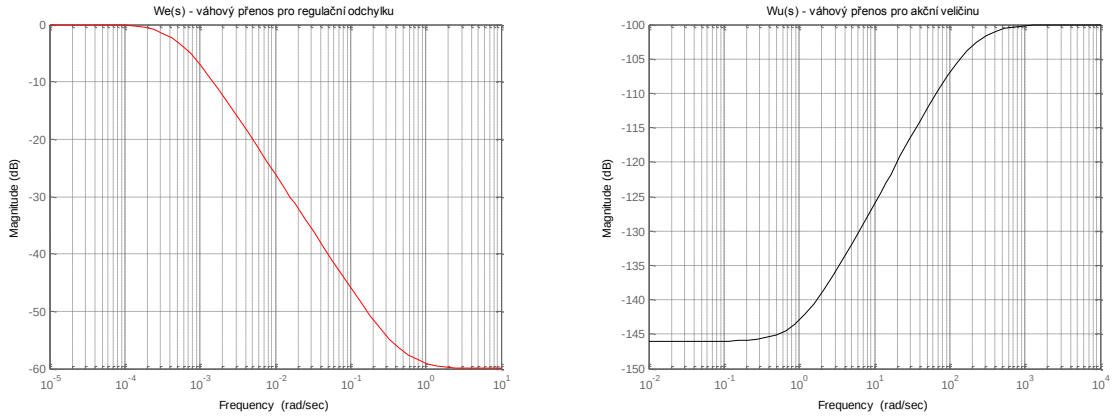
váhový přenos pro regulační odchylku

$$\widehat{W}_e(s) = K_e \cdot \frac{\frac{1}{M_e} \cdot s+\omega_{be}}{s+\omega_{be} \cdot \varepsilon_e} = 0{,}001 \cdot \frac{s+0{,}5}{s+0{,}5 \cdot 0{,}001} = 0{,}001 \cdot \frac{s+0{,}5}{s+0{,}0005} \qquad (3-11d)$$

váhový přenos pro akční veličinu

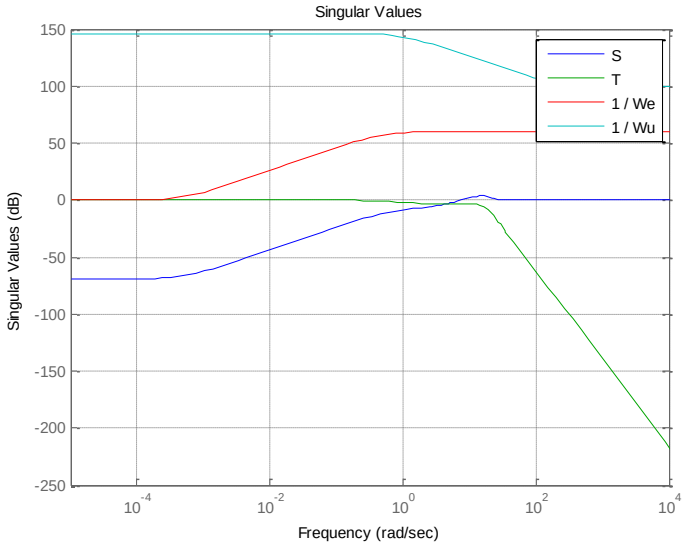
$$\widehat{W}_u(s) = K_u \cdot \frac{s+\frac{\omega_{bu}}{M_u}}{\varepsilon_u \cdot s+\omega_{bu}} = 10^{-7} \cdot \frac{s+\frac{2}{2}}{0{,}01 \cdot s+2} = 10^{-7} \cdot \frac{s+1}{0{,}01 \cdot s+2} \qquad (3-11e)$$

Váhové filtry pro regulační odchylku a akční veličinu mají předepsaný tvar přenosové funkce dle [14] – přenos obsahuje vždy stejný řád čitatele a jmenovatele, aby byla zajištěna také stabilita převráceného přenosu. Filtr pro regulační odchylku je dolnopropustný a pro akční veličinu hornopropustný.



Obr. 3.3.3b: amplitudové frekvenční charakteristiky dolnopropustného váhového filtru $\widehat{W}_e(s)$ (vlevo) a hornopropustného váhového filtru $\widehat{W}_u(s)$ (vpravo) – matematický model v elevaci

Grafické závislosti citlivostních funkcí a převrácených přenosů pro matematický model v elevaci jsou následující:

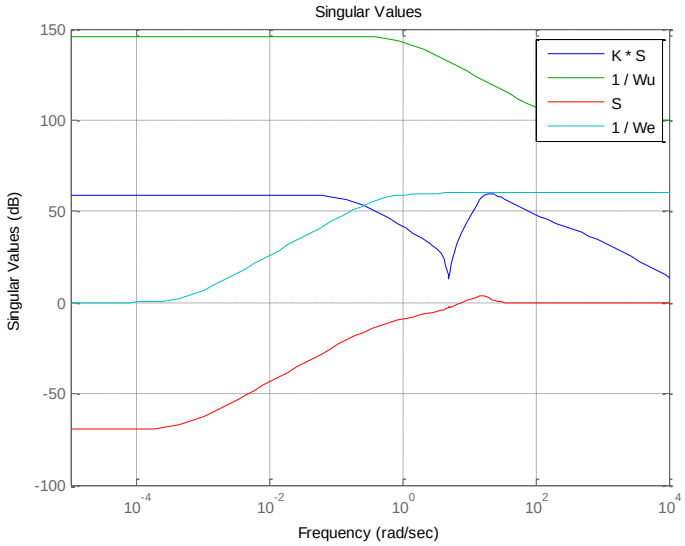


Obr. 3.3.3b: amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$, komplementární citlivostní funkce $\hat{T}(s)$, převrácených přenosů $1/\widehat{W}_e(s)$ a $1/\widehat{W}_u(s)$ – matematický model v elevaci

Jeden z obecných požadavků na amplitudové frekvenční charakteristiky citlivostní funkce $\hat{S}(s)$ a převráceného přenosu $1/\widehat{W}_e(s)$ definuje robustní chování:

$$\forall\, \omega \in \mathbb{R}: \left|\hat{S}(j\cdot\omega)\right| \leq \left|1/\widehat{W}_e(j\cdot\omega)\right| = 1/\left|\widehat{W}_e(j\cdot\omega)\right| \Leftrightarrow \left\|\widehat{W}_e(s)\cdot\hat{S}(s)\right\|_\infty \leq 1 \qquad (3-11f)$$

Z obr. 3.3.3b plyne, že relace (3 – 11f) je zcela splněna.



Obr. 3.3.3c: amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$, součinu obrazů $\widehat{K}(s)\cdot\hat{S}(s)$, převrácených přenosů $1/\widehat{W}_e(s)$ a $1/\widehat{W}_u(s)$ – matematický model v elevaci

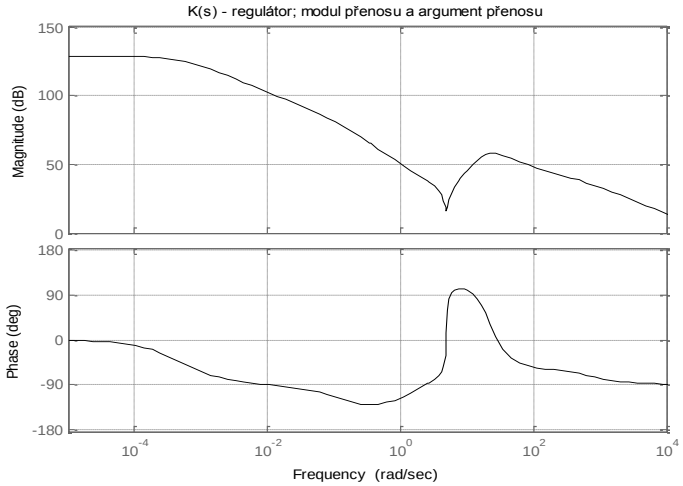
Obdobně lze definovat také pro amplitudovou frekvenční charakteristiku pro převrácený přenos $1/\widehat{W}_u(s)$ a součin přenosu regulátoru a citlivostní funkce $\widehat{K}(s)\cdot\hat{S}(s)$:

$$\forall\, \omega \in \mathbb{R}: \left|K(j\cdot\omega)\cdot\hat{S}(j\cdot\omega)\right| \leq \left|1/\hat{W}_u(j\cdot\omega)\right| \Leftrightarrow \left\|\hat{W}_u(s)\cdot\hat{K}(s)\cdot\hat{S}(s)\right\|_\infty \leq 1 \qquad (3-11g)$$

Z obr. 3.3.3c plyne, že relace (3 − 11g) je zcela splněna. Pro úplnost uveďme velikosti klíčových H∞ norem, tedy:

- **optimální H∞ norma:** $\gamma = 5{,}7520\cdot 10^{-4}$
- **H∞ norma uzavřené smyčky:** $\left\|F\left\{\hat{P}(s),\hat{K}(s)\right\}\right\|_\infty = 4{,}7837\cdot 10^{-4} < \gamma$
- **H∞ norma citlivostní funkce:** $\left\|\hat{S}(s)\right\|_\infty = 1{,}5357$
- **H∞ norma kompl. citlivostní funkce:** $\left\|\hat{T}(s)\right\|_\infty = 0{,}9997$

Řád navrženého H∞ regulátoru pro matematický model elevace odpovídá součtu řádů jednotlivých prvků rozšířené soustavy – nominální soustava je nejvýše čtvrtého řádu a všech pět váhových je nejvýše prvního řádu. Z uvedeného plyne, že regulátor bude systém nejvýše devátého řádu.

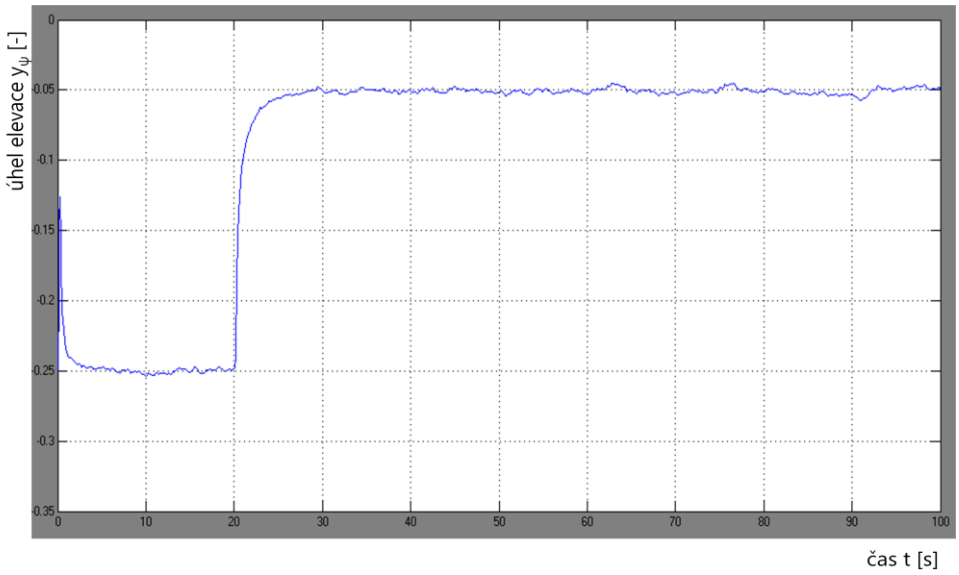


Obr. 3.3.3d: amplitudová frekvenční charakteristika a fázová frekvenční charakteristika H∞ regulátoru pro matematický model v elevaci

Podle obr. 3.3.2 byl v Simulinku vytvořen model H∞ regulátoru a rozšířené soustavy. Konfigurace skupiny externích vstupních signálů je následující:

- **reference:** $d_1(t=0) = r(t=0) = -0{,}25\ [-]$ pro $t \in (0;20\rangle$
  $d_1(t) = r(t=20) = -0{,}05\ [-]$ pro $t \in (20;100\rangle$
- **NF porucha:** není zahrnuta do modelu
- **VF porucha:** pásmově omezený bílý šum při výkonu $0{,}00001\ [W]$

Odezva namodelovaného elevačního systému na referenci $d_1(t)$ je následující:



Obr. 3.3.3e: závislost výstupního úhlu elevace y_ψ na čase t v matematickém modelu elevace; $y_\psi = f(t)$

Díky vyváženému poměru citlivostní funkce a komplementární citlivostní funkce bude eliminace šumu i poruchy účinná.

Elevační regulátor pro reálný model

Při návrhu regulátoru pro reálný model v elevaci využijeme přenosovou funkci z matematického modelu, protože regulátor se vytváří na základě přenosových funkcí nominální soustavy a váhových filtrů.

Tvary váhových filtrů pro zadanou soustavu jsou následující, tedy:

váhový přenos pro referenci

$$\widehat{W}_{cmd}(s) = \frac{1}{0{,}25 \cdot s + 1} \qquad (3-12a)$$

váhový přenos pro nízkofrekvenční poruchovou veličinu

$$\widehat{W}_{d}(s) = \frac{0{,}5}{0{,}1 \cdot s + 1} \qquad (3-12b)$$

váhový přenos pro vysokofrekvenční poruchovou veličinu

$$\widehat{W}_{noise}(s) = \frac{0{,}01 \cdot s + 1}{s + 1} \qquad (3-12c)$$

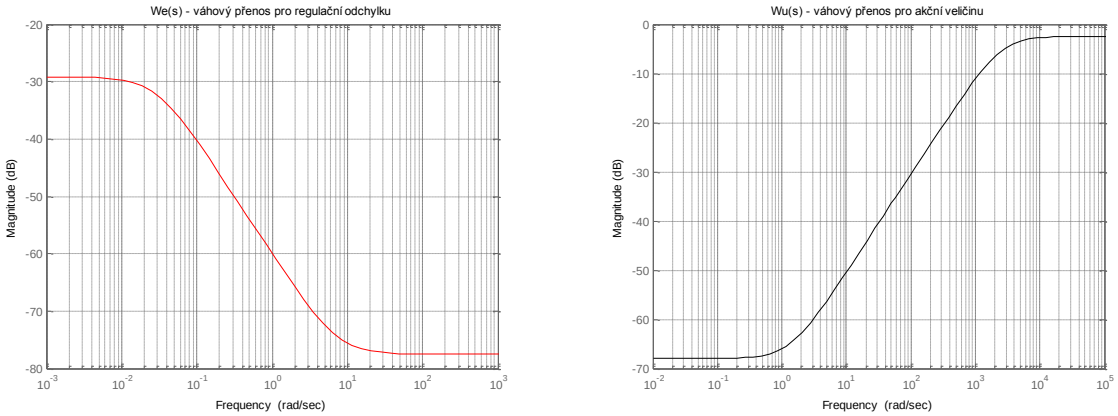
váhový přenos pro regulační odchylku

$$\widehat{W}_{e}(s) = 0{,}003 \cdot \frac{0{,}1s + 0{,}75}{2{,}25 \cdot s + 0{,}065} = \frac{0{,}003}{2{,}25} \cdot \frac{\frac{s}{10} + 0{,}75}{s + \frac{0{,}75 \cdot \frac{0{,}065}{0{,}75}}{2{,}25}} = \frac{0{,}003}{2{,}25} \cdot \frac{\frac{s}{10} + 0{,}75}{s + 0{,}75 \cdot \frac{0{,}065}{2{,}25 \cdot 0{,}75}} \qquad (3-12d)$$

váhový přenos pro akční veličinu

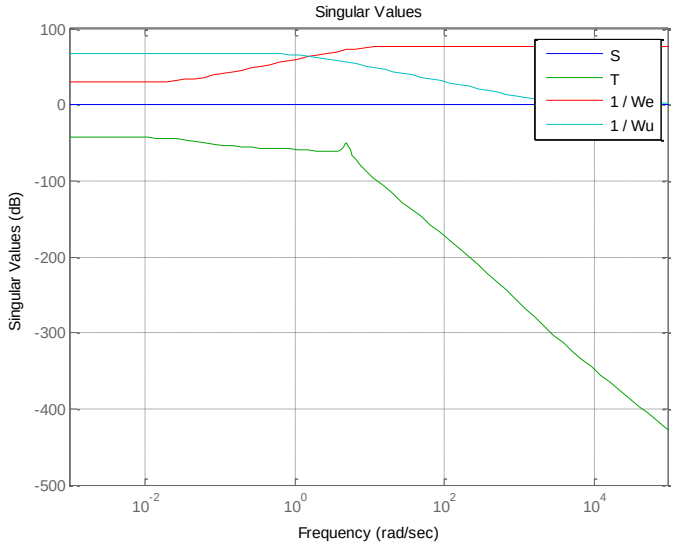
$$\widehat{W}_{u}(s) = \frac{s + 2500}{0{,}75 \cdot s + 1} = 1 \cdot \frac{s + \frac{1}{4 \cdot 10^{-4}}}{0{,}75 \cdot s + 1} \qquad (3-12e)$$

V porovnání s matematickým modelem elevace byla u reálného ponechána stejná penalizace skupiny vstupních externích signálů, kdežto v případě penalizace chybových výstupů byly váhové filtry více přizpůsobeny reálnému modelu. Jejich charakter však zůstal stejný.



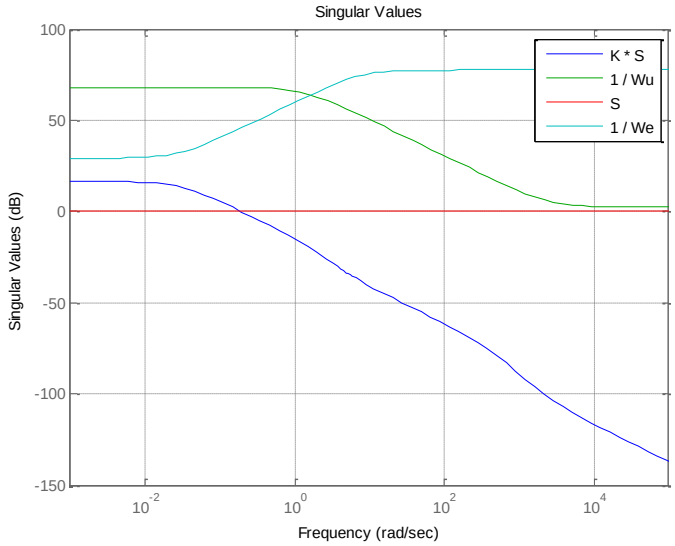
Obr. 3.3.4a: amplitudové frekvenční charakteristiky dolnopropustného váhového filtru $\widehat{W}_e(s)$ (vlevo) a hornopropustného váhového filtru $\widehat{W}_u(s)$ (vpravo) – reálný model v elevaci

Grafické závislosti citlivostních funkcí a převrácených přenosů pro reálný model elevace jsou následující:



Obr. 3.3.4b: amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$, komplementární citlivostní funkce $\hat{T}(s)$, převrácených přenosů $1/\widehat{W}_e(s)$ a $1/\widehat{W}_u(s)$ – reálný model v elevaci

Z obr. 3.3.4b je zřejmé, že amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$ a převráceného přenosu $1/\widehat{W}_e(s)$ splňuje podmínku uvedenou v relaci (3 – 11f).

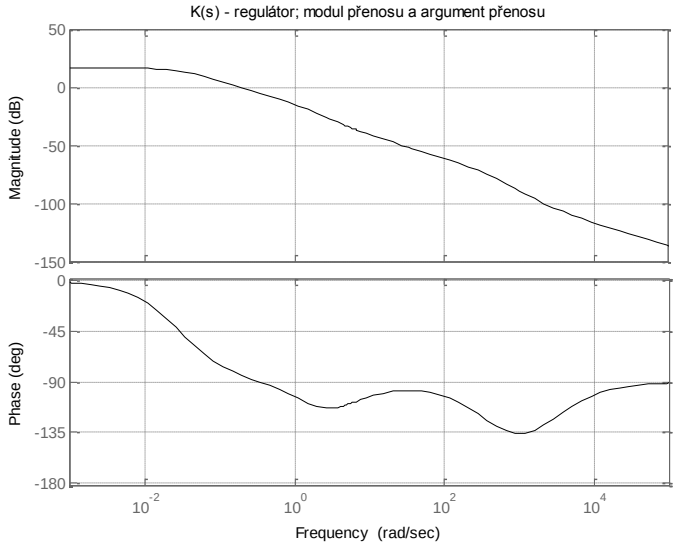


Obr. 3.3.4c: amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$, součinu obrazů $\hat{K}(s) \cdot \hat{S}(s)$, převrácených přenosů $1/\widehat{W}_e(s)$ a $1/\widehat{W}_u(s)$ – reálný model v elevaci

Z obr. 3.3.4c je zřejmé, že amplitudová frekvenční charakteristika součinu Laplaceových obrazů $\hat{K}(s) \cdot \hat{S}(s)$ a převráceného přenosu $1/\widehat{W}_u(s)$ splňuje podmínku uvedenou v relaci (3 – 11g). Pro úplnost uveďme velikosti klíčových H∞ norem, tedy:

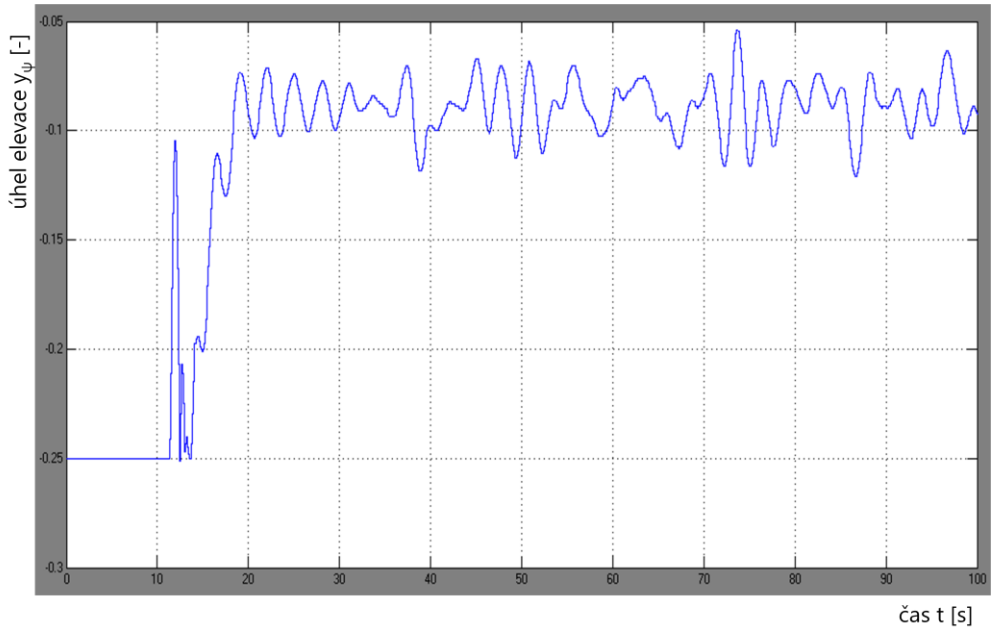
- **optimální H∞ norma:** $\gamma = 0{,}0512$
- **H∞ norma uzavřené smyčky:** $\left\| F\{\hat{P}(s), \hat{K}(s)\} \right\|_\infty = 0{,}0487 < \gamma$
- **H∞ norma matice:** $\left\| \begin{matrix} W_e(s) \cdot \hat{S}(s) \\ W_u(s) \cdot \hat{K}(s) \cdot \hat{S}(s) \end{matrix} \right\|_\infty = 0{,}0344 < \gamma$
- **H∞ norma citlivostní funkce:** $\left\| \hat{S}(s) \right\|_\infty = 1{,}0028$
- **H∞ norma kompl. citlivostní funkce:** $\left\| \hat{T}(s) \right\|_\infty = 0{,}0073$
- **H∞ norma citlivosti řízení:** $\left\| \hat{R}(s) \right\|_\infty = 6{,}6069$
- **H∞ norma otevřené smyčky:** $\left\| \hat{L}(s) \right\|_\infty = 0{,}0074$
- **H∞ norma regulátoru:** $\left\| \hat{K}(s) \right\|_\infty = 6{,}6557$

Z číselné interpretace H∞ norem je zřejmé, že se jedná o suboptimální řešení. Regulátor je, stejně jako v případě matematického modelu, nejvýše devátého řádu.



Obr. 3.3.4d: amplitudová frekvenční charakteristika a fázová frekvenční charakteristika H∞ regulátoru pro reálný model v elevaci

Blokové schéma v Simulinku je formálně stejné jako v případě matematického modelu v elevaci; jediný rozdíl tedy spočívá pouze v nominální soustavě, kterou tvoří reálný model vrtulníku. Konfigurace vstupů se také neliší od matematického modelu, neboť odezvy chceme vzájemně porovnat. Odezva namodelovaného elevačního systému na referenci $d_1(t)$ je následující:



Obr. 3.3.4e: závislost výstupního úhlu elevace y_ψ na čase t v reálném modelu elevace; $y_\psi = f(t)$

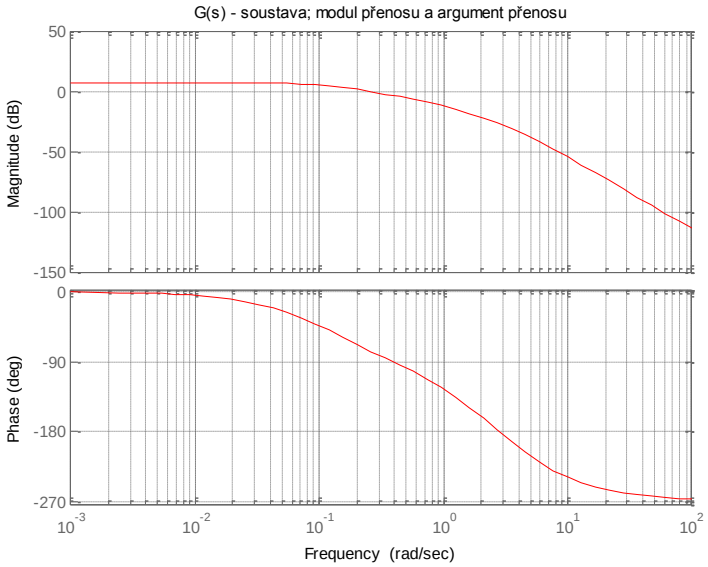
Díky převažující citlivostní funkci nebude eliminace šumu dosti účinná, což plyne z obr. 3.3.4a.

Azimutový regulátor pro matematický model

Přenosová funkce dynamiky matematického modelu v azimutu je dle relace (2 – 34b) dána relací, tedy:

$$\hat{G}_{\varphi}(s) = \frac{\hat{\Phi}(s)}{\hat{U}_T(s)} = \frac{13}{1024} \cdot \frac{0{,}033 \cdot 0{,}20 \cdot s + 0{,}0294}{(0{,}25 \cdot s + 1)^2 \cdot \{0{,}00414 \cdot s^2 + 0{,}00869 \cdot s\}} \qquad (3-13a)$$

Amplitudová frekvenční charakteristika (modul přenosu) a fázová frekvenční charakteristika (argument přenosu) vypadají následovně:



Obr. 3.3.5a: amplitudová frekvenční charakteristika a fázová frekvenční charakteristika zadané přenosové funkce matematického modelu v azimutu

Z amplitudové frekvenční charakteristiky na obr. 3.3.3a je zřejmé, že obsahuje nejvyšší hodnotu při:

- **frekvence:** $\omega_{MAX} \in (-\infty; 0{,}0437\rangle \; [rad \cdot s^{-1}]$
- **modul přenosu:** $\left|\hat{G}_{\varphi}(j \cdot \omega)\right|_{MAX} = +7{,}00 \; [dB] = 10^{\frac{7}{20}} \; [-] \cong 2{,}2388 \; [-]$

Soustava je čtvrtého řádu a obsahuje čtyři stabilní póly – z toho dva různé reálné a jeden dvojnásobný pól:

$p_1 = -0{,}1222$

$p_2 = -1{,}9768$

$p_3 = p_4 = -4$

Maximová norma zadané soustavy činí (v souladu s maximální hodnotou modulu přenosu):

$\left\|\hat{G}_{\varphi}(s)\right\|_{\infty} = 2{,}2395 \; [-]$

Tvary váhových filtrů pro zadanou soustavu jsou následující, tedy:

váhový přenos pro referenci

$$\hat{W}_{cmd}(s) = \frac{1}{0{,}25 \cdot s + 1} \qquad (3-11b)$$

váhový přenos pro nízkofrekvenční poruchovou veličinu

$$\hat{W}_{d}(s) = \frac{0{,}5}{0{,}1 \cdot s + 1} \qquad (3-11c)$$

váhový přenos pro vysokofrekvenční poruchovou veličinu

$$\widehat{W}_{noise}(s) = \frac{0{,}01 \cdot s+1}{s+1} \qquad (3-11d)$$

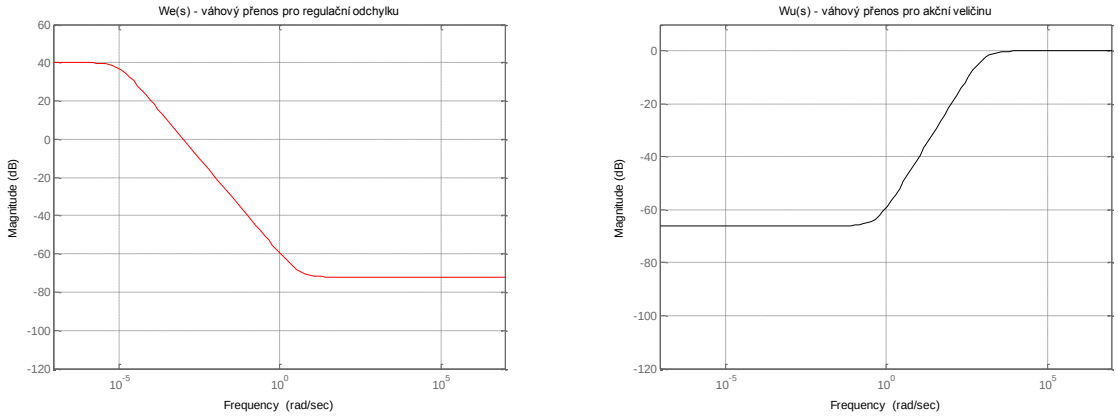
váhový přenos pro regulační odchylku

$$\widehat{W}_e(s) = K_e \cdot \frac{\frac{1}{M_e} \cdot s+\omega_{be}}{s+\omega_{be} \cdot \varepsilon_e} = 0{,}001 \cdot \frac{0{,}25 \cdot s+1}{s+1 \cdot 0{,}00001} = 0{,}001 \cdot \frac{0{,}25 \cdot s+1}{s+0{,}00001} \qquad (3-11d)$$

váhový přenos pro akční veličinu

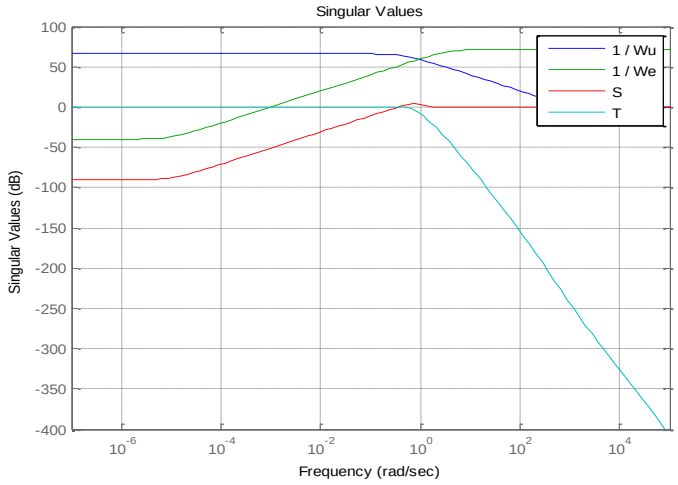
$$\widehat{W}_u(s) = K_u \cdot \frac{s+\frac{\omega_{bu}}{M_u}}{\varepsilon_u \cdot s+\omega_{bu}} = 10^{-7} \cdot \frac{s+\frac{1}{2}}{0{,}001 \cdot s+1} = 10^{-7} \cdot \frac{s+0{,}5}{0{,}001 \cdot s+1} \qquad (3-11e)$$

Váhové filtry pro regulační odchylku a akční veličinu mají předepsaný tvar přenosové funkce dle [14] – přenos obsahuje vždy stejný řád čitatele a jmenovatele, aby byla zajištěna také stabilita převráceného přenosu. Filtr pro regulační odchylku je dolnopropustný a pro akční veličinu hornopropustný.



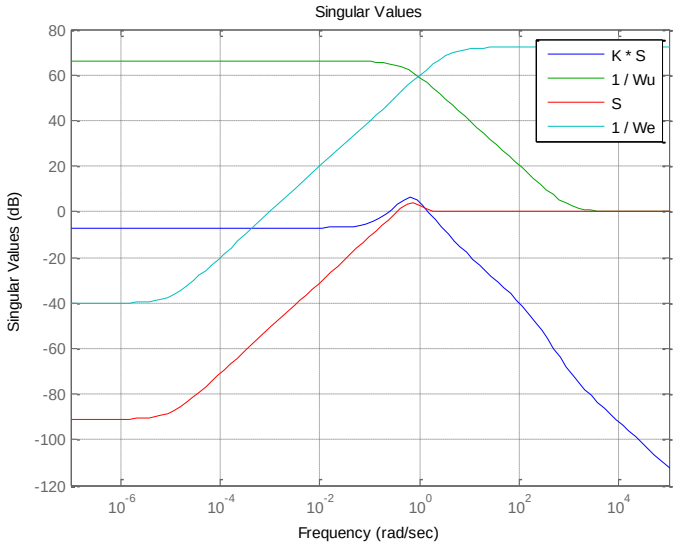
Obr. 3.3.5b: amplitudové frekvenční charakteristiky dolnopropustného váhového filtru $\widehat{W}_e(s)$ (vlevo) a hornopropustného váhového filtru $\widehat{W}_u(s)$(vpravo) – matematický model v azimutu

Grafické závislosti citlivostních funkcí a převrácených přenosů pro reálný model elevace jsou následující:



Obr. 3.3.5c: amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$, komplementární citlivostní funkce $\hat{T}(s)$, převrácených přenosů $1/\widehat{W}_e(s)$ a $1/\widehat{W}_u(s)$ – matematický model v azimutu

Z obr. 3.3.5c je zřejmé, že amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$ a převráceného přenosu $1/\widehat{W}_e(s)$ splňuje podmínku uvedenou v relaci (3 – 11f).

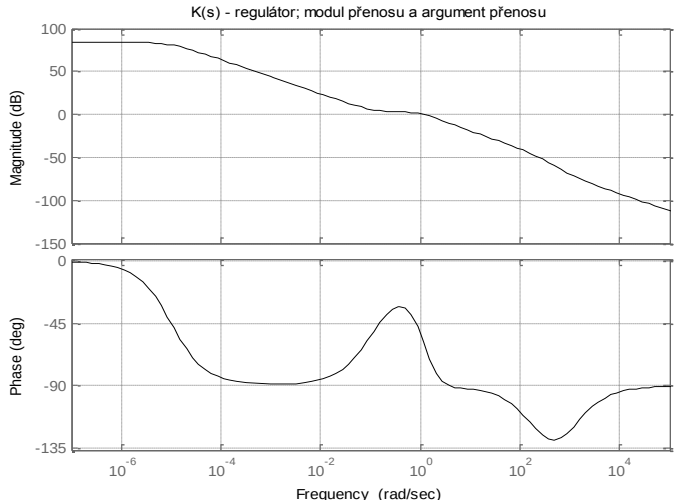


Obr. 3.3.5d: amplitudová frekvenční charakteristika citlivostní funkce $\hat{S}(s)$, součinu obrazů $\widehat{K}(s) \cdot \hat{S}(s)$, převrácených přenosů $1/\widehat{W}_e(s)$ a $1/\widehat{W}_u(s)$ – matematický model v elevaci

Z obr. 3.3.5d je zřejmé, že amplitudová frekvenční charakteristika součinu Laplaceových obrazů $\widehat{K}(s) \cdot \hat{S}(s)$ a převráceného přenosu $1/\widehat{W}_u(s)$ splňuje podmínku uvedenou v relaci (3 – 11g). Pro úplnost uveďme velikosti klíčových H∞ norem, tedy:

- **optimální H∞ norma:** $\gamma = 0{,}0049$
- **H∞ norma uzavřené smyčky:** $\left\| F\{\hat{P}(s), \widehat{K}(s)\} \right\|_\infty = 0{,}0041 < \gamma$
- **H∞ norma citlivostní funkce:** $\left\| \hat{S}(s) \right\|_\infty = 1{,}5636$
- **H∞ norma kompl. citlivostní funkce:** $\left\| \hat{T}(s) \right\|_\infty = 1{,}0003$

Z číselné interpretace H∞ norem je zřejmé, že se jedná o suboptimální řešení. Regulátor je, stejně jako v případě matematického modelu, nejvýše devátého řádu.

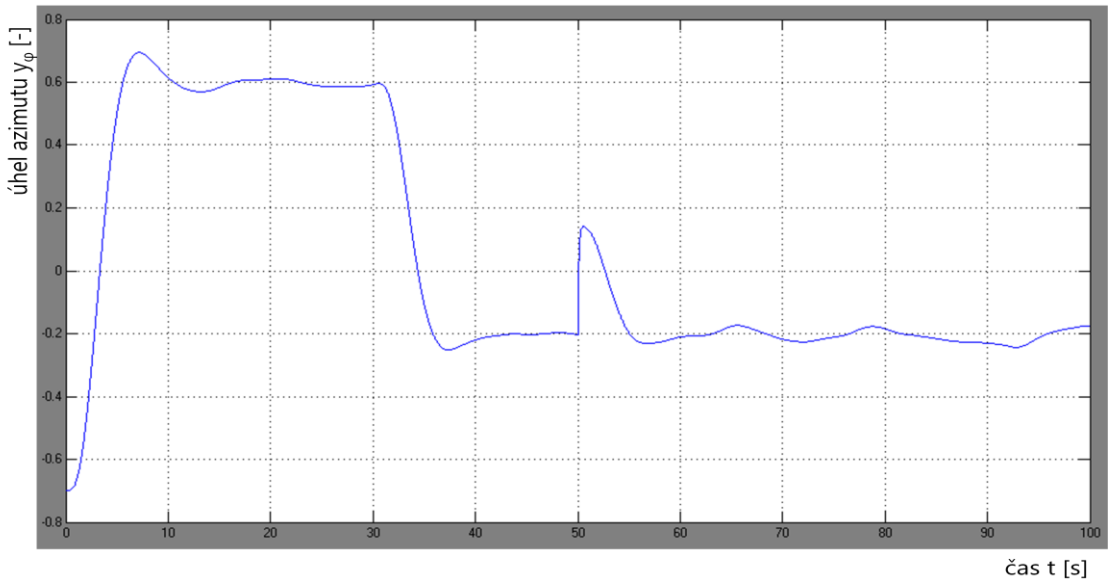


Obr. 3.3.5e: amplitudová frekvenční charakteristika a fázová frekvenční charakteristika H∞ regulátoru pro matematický model v azimutu

Podle obr. 3.3.2 byl v Simulinku vytvořen model H∞ regulátoru a rozšířené soustavy. Konfigurace skupiny externích vstupních signálů je následující:

- **reference:** $d_1(t=0) = r(t=0) = \ 0{,}60\ [-]$ pro $t \in (0;30\rangle$
  $d_1(t) = r(t=30) = -0{,}20\ [-]$ pro $t \in (30;100\rangle$
- **NF porucha:** $d_2(t) = d_2(t=50) = 0{,}70\ [-]$
- **VF porucha:** pásmově omezený bílý šum při výkonu $0{,}001\ [W]$

Odezva namodelovaného azimutového systému na referenci $d_1(t)$ a skok poruchy $d_2(t)$ je následující:



Obr. 3.3.5f: závislost výstupního úhlu elevace y_φ na čase t v matematickém modelu elevace; $y_\varphi = g(t)$

Díky vyváženému poměru citlivostní funkce a komplementární citlivostní funkce je eliminace šumu i poruchy velice účinná.

4 Filozofie .NET

4.1 Úvod do problematiky

Každých přibližně pět let musí být programátor ochoten vstřebat nové vědomosti, aby udržel krok se stále se měnícími novými technologiemi dnešní doby. Jazyky (C++, Visual Basic 6.0, Java), pracovní rámce (MFC, ATL, STL) a architektury (COM, COBRA, EJB) byly nakonec zastíněny něčím lepším, nebo přinejmenším něčím novým. V oboru softwarového inženýrství je současnou nabídkou společnosti Microsoft platforma .NET Framework. [15]

Důležitá témata, vztahujících se k platformě .NET, jsou dle [15]:

- sestavení neboli distribuční jednotka (Assembly)
- společný intermediální jazyk CIL (Common Intermediate Language)
- relace mezi různými aspekty .NET Frameworku (CLR, CTS, CLS)
- klíčové schopnosti programovacího jazyka C#

4.1.1 Problematika API C / Win32

V tradičním pojetí znamená vývoj softwaru pro rodinu operačních systémů Windows pracovat s programovacím jazykem C ve spolupráci s API (Application Programming Interface). I když je jistě pravda, že bylo tímto kdysi vysoce ceněným přístupem úspěšně vytvořeno mnoho aplikací, faktem zůstává, že pustit se do budování aplikace pomocí API v surovém stavu bývá komplikovaný a do jisté míry riskantní podnik. [15]

Prvním evidentní problém přináší komplikovanost samotného jazyka C, neboť vývojáři musí zápasit s ruční správou paměti, s aritmetikou ukazatelů a složitějšími syntaktickými konstrukcemi apod. Byť je C strukturovaný jazyk, postrádá objektový přístup. Když se zkombinují tisíce globálních funkcí a datových typů, definovaných v API Win32, do nějakého jazyka, nelze se pak divit, že existuje celá řada aplikací, v nichž se permanentně vyskytují nějaké chyby. [15]

4.1.2 Problematika C++ / MFC

Vývojářské práce se nesmírně zdokonalí, když se namísto surového (RAW formát) C a surového API použije programovací jazyk C++, na nějž lze v mnoha ohledech pohlížet jako na objektově orientovanou vrstvu, přikrývající C. Programátoři jazyka C++ sice mohou stavět na pilířích objektově orientovaného přístupu (zapouzdření, dědičnost a polymorfismus), ale stále se musí potýkat s již zmíněnými aspekty jazyka C, tj. ruční správa paměti, aritmetika ukazatelů a syntaktické konstrukce. [15]

Navzdory jeho složitosti existují v současné době pro jazyk C++ různé pracovní rámce – kupříkladu Microsoft Foundation Classes (MFC) poskytuje vývojářům sadu tříd C++, usnadňující vytváření aplikací pod Win32. Hlavní úloha MFC spočívá v zabalení nezpracovaného „zdravého jádra“ API Win32 do řady tříd, maker a četných nástrojů pro generování kódu. Avšak bez ohledu na to, jak prospěšnou asistenci tento rámec nabízí, stejně tak mnohé další soupravy nástrojů jazyka C++ pro prostředí založená na oknech, pořád platí, že programování v C++ je obtížné a hlavně náchylné k chybám. [15]

4.1.3 Problematika jazyka Visual Basic 6.0

Upřímná touha po prostším životním stylu přiměla mnohé programátory k tomu, že opustili svět pracovních rámců založených na C++ a přešli k přívětivějším jazykům typu Visual Basic 6.0 (VB6). Tento jazyk je populární především díky své schopnosti budovat složitá uživatelská rozhraní, knihovny kódu (COM servery) a logiku pro komfortní přístup k datům. VB6 skrývá z pohledu složitosti API Win32 v surovém stavu ještě více detailů než MFC, a to pomocí mnoha integrovaných průvodců kódu, interně zabudovaných datových typů, tříd a funkcí specifikovaných v jazyce Visual Basic. Hlavní nevýhodou (před příchodem jazyka Visual Basic .NET) bylo, že VB6 není plně objektově orientovaným jazykem; spíše bychom řekli, že „bere objekty na vědomí“. [15]

4.1.4 Problematika Java / J2EE

Programovací jazyk Java je téměř kompletně objektově orientovaný programovací jazyk se syntaktickými kořeny v jazyce C++ a podporou pro nezávislost na různých platformách. Java, jakožto jazyk, značně upravila mnohé nepříjemné syntaktické aspekty C++. Java, jakožto platforma, zas programátorům poskytuje nepřeberné množství předdefinovaných balíčků (Packages), obsahujících různé definice typů, pomocí nichž jsou programátoři jazyka Java schopni budovat čistě javovské aplikace s připojováním k databázím, podporou systémů zpráv, webovými stránkami a bohatě vybaveným uživatelským rozhraním. [15]

Byť Java představuje velmi elegantní jazyk, skýtá jednu potenciální potíž – musí být použita úplně všude kvůli limitované možnosti přístupu k nejavovskému API, čímž značně komplikuje jakoukoliv naději na integraci s jinými jazyky, neboť tento fakt ostře kontrastuje s primárním cílem jazyka Java, tj. užití jediného programovacího jazyka pro veškeré potřeby. V reálném světě totiž existují fragmenty kódů, které by se rády smísily s novějším javovským kódem, a vytvořily tak ideální program. S Javou se realizace této idey stává poněkud problematickou. [15]

4.1.5 Problematika COM

Objektový model komponent COM (Component Object Model) byl předchozím pracovním rámcem pro vývoj aplikací u společnosti Microsoft. COM je architektura říkající, že budete-li své třídy v souladu s pravidly COM, dostanete nakonec blok opětovně využitelného binárního kódu. [15]

Elegance binárního serveru COM spočívá v tom, že se k němu dá přistupovat nezávisle vzhledem k použitému programovacímu jazyku. Programátoři C++ mohou vytvořit třídy COM, následně využitelné v VB6; programátoři Delphi mohou využít třídy COM, vybudované v jazyce C atd. Ovšem i u jazykové nezávislosti COM se uplatňuje limitace, tedy neexistuje žádný způsob, jak odvodit novou třídu COM pomocí již existující třídy COM, protože jednoduše není podpora klasického dědění. Pro opětovné využití typů tříd COM se tudíž musí využít těžkopádnější relace typu „má“. [15]

Jiná přednost COM spočívá v transparentním umístění; pomocí identifikátorů (AppID), kostry (Stub), proxy či běhového prostředí COM se programátoři mohou vyhnout práci s nezpracovanými sokety, vzdálenými voláními procedur RPC a jinými nízkoúrovňovými detaily. [15]

Přestože lze COM prohlásit za velice úspěšný model, vnitřně je extrémně složitý. Aby se zjednodušil vývoj binárních jednotek COM, vzniklo mnoho pracovních rámců, kupříkladu rámec ATL (Active Template Library) poskytuje sadu tříd C++, šablon a maker, usnadňujících vytváření typů COM. [15]

4.2 Platforma .NET

Z výše uvedené stručné historie vývoje je zřejmé, že programátoři mají v tomto ohledu nesnadnou práci, proto .NET Framework s sebou přináší dosti radikální přístup pomocí hrubé síly, kdy se snaží programátorům jejich práci co nejvíce ulehčit. Nabízené řešení problematiky znamená vše od základu změnit. Platforma .NET Framework umožňuje nejen budování aplikací na rodině Windows, ale také na jiných operačních systémech – Apple Mac OS X a distribuce Unix / Linux. [15]

Platforma .NET – také .NET Framework, prostředí .NET nebo jen .NET – byla odborné veřejnosti představena poprvé v roce 2001; o rok později pak firma Microsoft uvolnila její první verzi společně s vývojovým prostředím Visual Studio 2002. Dosud byly uvolněny verze dle [15]:

- **.NET Framework 1.1** *2004*
- **.NET Framework 2.0** *podzim 2005*
- **.NET Framework 3.0** *podzim 2006*
- **.NET Framework 3.5** *podzim 2007*

Klíčové schopnosti této platformy jsou následující, tedy dle [15]:

- **úplná interoperabilita s existujícím kódem:**
  Existující binární jednotky COM se mohou kombinovat s novějšími binárními jednotkami platformy .NET a naopak. Navíc PInvoke (Platform Invocation Services) umožňuje volat z kódu .NET knihovny založené na jazyce C, a to včetně podkladového API operačního systému.
- **kompletní a totální integrace jazyků:**
  Na rozdíl od COM platforma .NET podporuje dědění přes jazyky, zpracování výjimek přes jazyky a také ladění přes jazyky.
- **engine společného runtime režimu, sdíleného všemi jazyky platformy .NET:**
  Jedná se o dobře definovanou sadu typů, jimž jazyky platformy .NET rozumí.
- **knihovna základních tříd:**
  Poskytuje úkryt před složitostmi volání API v surovém stavu a nabízí jednotný objektový model využitelný pro všechny jazyky platformy .NET.
- **eliminace extrémní složitosti COM**
- **zjednodušený rozmisťovací model:**
  Není třeba registrovat binární jednotku v registru operačního systému. Platforma .NET navíc povoluje, aby na jediném stroji existovalo i více verzí jedné dynamicky linkované knihovny DLL. Tento problém byl programátory často označován termínem „peklo dynamických knihoven" (DLL Hell). Instalace nové verze DLL, a tudíž také přepsání starší verze, často znamenalo, že nová softwarová aplikace fungovala, ale přestaly fungovat starší aplikace využívající verzi některé z knihoven.
- **podpora webových aplikací:**
  Vývoj aplikací pro web se stává jednou z priorit současnosti, proto je třeba poskytnout vývojářům odpovídající nástroje.
- **systematická podpora nástrojů pro lokalizaci a internacionalizaci programů:**

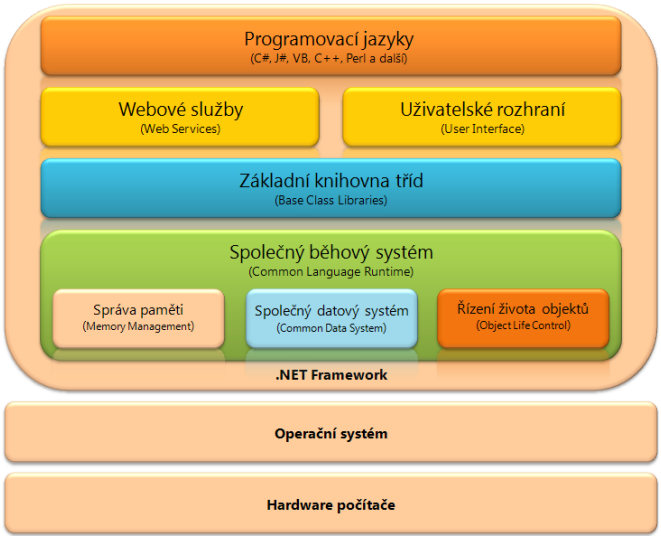
Starší programovací nástroje a jazyky tuto problematiku téměř neřešily. Prvním široce rozšířeným jazykem se snahou řešit problém komplexně byla Java (1995). Platforma .NET podporuje vytváření jazykových mutací softwaru, používání informací o lokálních kulturních zvyklostech, např. pořadí znaků při abecedním řazení, psaní desetinných čárek či teček, používání lokálně obvyklého formátu data, času, měny apod.

- **možnost práce s datovými typy a distribučními jednotkami, neznámými v době překladu:**
  Součástí knihoven platformy .NET je také kvalitní podpora mechanismu reflexe.
- **zabezpečení softwaru:**
  Mechanismus digitálního podpisu distribučních jednotek, založený na veřejném klíči (Public Key) a soukromém klíči (Private Key) poskytuje elegantní řešení.

Je patrné, že platforma .NET nemá s COM vůbec nic společného kromě firmy Microsoft coby výrobce, tudíž jediným způsobem komunikace .NET a COM je vrstva interoperability. [15]

4.2.1 Struktura platformy .NET

Blokové schéma struktury platformy .NET Framework je následující, tedy dle [15]:



Obr. 4.2.1: blokové schéma základní struktury platformy .NET Framework

Na hardwaru počítače leží jako základní vrstva operační systém, nad nímž stojí platforma .NET, skládající se z několika dílčích vrstev. Základní vrstvu tvoří společný běhový systém CLR (Common Language Runtime), definující správu paměti (Memory Management), společný datový systém (Common Data System) a řízení života objektů (Object Life Control). Nad základní vrstvou CLR stojí základní knihovna tříd BCL (Base Class Libraries), obsahující dle [15]:

- řadu nástrojů od atomických datových typů až po třídy představující datové kontejnery neboli kolekce
- nástroje pro vstupní a výstupní operace
- nástroje pro práci s regulárními výrazy
- nástroje pro práci s více vlákny
- nástroje pro zpracování souborů v jazyce XML / SQAP

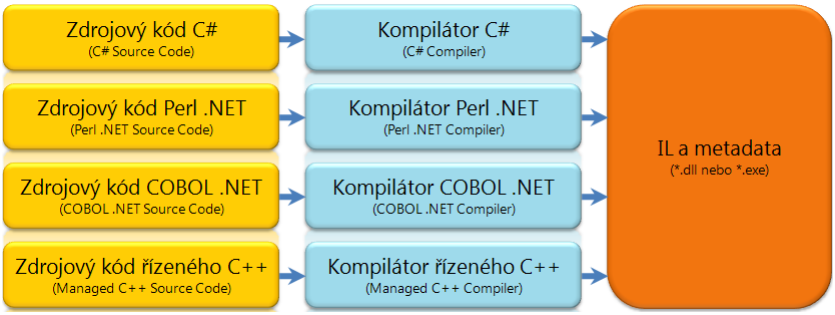
- nástroje pro práci s formuláři Windows
- nástroje pro práci se sítí a webovými formuláři
- nástroje pro zpracování lokálního nastavení
- a další

Nad základní knihovnou tříd stojí knihovny pro podporu grafického rozhraní (User Interface), webových služeb (Web Services) nebo přístupu k databázím apod. [15]

Nejvyšší vrstvu tvoří překladače programovacích jazyků platformy .NET, překládající zdrojový kód do jazyka IL (Intermediate Language). Standardní součástí platformy .NET jsou vedle překladače jazyka C# také překladače jazyka J#, jazyka Visual Basic a jazyka IL. Rovněž zde existují nástroje třetích stran, např. Borland pro jazyk Object Pascal nástroje Delphi a mnohé další. [15]

4.2.2 Assembly .NET – binární (distribuční) jednotky

Bez ohledu na zvolený programovací jazyk platformy .NET je nutno pochopit, že byť distribuční jednotky .NET mají stejnou příponu jako servery COM a neřízené binární jednotky Win32, nevykazují absolutně žádné vnitřní podobnosti. Jednotky .NET se rovněž nepopisují pomocí typových knihoven COM a neregistrují se v systémovém registru; nejdůležitější je však skutečnost, že jednotky .NET neobsahují instrukce specifické pro platformy, nýbrž intermediární jazyk IL a metadata typů. V operačním systému Windows má jednotka formát PE (Portable Executable). [15]



Obr. 4.2.2: blokové schéma vytvoření distribuční jednotky (instrukce IL a metadata) pomocí jazyků platformy .NET

Jednotka obsahuje kód IL, který je koncepčně podobný bajtovému javovskému kódu v tom smyslu, že se absolutně nezbytně nekompiluje do instrukcí specifických pro danou platformu. Jedná se o referenční místo, v němž nějaký blok instrukcí CIL má používat runtime .NET. [15]

Kromě instrukcí CIL obsahují jednotky také metadata, popisující charakteristiky každého typu, jenž se v jednotce nachází. Oproti metadatům COM představují metadata .NET značné zdokonalení. Potíže s informacemi o typech COM spočívají v tom, že není garantována jejich přítomnost a že kód IDL (Interface Definition Language) nemá k dispozici žádný způsob dokumentace externě odkazovaných serverů, nezbytných pro správnou funkci aktuálního COM serveru. Naproti tomu metadata .NET jsou k dispozici vždy a automaticky je generuje konkrétní kompilátor jazyka platformy .NET. [15]

Pomocí metadat však lze také popsat samotnou binární jednotku, čemuž se říká manifest, obsahující informace o aktuální verzi jednotky, informace o kultuře (lokalizace řetězcových a obrázkových zdrojů) a seznam všech externě odkazovaných binárních jednotek. [15]

Jednosouborová binární jednotka

V drtivé většině případů existuje jednoduchá korespondence typu jeden k jednomu, a to mezi jednotkou .NET a binárním souborem (*.dll nebo *.exe), tedy binární soubor a jednotka jsou v tomto případě identické. Jinými slovy skládá-li se jednotka z jediného modulu (*.dll nebo *.exe), mluvíme o jednosouborové binární jednotce, obsahující veškeré nutné instrukce CIL, metadata i manifest v autonomním a unikátním, dobře definovaném balíčku. [15]

Vícesouborová binární jednotka

Vícesouborová jednotka se skládá z četných binárních jednotek; každé z nich se nazývá modul. Takovýto druh jednotky musí obsahovat primární modul obsahující manifest; ostatní přidružené moduly obsahují manifest úrovně modulu, CIL a metadata typů. Primární modul uvnitř manifestu jednotky dokumentuje množinu požadovaných sekundárních modulů. [15]

Rozčleníme-li jednotku do separátních modulů, máme k dispozici flexibilnější možnosti při rozmísťování. Cílem je vytvořit binární jednotku s opravdu logickou strukturou jednoho nebo několika provázaných modulů a možností lokace jednotky jako celku s unikátním číslem verze. [15]

4.2.3 Základní stavební kameny platformy .NET

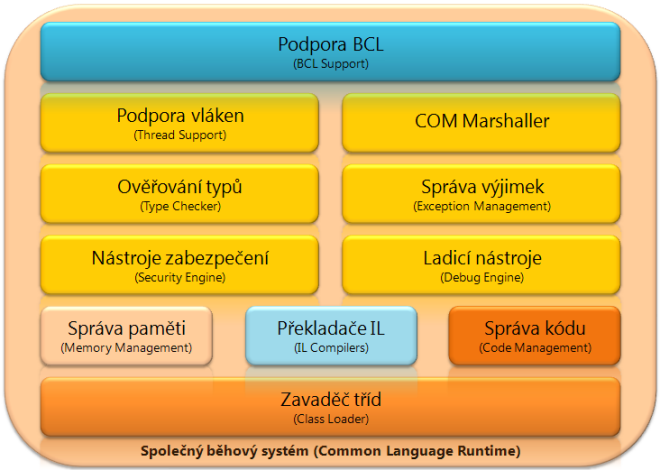
Nyní když už víme o výhodách platformy .NET, prozkoumejme tři klíčové a vzájemně provázané entity dle [15]:

- CLR
- CTS
- CLS

Z hlediska programátora lze .NET chápat jako běhové prostředí (Runtime) nové generace a vyčerpávající knihovna základních tříd. Na runtime vrstvu se má správně odkazovat jako na společný běh jazyků neboli společný běhový systém CLR, jehož primární rolí je, aby automaticky vyhledával, načítal, spravoval typy platformy .NET a také se staral o řadu nízkoúrovňových detailů, tj. správa paměti a provádění bezpečnostních kontrol. [15]

CLR – společný běhový systém

Blokové schéma společného běhového systému platformy .NET vypadá následovně dle [15]:



Obr. 4.2.3: základní struktura společného běhové systému CLR platformy .NET

Platforma .NET je čistě objektově řešena, čemuž odpovídají i programy v jazyce IL, proto je základním elementem CLR zavaděč tříd (Class Loader), který vyhledá distribuční jednotku (Assembly), v ní požadovanou třídu, zkontroluje správnost verze a případného digitálního podpisu. Následně se třída zavede do operační paměti. [15]

Překladač IL má za úkol konvertovat kód v jazyce IL do nativního kódu stroje. Automatická správa paměti se stará o odklízení objektů, na něž již neexistují platné reference (odkazy). Programátor tak alokuje nové objekty, ale o jejich odstranění se už nestará; ve skutečnosti k tomu nemá ani potřebné nástroje. [15]

V rámci CLR hovoříme o problematice řízeného a neřízeného kódu. V případě řízeného kódu (Managed Code) CLR hlídá a poskytuje rozsáhlejší podporu, než tomu bylo u klasických aplikací pro Win32. Většina programovacích jazyků platformy .NET umožňuje vyvíjet pouze řízený kód; jazyk C# však poskytuje rovněž nástroje pro vývoj neřízeného kódu, někdy označovaného jako nebezpečný či nezabezpečený (Unsafe Code), v němž lze používat ukazatele a jejich aritmetiku jako v jazyce C. Ovšem používání takovéhoto kódu v jazyce C# je spíše výjimečné a typicky omezeno na nízkoúrovňový přístup. Připomeňme, že nezabezpečené jsou jenom jednotlivé bloky nebo metody, nikoliv celý program. [15]

CTS – společný systém typů

Dalším stavebním blokem platformy .NET je společný systém typů CTS (Common Type System). Tato specifikace plně popisuje všechny možné druhy datových typů a programovacích konstrukcí, podporovaných runtimem. Má za úkol specifikovat, jakým způsobem mohou mezi sebou entity komunikovat nebo reprezentovat ve formátech metadat platformy .NET, např. dle [15]:

- třídy smějí mít jediného předka
- struktury nesmějí mít konstruktor bez parametrů
- struktury musí být odvozeny od typu ***System.ValueType***
- výčtové typy neboli enumerace musí být odvozeny od typu ***System.Enum***

Daná binární jednotka .NET může obsahovat libovolný počet odlišných typů. Ve světě platformy .NET se slovem typ označuje všeobecný termín, jímž se odkazuje na členy množiny, tedy dle [15]:

- třída (Class)
- struktura (Structure)
- rozhraní (Interface)
- výčet (Enumeration)
- delegát (Delegate)

Pro programátory jazyka C# nemá tato specifikace žádný speciální význam, neboť tento jazyk jí samozřejmě vyhovuje; v případě užití jiného jazyka lze narazit na jistá omezení v rámci CTS, která však nemusí být na první pohled zcela zřejmá. Opravdu detailněji se vnitřním fungováním CTS zabývají programátoři, budující nástroje a kompilátory zacílené na platformu .NET. [15]

Typ třída CTS

Každý jazyk platformy .NET minimálně podporuje typ třída (Class Type), základní stavební kámen objektově orientovaného programování OOP (Object-Oriented Programming). [15]

Třída může obsahovat libovolný počet členů dle [15]:

- vlastnosti (Properties)

- metody (Methods)
- události (Events)

Rovněž může obsahovat datové členské proměnné:

- pole (Array)
- členské proměnné (Member Variables)

Charakteristika třídy	Význam
Zapečetěná nebo nezapečetěná třída	Zapečetěná třída (Sealed Class) nemůže fungovat jako základní třída (Base Class) jiných tříd.
Implementace rozhraní	Rozhraní je kolekce abstraktních členů, která slouží jako spojení mezi objektem a jeho uživatelem. CTS umožňuje třídám implementovat libovolný počet rozhraní.
Abstraktní nebo konkrétní třída	Abstraktní třídy nelze vytvářet přímo; jejich úkol spočívá v definici společného chování pro odvozené typy. Konkrétní třídy lze vytvářet přímo.
Viditelnost třídy	Každá třída musí obsahovat informaci o své viditelnosti pomocí modifikátoru. Tato charakteristika definuje, zda třídu mohou používat externí jednotky, nebo definující jednotka.

Tab. 4.2.1: charakteristika třídy společného systému typů CTS

Typ struktura CTS

Na strukturu lze nahlížet jako na lehkotonážní typ třída se sémantikou založenou na hodnotách. Struktury se obvykle nejlépe hodí pro modelování geometrických a matematických dat. [15]

Typ rozhraní CTS

Rozhraní je definována jako pojmenovaná kolekce definic abstraktních členů, jež může implementovat daná třída nebo struktura. Na rozdíl od COM se rozhraní platformy .NET neodvozují ze společného základního rozhraní; v jazyce C# se typy rozhraní definují klíčovým slovem interface. [15]

Sama o sobě jsou rozhraní téměř k ničemu. Ovšem když daná třída nebo struktura implementuje svým unikátním způsobem dané rozhraní, pak se k němu přistupuje pomocí reference na rozhraní s využitím polymorfismu. [15]

Typ výčet CTS

Výčty neboli enumerace jsou programovací konstrukce, umožňující seskupit dohromady dvojice typu název / hodnota. Standardně se každý prvek výčtu uchovává jako 32bitové číslo. CTS také požaduje, aby se výčtové typy odvozovaly ze společné třídy Systém.Enum. [15]

Typ delegát CTS

Delegáti jsou ekvivalenty platformy .NET pro typově bezpečný funkční ukazatel podobně jako v jazyce C. Klíčový rozdíl spočívá v tom, že delegát platformy .NET je třída odvozená ze třídy Systém.MulticastDelegate, a tudíž se nejedná o prostý ukazatel ukazující na nějakou adresu v paměti. [15]

CLS – společná specifikace jazyků

Je třeba si uvědomit, že daný jazyk platformy .NET nemusí podporovat úplně všechny schopnosti dle CTS. Příbuznou specifikací je společná specifikace jazyků CLS (Common Language Specification), definující podmnožinu, respektive minimum společných typů a programovacích konstrukcí pro jazyky .NET, i když je mohou vyjadřovat různými způsoby. [15]

Smysl této specifikace spočívá v definování pravidel zajišťující, že třídu napsanou v jednom programovacím jazyce platformy .NET lze použít (vytvořit instance, volání metod a odvození potomků) v kterémkoliv jazyce .NET. [15]

Různé jazyky vyjadřují stejné programovací konstrukce pomocí unikátních termínů, ovšem jazyky se mohou lišit co do své funkcionality, např. některý jazyk platformy .NET může, ale nemusí mít klíčové slovo pro reprezentaci dat bez znaménka a může, ale nemusí podporovat typy ukazatelů (Pointers). Uvážíme-li, kolik existuje možných variant, bylo by ideální, kdyby existovala základna, na které by se shodly všechny jazyky platformy .NET. [15]

V platformě .NET takovouto základnu představuje společná specifikace jazyků CLS – sada pravidel, popisujících do nejmenších detailů minimální a kompletní sadu schopností, kterou musí podporovat každý kompilátor jazyka platformy .NET, aby samozřejmě produkoval takový kód, jehož hostitelem může být společný běhový systém CLR, a aby se k nim dalo v jednotném duchu přistupovat ze všech jazyků platformy .NET. [15]

CLS ztělesňuje hlavní sadu pravidel, která musí striktně dodržovat tvůrci překladačů, chtějí-li požadovanou funkcionalitu svých produktů ve světě .NET. Každé pravidlo má přiřazený jednoduchý název, např. CLS Rule 6, a popisuje, jak toto pravidlo ovlivňuje výrobce překladačů a ty, kdo s nimi komunikují. Mocné pravidlo 1 (Rule 1) zní dle [15]:

- ***Pravidla CLS se aplikují pouze na ty části typu, které jsou vystaveny vně assembly neboli binární jednotky.***

Z tohoto pravidla se dá vydedukovat, že zbývající pravidla CLS se netýkají interní logiky. Jediné aspekty typu, které musejí respektovat pravidla CLS, bývají samotné definice členů, tj. jmenné konvence, parametry a návratové typy. Implementační logika pro daný člen může použít jakékoliv množství technik nekorespondujících s CLS, neboť vnější svět stejně žádný rozdíl nepozná. [15]

4.2.4 Jmenné prostory

Knihovny platformy .NET jsou hierarchicky uspořádány do tzv. prostorů jmen či jmenných prostorů (Namespace). Jméno třídy z knihovny je třeba kvalifikovat tímto jménem, k čemuž se používá tečková konvence (Dot Convention), tak charakteristická pro platformu .NET. Bázový prostor se jmenuje Systém a všechny ostatní standardní jmenné prostory jsou do něj vnořeny. Platforma .NET těmito prostory udržuje solidní uspořádání všech typů z knihoven základních tříd BCL. [15]

Jmenný prostor seskupuje typy uložené v binární jednotce a mající mezi sebou určitou relaci. Je vhodné zmínit, že jediná binární jednotka, např. dynamická knihovna ***mscorlib.dll***, může obsahovat libovolný počet jmenných prostorům, přičemž každý z nich může obsahovat libovolný počet typů. [15]

Jmenný prostor platformy .NET	Význam
System	Obsahuje užitečná typy, slouží pro práci s interními daty, pro matematické operace, generování náhodných čísel a pro tzv. odvoz odpadků (Garbage Collector). Jsou zde také četné typy běžně používaných výjimek a atributů.
System.Collections **System.Collections.Generic**	Definují řadu skladových kontejnerových objektů (ArrayList, Queue apod.) a také základní typy a rozhraní s možností vytváření vlastních kolekcí. Od verze .NET 2.0 jsou typy kolekcí rozšířené o generické schopnosti.
System.Drawing **System.Drawing.Drawing2D** **System.Drawing.Printing**	Obsahuje typy, které obalují primitivní grafické prvky, tj. bitmapy, písma a ikony, a také možnosti tisku.
System.IO **System.IO.Compression** **System.IO.Ports**	Obsahují vstupní a výstupní operace se soubory. Od verze .NET 2.0 zahrnují jmenné prostory IO pro komprimaci a manipulaci s porty.
System.Reflection **System.Reflection.Emit**	Obsahují typy podporující odhalování typů při běhu a dynamické vytváření typů.
System.Runtime.InteropServices	Poskytuje schopnosti umožňující typům platformy .NET komunikaci s neřízeným kódem, tj. *.dll založená na jazyce C a servery COM, a naopak.
System.Threading	Definuje typy, s nimiž lze budovat aplikace s více vlákny.
System.Web	Obsahuje jmenné prostory zaměřené na vývoj webových aplikací platformy .NET, a to včetně webových služeb ASP.NET a XML.
System.Windows.Forms	Obsahuje typy, usnadňující budování klasických desktopových aplikací s grafickým uživatelským rozhraním operačního systému Windows.
System.Xml	Obsahuje četné typy pro interakci s daty formátu XML.

Tab. 4.2.2: charakteristika významných jmenných prostorů platformy .NET

4.2.5 Datové typy platformy .NET

Platforma .NET Framework nabízí programátorům širokou paletu datových typů, definovaných BCL. Datové typy jsou přitom architektonicky navrženy tak, aby je bylo možné použít z libovolného jazyka platformy .NET. CTS zavádí definice a členění datových typů, zatímco CLS má na starosti kontrolu implementace jazykově-syntaktických požadavků, kladených na kompilátory jazyků .NET platformy. Podle společného typového systému lze datové typy dělit na dvě základní kategorie, tedy dle [15]:

- hodnotové typy
- odkazové (referenční) typy

Obě skupiny typů mají však také své přímé protějšky v systémové vrstvě platformy .NET – systémové datové typy.

Datové typy lze dělit dle [15] na:

- typy vestavěné
- typy uživatelsky definované

Kritériem pro zařazení určitého typu do jedné z výše deklarovaných skupin je známost tohoto typu z pohledu kompilátoru. S vestavěnými typy si kompilátor rozumí. Jiná situace ovšem nastane v případě uživatelsky definovaných typů, deklarovaných uživatelem. Tento typ není možno použít před jeho definicí, protože překladač by o tomto typu vůbec nic nevěděl, a tudíž by nemohl alokovat dostatečný jazykový prostor pro instanci takového typu. [15]

Hodnotové datové typy

V rámci kategorie vestavěných hodnotových typů také hovoříme ještě o speciální typové podmnožině, tvořící atomické (primitivní) hodnotové datové typy, které slouží pro práci s daty (celá čísla, čísla v pohyblivé řádové čárce, textové znaky, pravdivostní hodnoty). Dle charakteru zpracovávaných dat lze vestavěné hodnotové datové typy zařadit do několika homogenních skupin, tedy dle [15]:

- **datové typy pro práci s celočíselnými hodnotami v pevné řádové čárce:**
  Datové typy tohoto druhu se také označují jako integrální datové typy. Zároveň se jedná o nejpočetnější skupinu, neboť zahrnuje nejen znaménkové, ale také bezznaménkové typy.

Název hodnotového typu	.NET	C#	VB .NET	Řízená rozšíření pro C++	CLS kompatibilita
byte	System.Sbyte	byte	Byte	unsigned char	ano
sbyte	System.Byte	sbyte	Sbyte	signed char	ne
short	System.Int16	short	Short	short	ano
ushort	System.UInt16	ushort	Ushort	unsigned short	ne
int	System.Int32	int	Integer	int / long	ano
uint	System.UInt32	uint	UInteger	unsigned int / unsigned long	ne
long	System.Int64	long	Long	_int64	ano
ulong	System.UInt64	ulong	ULong	unsigned _int64	ne

Tab. 4.2.3a: přehled hodnotových datových typů pro práci s celočíselnými hodnotami v pevné řádové čárce u vybraných jazyků platformy .NET

- **datové typy pro práci s číselnými hodnotami v pohyblivé řádové čárce:**
  Pokud bychom hledali napříč všemi programovacími jazyky, pravděpodobně bychom jenom stěží nalezli zástupce, který by neobsahoval implementaci datových typů pro realizaci operací s desetinnými čísly.

Název hodnotového typu	.NET	C#	VB .NET	Řízená rozšíření pro C++	CLS kompatibilita
float	System.Single	float	Single	float	ano
double	System.Double	double	Double	double	ano
decimal	System.Decimal	decimal	Decimal	decimal	ano

Tab. 4.2.3b: přehled hodnotových datových typů pro práci s číselnými hodnotami v pohyblivé řádové čárce u vybraných jazyků platformy .NET

Výše uvedené datové typy plně vyhovují pravidlům CLS, a tudíž je lze bez obav používat i při vývoji komponent, ovládacích prvků či knihoven tříd. [15]

- **ostatní vestavěné datové typy:**

Název hodnotového typu	.NET	C#	VB .NET	Řízená rozšíření pro C++	CLS kompatibilita
char	System.Char	char	Char	wchar_t	ano
bool	System.Boolean	bool	Boolean	Bool	ano

Tab. 4.2.3c: přehled hodnotových vestavěných hodnotových datových typů u vybraných jazyků platformy .NET

Byly uvedeny všechny hodnotové vestavěné typy, s nimiž vývojáři .NET platformy mohou pracovat. Kromě jednoho hodnotového datového typu, a to typu decimal, jsou všechny ostatní hodnotové vestavěné typy atomické, tedy primitivní. Nutno ještě doplnit, že ve sloupci jazyka C++ je uvedena reprezentace příslušného datového typu v rámci Visual C++ .NET s řízenými rozšířeními ME (Managed Extensions), což znamená, že nemusí odpovídat standardním typům v jazyce C++ dle ISO 14882-2003. [15]

Odkazové datové typy

Do odkazových neboli referenčních datových typů patří dle [15]:

- **vestavěné odkazové typy:**

Název odkazového typu	.NET	C#	VB .NET	Řízená rozšíření pro C++	CLS kompatibilita
string	System.String	string	String	String^	ano
object	System.Object	object	Object	Object^	ano

Tab. 4.2.4: přehled dostupných vestavěných odkazových datových typů u vybraných jazyků platformy .NET

- **uživatelsky definované odkazové typy:**
  Zde patří třídy, pole, kolekce, rozhraní a delegáti.
- **speciální odkazové typy:**
  Zde patří ukazatele.

Bázová knihovna tříd BCL platformy .NET definuje vestavěné odkazové typy string a object, které ovšem nepatří mezi primitivní. Instance typu string lze použít pro práci s textovými řetězci neboli kolekcemi textových znaků sady Unicode. Každý textový řetězec reprezentuje objekt, sdružující jednotlivé textové znaky, z nichž se řetězec skládá. [15]

Zcela výsostné postavení zastává odkazový datový typ object, jehož odkazová proměnná může uchovávat referenci na jakákoliv platná data jiných hodnotových i odkazových datových typů. Tato data jsou vždy zapouzdřena v objektu, tedy instalaci odkazového typu. Pokud do odkazové proměnné typu object vložíme hodnotový datový typ, implicitně se spustí mechanismus sjednocení typů, jehož výsledkem bude vygenerování instance třídy ***System.ValueType*** a její umístění na řízené hromadě (Managed Heap). [15]

4.2.6 Nezávislost na .NET platformě

K velkému překvapení vývojářů se binární jednotka (Assembly) platformy .NET mohou vyvíjet a vykonávat na operačních systémech, nepocházejících od společnosti Microsoft, např. Mac OS X, různé distribuce Linuxu, dále pak BeOS nebo FreeBSD apod. [15]

Když společnost Microsoft uvolnila platformu .NET Framework a programovací jazyk C#, vytvořila také sadu formálních dokumentů, popisujících syntaxi i sémantiku jazyků CIL a C#, formát binární jednotky .NET (Assembly .NET), jádro jmenných prostorů platformy .NET a mechaniku hypotetických enginů runtime .NET neboli virtuální vykonávací systém VES (Virtual Execution System). Určitě stojí za zmínku, že tyto dokumenty byly odeslány do organizace Ecma International a schváleny coby oficiální mezinárodní standardy pro tyto specifikace dle [15]:

- **ECMA-334:** The C# Language Specification *specifikace jazyka C#*
- **ECMA-335:** The Common Language Infrastructure *společná jazyková infrastruktura*

Ecma International je nezisková organizace vývojářů technologií, dodavatelů a uživatelů. Ecma International předkládá výsledky své práce ke schválení jako normy ISO, IEC, ISO/IEC a ETSI a je hlavním tvůrcem zrychlených specifikací v rámci přípravy standardu. [15]

Specifikace ECMA-335 je z obou specifikací tou robustnější, a tudíž rozdělenou do pěti sekcí, tedy dle [15]:

Sekce specifikace ECMA-335	Význam
Část I: Architektura	popis všeobecné architektury CLI, pravidel CTS a CLS a také mechaniku engine runtimu .NET
Část II: Metadata	popis detailů metadata platformy .NET
Část III: CIL	popis syntaxe a sémantiky kódu jazyka CIL
Část IV: Knihovny	vysokoúrovňový přehled o minimálních a kompletních knihovnách tříd, které musí podporovat distribuce .NET
Část V: Přílohy	kolekce maličkostí – vodítka pro design knihovny tříd a implementační detaily kompilátoru CIL

Tab. 4.2.5: význam jednotlivých sekcí specifikace ECMA-335

Je třeba si uvědomit, že čtvrtá sekce Knihovny definuje pouze minimální množinu jmenných prostorů, reprezentujících jádro služeb očekávaných od distribuce CLI obsahující dle [15]:

- kolekce
- vstupní a výstupní operace na konzole
- vstupní a výstupní operace se soubory
- práce s vlákny
- reflexe
- síťový přístup
- jádro bezpečnostních služeb
- manipulace s XML

CLI naopak nedefinuje jmenné prostory, usnadňující dle [15]:

- webový vývoj (ASP.NET)

- přístup k databázím (ADO.NET)
- vývoj desktopových aplikací s grafickým uživatelským rozhraním (Windows Forms, WinForms)

4.3 Jazyk C#

Vzhledem k tomu, že platforma .NET představuje radikální odklon od předcházejících technologií, vyvinula specificky pro ni firma Microsoft nový programovací jazyk – jazyk C#, používající podobnou syntaxi jako Java, přesto nelze C# považovat za pouhý javovský odvar. Oba jazyky pocházejí ze stejné rodiny, a tudíž vycházejí ze společného základu – syntaktických konstrukcí jazyka C++; lze také říci, že jazyk C# představuje vyčištěnou verzi Javy. [15]

Mnohé syntaktické konstrukce jazyka C# jsou vymodelované dle různých aspektů jazyků Visual Basic 6.0 a C++, například – podobně jako VB6 – podporuje i jazyk C# představu formálních vlastností typů a schopnost deklarovat metody s vyšším počtem přebíraných argumentů (pole argumentů); v případě podobnosti s jazykem C++ povoluje jazyk C# přetěžování operátorů, vytváření struktur, výčtů a funkce zpětného volání (Callback). [15]

Byť jazyk C# představuje hybrid mnoha programovacích jazyků, výsledkem je syntakticky tak čistý produkt jako Java, jednoduchý jako VB6, efektivní a flexibilní jako C++. Jazyk C# zkrátka nabízí zajímavé schopnosti, přičemž mnohé z nich sdílejí i další jazyky platformy .NET, tedy dle [15]:

- jazyk C# nevyužívá přímou manipulaci s ukazateli, přesto lze s nimi pracovat při nízkoúrovňovém přístupu, je-li to nezbytně nutné
- automatická správa paměti prostřednictvím odvozu odpadu (Garbage Collection) zajišťuje, že jazyk C# nepodporuje klíčové slovo `delete`
- schopnost jednoduchého přetěžování operátorů pro vlastní datový typ
- od C# 2005 lze budovat také generické typy a generické členy pomocí syntaxe, velmi podobné šablonám jazyka C++
- úplná podpora programovacích technik, založených na rozhraních
- úplná podpora technik aspektově orientovaného programování AOP (Aspect-Oriented Programming) přes atributy, umožňující přiřazovat charakteristiky typům a jejich členů

Asi nejdůležitější věcí, kterou je nutné si uvědomit v souvislosti s jazykem C#, je to, že umí produkovat pouze kód, vykonávaný pouze uvnitř runtime .NET, proto nelze pomocí jazyka C# vytvořit domácí server COM nebo neřízenou aplikaci API Win32. [15]

4.4 Formuláře Windows

Vývoj aplikací Windows se podstatně změnil od časů, kdy byl v roce 1983 vydán operační systém Windows 1.0. Způsob, jakým programátoři Windows píší software, a to včetně jeho architektury se dramaticky změnily. Tuto změnu přinesla právě platforma .NET Framework. [15]

Ovšem aplikace platformy .NET Framework nemusí být nutně webové, neboť platforma poskytuje sadu tříd určenou pro formuláře Windows (Windows Forms, WinForms). Tato sada byla navržena pro implementaci smart klientských aplikací, což jsou aplikace poskytující místní uživatelské rozhraní přímo pomocí schopností uživatelského rozhraní Windows a nepoužívající HTML jako prezentační vrstvu. [15]

Chytré klientské aplikace mají oproti webově orientovaným aplikacím několik výhod. Protože se kód vykonává namístě, není nutno podnikat okružní cestu přes síť na server a zpět, což znamená, že chytré klientské aplikace jsou méně dostupné vzhledem k latentním nebezpečím na síti. [15]

Protože je chytrá klientská aplikace v důvěrné relaci vůči stroji, na němž běží, může obvykle poskytnout uživateli větší komfort ve formě vlastní grafiky, nastavení fontu písma a ovládacích prvků. Takovéto klientské aplikace mohou také, díky své povaze, používat všechny prostředky, dostupné na klientském počítači. [15]

Vývojáři formulářů Windows mohou využívat kompletnější sadu lightweight obalů systémových API než vývojáři MFC. Velice široké pokrytí knihovny tříd platformy .NET znamená, že už vývojáři nemusí až na výjimky volat systémová API, což v případě MFC neplatí. [15]

Zkrátka formuláře Windows jsou tváří klienta platformy .NET, který poskytuje integrované vývojové prostředí IDE (Integrated Development Environment) založené na formulářích a do něhož byly zakomponovány nejlepší objektové modely předchozích typů uživatelských rozhraní. Kromě toho mají schopnost, kterou nemá žádný jiný rámec založený na Windows, a to rozmísťovací dispozice webových aplikací založených na kódu HTML. [15]

Podobně jako každý jiný jmenný prostor platformy .NET také ***System.Windows.Forms*** se skládá z jednotlivých tříd, delegátů, rozhraní a výčtů. Přestože se vzhled uživatelského rozhraní konzolové aplikace CUI (Console User Interface) a grafického uživatelského rozhraní GUI (Grafical User Interface) liší na první pohled, ve skutečnosti proces budování formulářových aplikací Windows nic víc, než se naučit manipulovat s novou množinou typů pomocí syntaxe jazyka C#, popř. dalších jazyků platformy .NET. [15]

Z vysokoúrovňového hlediska lze stovky typů tohoto jmenného prostoru zhruba seskupit do těchto všeobecných kategorií, tedy dle [15]:

- **jádro infrastruktury:**
  Jedná se o typy, reprezentující klíčové operace programu formulářů platformy .NET, např. Form, Application apod., a další typy, usnadňující interoperabilitu se zděděnými ovládacími prvky ActiveX.
- **ovládací prvky:**
  S těmito typy se vytvářejí bohatě vybavená uživatelská rozhraní, přičemž všechny tyto prvky se odvozují ze základní třídy Control. Ovládací prvky lze konfigurovat v době návrhu a za běhu jsou viditelné.
- **komponenty:**
  Tyto typy sice nejsou odvozeny ze základní třídy Control, poskytují však programu formulářů platformy .NET vizuální schopnosti. Mnohé komponenty nejsou při běhu viditelné, nicméně se dají konfigurovat v době návrhu.
- **společná dialogová okna:**
  Formuláře Windows poskytují pro běžně používané operace řadu předem připravených dialogových oken. Samozřejmě si lze definovat svá vlastní dialogová okna, pokud standardní dialog plně neuspokojuje potřeba programátora.

Význam základních typů jmenného prostoru ***System.Windows.Forms*** je následující, tedy dle [15]:

Třídy jmenného prostoru	Význam
Application	Tato třída zapouzdřuje funkci formulářových aplikací Windows při běhu.
Button, CheckBox, ComboBox, DateTimePicker, ListBox, MaskedListBox, MonthCalendar, PictureBox, TreeView	Tyto třídy odpovídají různým prvkům GUI.
FlowLayoutPanel, TableLayoutPanel	Verze .NET 2.0 obsahuje manažery rozvržení, kteří automaticky uspořádají ovládací prvky formuláře při změně jejich velikosti.
Form	Tato třída reprezentuje hlavní okno aplikace nebo dceřiné okno vícedokumentového rozhraní formulářové aplikace Windows.
ColorDialog, OpenFileDialog, SaveFileDialog, FontDialog, PrintPreviewDialog, FolderBrowserDialog	Tato třída reprezentuje standardní dialogová okna pro běžné operace v GUI.
Menu, MainMenu, MenuItem, ContextMenu, MenuStrip, ContextmenuStrip	Pomocí těchto typů se vytvářejí systémy menu s pruhem nabídek nejvyšší úrovně a systémy kontextových nabídek. Nové ovládací prvky ve verzi .NET 2.0 MenuStrip a ContextMenuStrip umožňují budovat takové nabídky, které mohou obsahovat nejenom tradiční prvky rozevírací nabídky, ale také jiné ovládací prvky.
StatusBar, Splitter, ToolBar, ScrollBar, StatusStrip, ToolStrip	S těmito typy se zkrášlují formuláře pomocí populárních dceřiných ovládacích prvků.

Tab. 4.4.1: význam základních typů jmenného prostoru ***System.Windows.Forms***

4.4.1 Ovládací prvky

Základní jednotku uživatelského rozhraní UI (User Interface) představuje ve WinForms ovládací prvek, přičemž jím může být cokoliv, co přímo komunikuje s uživatelem v nějakém regionu definovaném kontejnerem. Patří zde dle [15]:

- automatické ovládací prvky
- standardní ovládací prvky
- uživatelské ovládací prvky
- třída formulářů Form

Standardní ovládací prvky

Ovládací prvek (Control) je třída odvozená ze základní třídy ***System.Windows.Forms.Control*** a zodpovědná za kreslení části kontejneru, což může být buď formulář, nebo jiný ovládací prvek. WinForms nabízejí celou řadu standardních ovládacích prvků, dostupných v toolboxu vývojového prostředí Microsoft Visual Studio .NET, tedy dle [15]:

- **akční ovládací prvky:**
  Typickými představiteli jsou tlačítko (Button) nebo panel nástrojů (ToolBar). Uživatel jimi vyvolává událost. Dále zde patří také nabídky, tj. hlavní nabídka (MainMenu) a kontextová nabídka (ContextMenu). S výjimkou tlačítka všechny ostatní představují kontejnery pro více podobjektů.
- **hodnotové ovládací prvky:**

Některé hodnotové ovládací prvky, např. popisek (Label) nebo obrázek (PictureBox), pouze zobrazují uživateli hodnotu ve formě textu nebo obrázku. Změnu hodnoty naopak umožňují např. prvky textové pole (TextBox) nebo prvek pro výběr data a času (DateTimePicker).

- **ovládací prvky pro seznam:**
  Otevřený seznam (ListBox) nebo pole se seznamem (ComboBox) zobrazují uživateli seznam údajů. Jiné ovládací prvky z této kategorie, např. mřížka dat (DataGrid), umožňují dané údaje měnit.

- **kontejnerové ovládací prvky:**
  Do této kategorie patří rámeček skupiny prvků (GroupBox), panel (Panel) a listovací rámeček (TabControl), umožňující seskupovat do kontejneru, v němž si lze jiné ovládací prvky libovolně uspořádat.

Akční ovládací prvky

Dále zde patří také nabídky, tj. hlavní nabídka (MainMenu) a kontextová nabídka (ContextMenu). S výjimkou tlačítka všechny ostatní představují kontejnery pro více podobjektů. [15]

Hodnotové ovládací prvky

Jak již bylo uvedeno, tyto prvky tvoří sadu ovládacích prvků, sloužících nejen pro zobrazení, ale také na editaci údajů. Podle datového typu hodnoty je lze rozdělit do několika skupin, tedy dle [15]:

- **řetězcové hodnoty:**
  Zde patří popisek (Label), hypertextový popisek (LinkLabel), textové pole s formátováním (TextBox) a stavový řádek (StatusBar).

- **číselné prvky:**
  Zde patří číselník (NumericUpDown), vodorovný posuvník (HScrollBar), svislý posuvník (VScrollBar), ukazatel průběhu akce (ProgressBar) a reostatový posuvník (TrackBar).

- **booleovské hodnoty:**
  Zde patří zaškrtávací políčko (CheckBox) a přepínač (RadioButton).

- **datum a čas:**
  Zde patří prvek pro výběr data a času (DateTimePicker) a několikaměsíční kalendář (MonthCalendar).

- **grafické hodnoty:**
  Zde patří obrázek (PictureBox) a náhled před tiskem dokumentu (PrintPreviewControl).

Ovládací prvky pro seznam

Tyto prvky najednou zobrazí v seznamu více hodnot; patří zde vedle prvků ComboBox, ListBox a DataGrid také CheckedListBox, DomainUpDown, ListView a TreeView. Většina těchto prvků prezentuje svůj seznam objektů jako kolekci jednotlivých položek (Items). [15]

Kontejnerové ovládací prvky

Zatímco ovládací prvky pro seznam obsahují několik objektů v kolekci, úkol kontejnerových ovládacích prvků spočívá v seskupení více ovládacích prvků různých druhů. Velikost kontejnerů lze samozřejmě libovolně měnit, a to buď přes vlastnost, nebo lze využit prvek dělící pruh (Splitter) přichycený na některé straně kontejneru. [15]

Uživatelské ovládací prvky

Ovládací prvky kreslené vlastníkem (Owner-Draw Control) sice umožňují do jisté míry řídit své vykreslování, ale v případě plné kontroly je vhodné si vytvořit vlastní ovládací prvek (Custom Control). Rozlišujeme tyto prvky dle [15]:

- **ovládací prvky přímo odvozené ze základní třídy Control:**
  Tento přístup umožňuje kompletně zpracovávat vstup y výstup ovládacího prvku.
- **ovládací prvky odvozené z třídy ScrollableControl:**
  Tento přístup využívá vlastností třídy Control a navíc nabízí integrovanou podporu posouvání.
- **ovládací prvky odvozené z existujícího prvku:**
  Tento přístup obohacuje chování již stávajícího standardního ovládacího prvku.

5 MATLAB Builder for. NET

5.1 Úvod do problematiky

Všestranné vývojové prostředí MATLAB poskytuje balíčky nástrojů (toolboxy) MATLAB Compiler a MATLAB Builder for .NET (v novějších verzích hovoříme o MATLAB Builder NE) k propojení s ostatními programovacími jazyky. Obecná úloha spočívá v podpoře zabudovaných matlabovských funkcí, určených pro programátory a vývojáře, např. pracujících s jazykem C#, který umožňuje pracovat třídami vytvořenými z MATLAB-souborů (M-files). Nástroj MATLAB Builder for .NET rovněž podporuje objektový model komponent COM (Component Object Model), využitelný ve všech programovacích jazycích podporujících tuto technologii. [16]

.NET komponenty lze obecně volat pomocí jakékoliv CLS jazyka, přičemž pro svůj běh potřebují MCR (MATLAB Compiler Runtime), kompletní sada sdílených knihoven podporujících MATLAB. MCR obsahuje MATLAB Compiler pro zajištění distribuovatelnosti .NET komponent nebo komponent COM. Jinými slovy, pokud je .NET komponenta vytvořena a zároveň implementována na tomtéž stroji, který má pochopitelně nainstalovaný MATLAB, není MCR potřebný, protože běh aplikací založených na MATLABu zajistí sám MATLAB; v opačném případě se musí na cílový stroj MCR nainstalovat. Výsledkem pak nemusí jen komponenta samotná, nýbrž samospustitelný balíček, obsahující uživatelskou komponentu a MCR. [16]

V dalším výkladu se již zaměříme pouze na M-soubory, platformu .NET a programovací jazyk C#. Vzájemnou relaci mezi jazykem C# a prostředím MATLAB lze využít hlavně pro řešení jednoduchých či složitějších matematických úloh v oblasti aplikované matematiky, například dle [16]:

- jednoduché a složitější matematické operace
- jednoduché i složitější operace s maticemi
- řešení soustav lineárních rovnic
- operace s maticemi – např. výpočet vlastních čísel matice
- řešení obyčejných lineárních diferenciálních rovnic
- numerická derivace
- numerická integrace
- interpolace
- operace s polynomy
- integrální transformace

Pro úplnost dodejme, že MATLAB zpracovává jakékoliv výpočty pomocí matic. Vygenerované funkce pro C# lze využít nejen pro aplikace se systémovou konzolí, ale také pro aplikace vybavené grafické-uživatelským rozhraním (GUI) ve formě Windows formulářů (WinForms). Aplikace (převážně konzolového typu) ovšem mohou obsahovat také různé typy grafů generovaných prostředím MATLAB, konkrétně pak MATLAB Graphics Figures, tedy dle [16]:

- vykreslení dvojdimenzionálního grafu (2D Graph) se zadaným polem dat
- vykreslení dvojdimenzionálního grafu (2D Graph) s daty uloženými v souboru
- vykreslení dvojdimenzionálního grafu (2D Graph) s více průběhy – zdroj zadaná data
- vykreslení dvojdimenzionálního grafu (2D Graph) s více průběhy – pole dat v jazyce C#
- vykreslení dvojdimenzionálního grafu (2D Graph) s více průběhy – data uložená v souboru

5.2 Popis aplikace Helikoptéra CE 150

Z výše uvedeného výčtu možností využití nástroje MATLAB Builder for .NET jsme vybrali složitější matematické operace, konkrétně pak v návaznosti na řešenou problematiku výpočet H∞ norem citlivostních funkcí, H∞ robustního regulátoru a nominální soustavy. V dalším výkladu rozebereme postup od M-souboru až po volání procedury v jazyce C# prostřednictvím grafických prvků čelního panelu formuláře systému Windows]

Vytváříme-li aplikace s formuláři Windows v jazyce C# pomocí nástroje MATLAB Builder .NET, zjednodušený postup se skládá z několika důležitých kroků, tedy:

- definování vstupních a výstupních dat
- vytvoření MATLAB-souborů obsahující funkci s libovolným počtem vstupních parametrů a jedním výstupním parametrem
- vygenerování .NET komponenty z M-souborů pomocí příkazu `deploytool` (ve starších verzích MATLABu příkazem `dotnettool`)
- přidání reference na dynamicky linkovanou knihovnu ***MWArray.dll***
- přiřazení vstupních dat proměnným datového typu double v jazyce C#
- vytvoření metody v jazyce C# s příslušnými vstupními parametry a jedním výstupním parametrem
- vytvoření instancí zástupních proměnných
- přetypování výstupního výsledku z metody na datový typ double v jazyce C#
- využití tohoto výsledku ve WinForms aplikaci – návaznost dat na grafické prvky formulářů

Kompletní postup pak lze rozložit do čtyř úloh, tedy:

- **vytvoření M-souborů** *MathWorks MATLAB 7.6 (R2008a)*
- **vytvoření .NET komponent** *MathWorks MATLAB Builder NE 2.2.2*
- **implementace .NET komponent** *Microsoft Visual Studio 2008 Professional Edition*
- **GUI aplikace** *Microsoft Visual Studio 2008 Professional Edition*

5.2.1 Vytvoření M-souborů

M-soubor tvoří zdroj nástroje MATLAB Builder for .NET. Prostředí MATLAB umožňuje M-soubory psát dvěma způsoby, respektive existují dva typy M-souborů:

- **skripty** *jednoduché M-soubory obsahující příkazy v jazyce MATLAB a neumožňující volání nebo předávání parametrů*
- **funkce** *sofistikovanější M-soubory umožňující vlastnit jeden nebo více vstupních i výstupních parametrů, a tudíž s nimi lze vytvářet bohatší aplikace*

Právě funkce nás budou při tvorbě M-souborů zajímat, protože i výstupem z nástroje MATLAB Builder for .NET musí být zase obecně funkce, byť v jiném formátu a určená pro jiný programovací jazyk.

Aplikace Helikoptéra CE 150 obsahuje celkem čtrnáct M-souborů (sedm pro elevaci a sedm pro azimut), přičemž každý z nich se zabývá výpočtem jiné H∞ normy:

pro elevaci

M-soubory pro elevaci		
Název souboru	**Význam**	**H∞ norma**
VypocetNormyCLE.m	výpočet normy uzavřené smyčky	$\left\|F\{\hat{P}(s),\hat{K}(s)\}\right\|_\infty$
VypocetNormyLE.m	výpočet normy otevřené smyčky	$\left\|\hat{L}(s)\right\|_\infty$
VypocetNormySE.m	výpočet normy citlivostní funkce	$\left\|\hat{S}(s)\right\|_\infty$
VypocetNormyTE.m	výpočet normy komplementární citlivosti	$\left\|\hat{T}(s)\right\|_\infty$
VypocetNormyRE.m	výpočet normy citlivosti řízení	$\left\|\hat{R}(s)\right\|_\infty$
VypocetNormyKE.m	výpočet normy robustního regulátoru	$\left\|\hat{K}(s)\right\|_\infty$
VypocetNormyGE.m	výpočet normy elevační soustavy	$\left\|\hat{G}(s)\right\|_\infty$

Tab. 5.2.1a: M-soubory pro výpočet H∞ norem v elevaci

pro azimut

M-soubory pro azimut		
Název souboru	**Význam**	**H∞ norma**
VypocetNormyCLA.m	výpočet normy uzavřené smyčky	$\left\|F\{\hat{P}(s),\hat{K}(s)\}\right\|_\infty$
VypocetNormyLA.m	výpočet normy otevřené smyčky	$\left\|\hat{L}(s)\right\|_\infty$
VypocetNormySA.m	výpočet normy citlivostní funkce	$\left\|\hat{S}(s)\right\|_\infty$
VypocetNormyTA.m	výpočet normy komplementární citlivosti	$\left\|\hat{T}(s)\right\|_\infty$
VypocetNormyRA.m	výpočet normy citlivosti řízení	$\left\|\hat{R}(s)\right\|_\infty$
VypocetNormyKA.m	výpočet normy robustního regulátoru	$\left\|\hat{K}(s)\right\|_\infty$
VypocetNormyGA.m	výpočet normy elevační soustavy	$\left\|\hat{G}(s)\right\|_\infty$

Tab. 5.2.1b: M-soubory pro výpočet H∞ norem v azimutu

Funkce pro výpočet H∞ norem v elevaci přebírají 9 vstupních parametrů; funkce pro výpočet H∞ norem v azimutu pak 8 vstupních parametrů. Všechny pak obsahují jeden výstupní parametr ve formě příslušné H∞ normy, přičemž algoritmus jejich výpočtu je pro všechny funkce stejný – vychází ze systémového propojení popsaného v podkapitole (3.3.2), přičemž přenosové funkce v elevaci a v azimutu jsou převzaty z jejich matematických modelů. Zobecněné prototypy M-funkcí pro výpočet norem jsou následující, tedy:

pro elevaci

```
function <Norma>E = VypocetNormy<Norma>E(Ub, MeE, wbeE, epsiloneE, KeE, MuE,
wbuE, epsilonuE, KuE)
```

pro azimut

```
function <Norma>A = VypocetNormy<Norma>A(MeA, wbeA, epsiloneA, KeA, MuA, wbuA,
epsilonuA, KuA)
```

Z prototypů funkcí je zřejmé, že vstupní parametry (s výjimkou napětí na servomechanismu U_B) se týkají tvarování váhových filtrů $\widehat{W}_e(s)$ a $\widehat{W}_u(s)$. Penalizace skupiny vstupních signálů (reference, NF porucha, VF porucha) je pevně v M-souborech nastavena. Důvod je ryze technický – pokud bychom chtěli upravovat váhy všech pěti filtrů, musely by elevační funkce přebírat celkem osmnáct vstupních parametrů, azimutové pak sedmnáct vstupních parametrů. Navíc z ryze programátorského hlediska se vždy snažíme omezovat počet vstupních parametrů na co nejnutnější počet – již více než pět parametrů není zcela vhodné. Druhým

důvodem, proč nejsou v tomto případě váženy vstupy, je fakt, že penalizace výstupů, respektive přenosové funkce $\widehat{W}_e(s)$ a $\widehat{W}_u(s)$, více ovlivňují výslednou odezvu v časové oblasti.

Každá z funkcí pak vrací pouze jediný parametr dle předpisů:

```
Norma<Norma>E = norm(<Norma>E, inf)

Norma<Norma>A = norm(<Norma>A, inf)
```

Pochopitelně by bylo možné vytvořit jedinou funkci pro elevaci a jedinou funkci pro azimut (popř. jedinou funkci se sedmnácti vstupními parametry) a výstupní parametry přebírat najednou. V tomto případě není problém na straně jazyka MATLAB, nýbrž na straně jazyka C#, který nedovoluje vracet více parametrů najednou pomocí několikanásobného volání klíčového příkazu `return()`. Při překladu se sice objeví pouze varování, ale prakticky se vykoná pouze první příkaz `return()`, proto se muselo vytvořit 14 M-souborů.

5.2.2 Vytvoření .NET komponent

MATLAB Builder for .NET, respektive MATLAB Builder NE, tvoří důležitý spojovací element mezi vytvořenými M-soubory a třídou jazyka C# platformy .NET, přičemž ne všechny verze MATLABu jej obsahují, neboť se nejedná o zcela standardní nástrojový balíček.
Abychom se ujistili, zdali daná verze MATLABu nástroj skutečně obsahuje, zadáme v příkazovém okně (Command Window) MATLABu příkaz `ver`, vypisující veškeré údaje o nainstalované verzi MATLABu, tedy:

```
MATLAB Builder EX                                 Version 1.2.10      (R2008a)
MATLAB Builder JA                                 Version 2.0.1       (R2008a)
MATLAB Builder NE                                 Version 2.2.2       (R2008a)
MATLAB Compiler                                   Version 4.8         (R2008a)
```

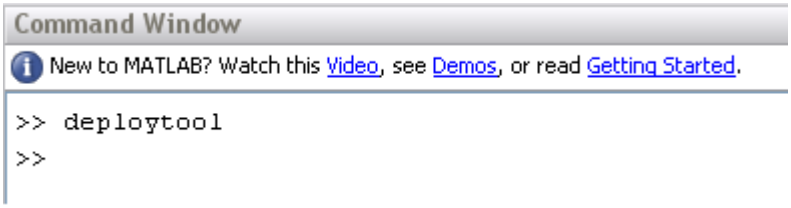
Obr. 5.2.1a: část výpisu nainstalovaných součástí prostředí MATLAB verze 7.6.0.324 (R2008a) s nainstalovaným balíčkem MATLAB Builder NE verze 2.2.2

Pokud se mezi instalovanými součástmi nástroj MATLAB Builder for .NET, respektive MATLAB Builder NE, neobjeví, nelze dále v postupu vytváření .NET komponenty z M-souborů pokračovat. Naše instalace jej ovšem obsahuje, proto můžeme přejít k dalšímu kroku, tj. vytvoření .NET komponenty.

Předpoklad pro práci s nástrojem MATLAB Builder for .NET a správně zkompilované zdrojové M-soubory umožňují vytvořit .NET komponentu pro jazyk C#. MATLAB Builder nabízí dva způsoby, jak vytvořit .NET komponentu, tedy:

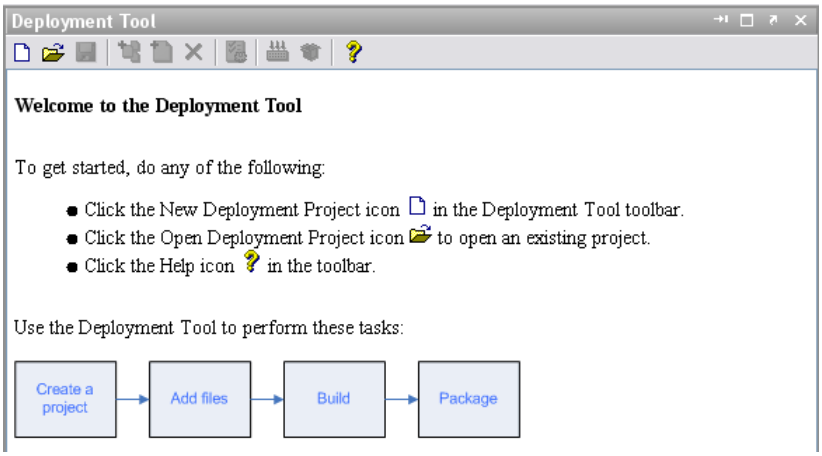
- `deploytool` *starší i novější verze MATLABu*
- `dotnettool` *výhradně starší verze MATLABu*

Oba způsoby umožňují generování .NET komponenty, v našem případě však využijeme univerzální možnosti deploytool. umožňující vytváření nejen samospustitelné aplikace (např. *.exe), ale také knihovny jazyka C++. V příkazovém okně (Command Window) napíšeme příkaz `deploytool`, tedy:



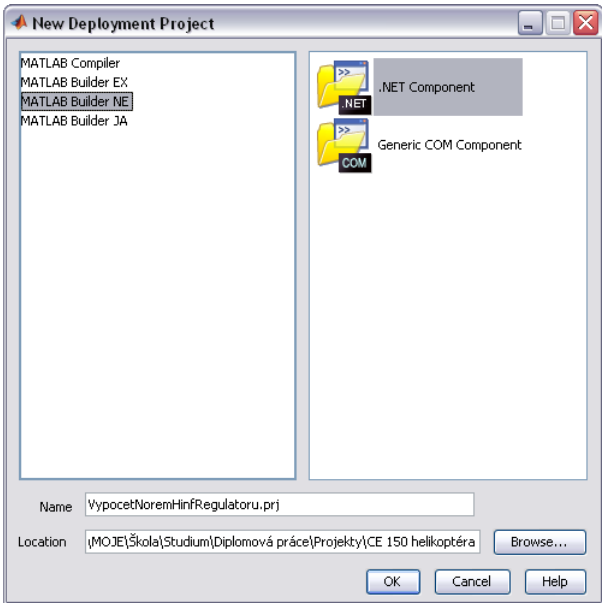
Obr. 5.2.1b: příkaz deploytool v příkazovém okně MATLABu pro otevření nástroje Deployment Tool

Alternativně lze k příkazu deploytool využít možnost ***Start -> MATLAB -> MATLAB Builder NE -> Deployment Tool***. Výsledkem této akce bude otevření úvodního formuláře nástroje Deployment Tool, tedy:



Obr. 5.2.1c: úvodní formulář nástroje Deployment Tool

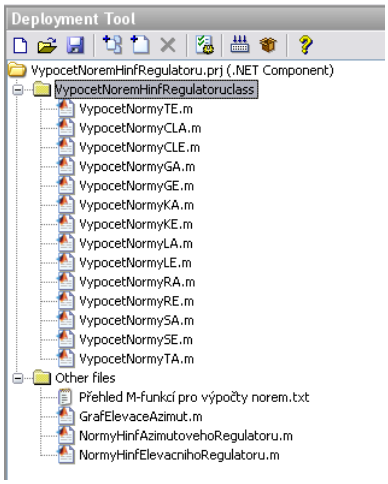
Úvodní formulář poskytuje nejzákladnější informace, co může uživatel udělat. Navíc obsahuje názorné blokové schéma vytvoření komponenty, tj. od vytvoření projektu, přidání souborů ke zpracování, vytvoření až po zabalení komponenty do samospustitelného balíčku. Nyní lze učinit první krok pro vytvoření komponenty, tj. založit nový projekt.



Obr. 5.2.1d: úvodní formulář pro založení nového projektu v nástroji Deployment Tool

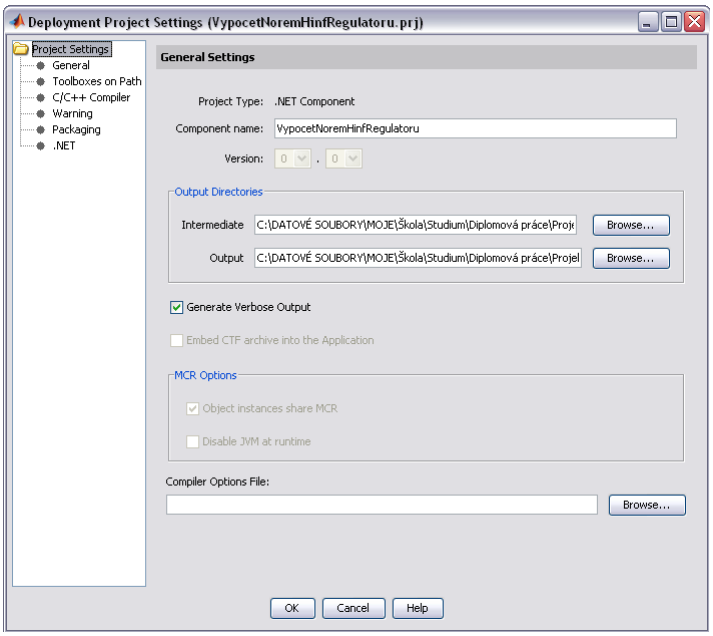
V úvodní formuláři pro založení nového projektu vybereme nástroj (MATLAB Builder NE) a generovaný objekt (.NET Component). Posléze si projekt pojmenujeme (v našem případě se jedná o

VypocetNoremHinfRegulatoru.prj) a udáme absolutní cestu jeho uložení na disk, popř. výměnné paměťové médium. Nově založený projekt obsahuje dvě prázdné složky ***VypocetNoremHinfRegulatoruclass*** a ***Other files***. Kompilované matlabovské soubory (nejčastěji M-soubory) vkládáme do složky ***VypocetNoremHinfRegulatoruclass***, čili tyto vložené soubory se stávají členy (objekty) budoucí třídy VypocetNoremHinfRegulatoruclass; vkládáme M-soubory uvedené v tab. 5.2.1a a tab. 5.2.1b. Do složky Other files si lze dle libosti vložit jakýkoliv soubor (např. textový soubor nebo M-soubor).



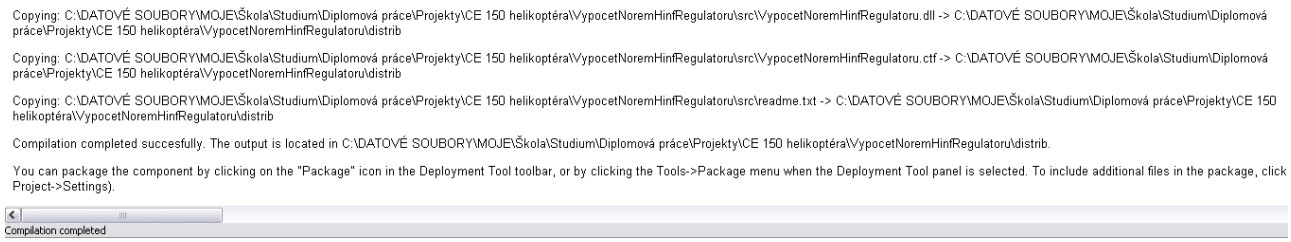
Obr. 5.2.1e: struktura projektu VypocetNoremHinfRegulatoru.prj

Máme-li vybrány požadované zdrojové M-soubory, můžeme zahájit vytváření .NET komponenty pomocí ***Menu - > Build - > .NET Object***. Případně si lze upravit nastavení projektu od všeobecného nastavení, dostupných toolboxů, nastavení C/C++ kompilátoru, varování při kompilaci, zabalení až po .NET. Nejdůležitější jsou samozřejmě všeobecná nastavení, definující absolutní cestu, kam se výsledná komponenta uloží, a umožňující si při kompilaci zobrazit průběh procesu (Generate Verbose Output). V případě nastavení pro .NET defaultně vytváříme privátní distribuční jednotku .NET (Private Assembly) pro defaultní verzi platformy Microsoft .NET Framework.



Obr. 5.2.1f: formulář nastavení parametrů projektu VypocetNoremHinfRegulatoru.prj – část všeobecného nastavení

Máme-li vybrány požadované zdrojové M-soubory, můžeme zahájit vytváření .NET komponenty pomocí ***Menu - > Build - > .NET Object***. Případně si lze upravit nastavení projektu od všeobecného nastavení, dostupných toolboxů, nastavení C/C++ kompilátoru, varování při kompilaci, zabalení až po .NET. Nejdůležitější jsou samozřejmě všeobecná nastavení, definující absolutní cestu, kam se výsledná komponenta uloží, a umožňující si při kompilaci zobrazit průběh procesu (Generate Verbose Output).

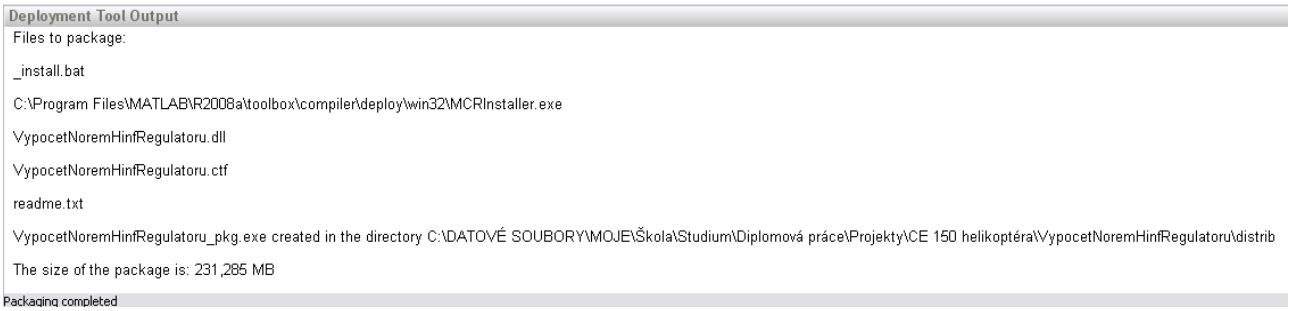


Obr. 5.2.1g: část výpisu průběhu kompilace projektu VypocetNoremHinfRegulatoru.prj

Samotná kompilace trvá pár minut v závislosti na výkonu počítače. Výstup pak představují dvě složky:

- složka distrib (Distribution)
- složka src (Sources)

V tomto stavu je možno .NET komponentu využít, a to za předpokladu, že cílový stroj má nainstalován MATLAB. Proces zabalení komponenty je tudíž volitelný, ovšem ovlivňuje obsah složky distrib.

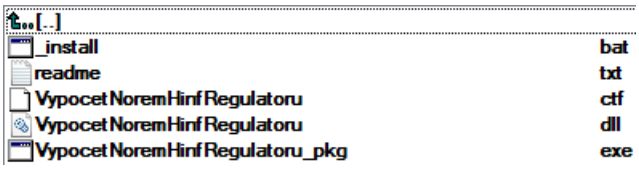


Obr. 5.2.1h: výpis průběhu zabalení .NET komponenty do balíčku VypocetNoremHinfRegulatoru_pkg.exe

Balíček je samospustitelný a obsahuje tyto soubory:

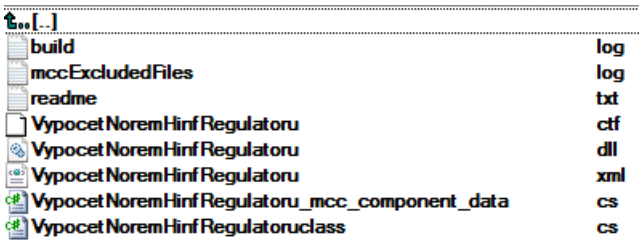
- **_install.bat** *jednoduchý skript informující o stavu instalace MCR*
- **VypocetNoremHinfRegulatoru.dll** *DLL knihovna .NET komponenty*
- **VypocetNoremHinfRegulatoru.ctf** *soubor s kompilovaným matlabovským kódem*
- **readme.txt** *informace a požadavky na nasazení MCR v projektu*
- **MCRInstaller.exe** *samospustitelný instalátor běhového prostředí MATLABu*

Obsah složky distrib je následující:



Obr. 5.2.1ch: obsah složky distrib (Distribution) obsahující také samospustitelný balíček VypocetNoremHinfRegulatoru_pkg.exe

Obsah složky src je následující:



Obr. 5.2.1i: obsah složky src (Sources)

Tyto dvě složky jsou výstupem z nástroje MATLAB Builder NE. V této chvíli přebírá veškerou práci na platformě .NET prostředí Visual Studio .NET.

5.2.3 Implementace .NET komponent

V této části budeme vytvořenou .NET komponentu implementovat na platformu .NET Framework pomocí integrovaného vývojového prostředí (IDE) Microsoft Visual Studio 2008 Professional Edition, v němž byl vytvořen projekt typu Windows Application s názvem Helikoptéra CE 150.

Přidání reference na knihovnu MWArray.dll

Protože vývojové prostředí implicitně neobsahuje potřebnou podporu prostředí MATLAB, je nutné tuto podporu zajistit explicitním přidáním reference na dynamicky linkovanou knihovnu MWArray.dll, přičemž opět záleží na verzi MATLABu, respektive nástroje MATLAB Builder, tedy:

pro verzi MATLAB 7.2 (R2006a)

> ***Solution Explorer -> Add Reference -> Browse -> Program Files -> MATLAB -> R2006a -> bin -> win32 -> MWArray.dll***

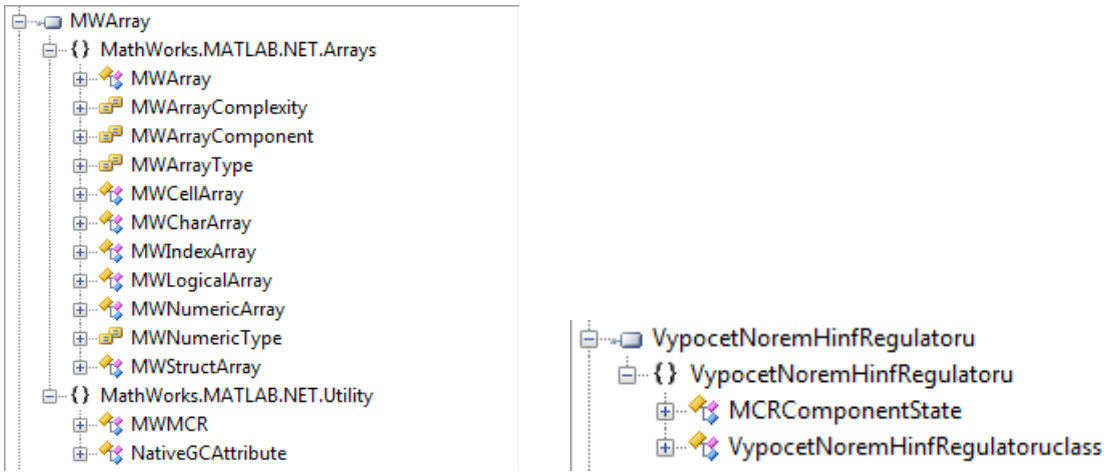
pro verzi MATLAB 7.6 (R2008a)

> ***Solution Explorer -> Add Reference -> Browse -> Program Files -> MATLAB -> R2008a -> toolbox -> dotnetbuilder -> bin -> win32 -> v2.0 -> MWArray.dll***

Tento krok je zcela zásadní a nesmí se opomenout, neboť knihovna ***MWArray.dll*** obsahuje dva důležité jmenné prostory:

- **MathWorks.MATLAB.NET.Arrays** *datové typy MW*
- **MathWorks.MATLAB.NET.Utility** *výjimky, inicializace MCR instance*

Podrobnější struktura knihovny ***MWArray.dll*** je následující:



Obr. 5.2.2a: základní struktura knihoven ***MWArray.dll*** a ***VypocetNoremHinfRegulatoru.dll***

V případě zástupných číselných vstupních a výstupních parametrů datového typu `MWNumericArray` se v jazyce C# přiřazuje takovým proměnným datový typ `double`. Pokud M-soubor obsahuje nečíselný parametr, např. řetězec datového typu `MWCharArray`, v jazyce C# se takovéto proměnné přiřazuje datový typ `string`. Stejný přístup lze aplikovat také pole v jazyce C#, tedy `double[]`.

Podpora matlabovských datových typů obsažených v knihovně ***MWArray.dll*** byla sice vytvořena, přesto musíme v kódu pomocí klíčového slova `using` deklarovat jmenný prostor, tedy:

```
using MathWorks.MATLAB.NET;
```

Nyní již můžeme podle potřeby přidávat části .NET komponenty vytvořené nástrojem MATLAB Builder NE pomocí sekvence ***Solution Explorer - > Add - > Existing Item…***, najdeme si umístění složky src (Sources), v níž vybereme následující soubor obsahující inicializační data pro MCR instanci:

- **VypocetNoremHinfRegulatoru_mcc_component_data.cs** MCC komponentní data

Volání funkcí na úrovni .NET komponenty

Na počátku jsme si vytvořili 14 M-souborů, z nichž jsme vytvořili .NET objekty, jejichž funkce jsou formálně stejné jako v M-souborech, přičemž vygenerovaný kód k jednotlivým funkcím obsahuje také varianty se sníženým počtem parametrů neboli k jedné funkci existuje několik prototypů s různým počtem parametrů – existuje možnost tzv. přetěžování funkcí. V našem případě jsme volali funkce se stejným počtem parametrů, jaký byl deklarován v M-souborech – 9 vstupních parametrů pro elevaci a 8 vstupních parametrů pro azimut. Při volání funkcí z .NET komponenty musíme napřed vytvořit referenci na knihovnu ***VypocetNoremHinfRegulatoru.dll***, a to stejným způsobem jako v případě ***MWArray.dll*** (pokud to za nás neudělá samo vývojové prostředí), tedy:

```
using VypocetNoremHinfRegulatoru;
```

Pokud by distribuční jednotka .NET obsahovala více dynamických knihoven, museli bychom deklarovat všechny tyto reference. Posléze na úrovni formuláře aplikace Helikoptéra CE 150 vytvoříme proceduru formálně odpovídající jedné z mnoha deklarovaných prototypů příslušné funkce v .NET komponentě. Zobecněné prototypy C# procedur pro výpočet norem jsou následující, tedy:

pro elevaci

```
private double Výpočet<Norma>ENormy(double argumentUb, double argumentMeE, double
argumentOmegabeE, double argumentEpsiloneE, double argumentKeE, double
argumentMuE, double argumentOmegabuE, double argumentEpsilonuE, double
argumentKuE)
```

pro azimut

```
private double Výpočet<Norma>ANormy(double argumentMeA, double argumentOmegabeA,
double argumentEpsiloneA, double argumentKeA, double argumentMuA, double
argumentOmegabuA, double argumentEpsilonuA, double argumentKuA)
```

Tato procedura v sobě obsahuje předání vstupních parametrů datového typu `double` parametrům datového typu `MWNumericArray` pomocí konstruktoru, respektive klíčového slova `new`; výstupní parametr je zatím inicializován nulovým ukazatelem `null`. Např. v elevačních funkcích lze situaci popsat následovně:

```
MWNumericArray ArgumentUb = new MWNumericArray(argumentUb);
MWNumericArray ArgumentMeE = new MWNumericArray(argumentMeE);
MWNumericArray ArgumentOmegabeE = new MWNumericArray(argumentOmegabeE);
MWNumericArray ArgumentEpsiloneE = new MWNumericArray(argumentEpsiloneE);
MWNumericArray ArgumentKeE = new MWNumericArray(argumentKeE);
MWNumericArray ArgumentMuE = new MWNumericArray(argumentMuE);
MWNumericArray ArgumentOmegabuE = new MWNumericArray(argumentOmegabuE);
MWNumericArray ArgumentEpsilonuE = new MWNumericArray(argumentEpsilonuE);
MWNumericArray ArgumentKuE = new MWNumericArray(argumentKuE);
MWNumericArray VýslednáNormaCLE = null;
```

Hlavní část této procedury tvoří deklaraci instance třídy `VypocetNoremHinfRegulatoruclass`, pochopitelně v konstrukci výjimky `try-catch`. Tato instance již přímo obsahuje položku s názvem M-souboru, tedy (v případě normy `CLE`):

```
VypocetNoremHinfRegulatoruclass ObjektNormaCLE = new
VypocetNoremHinfRegulatoruclass();

VýslednáNormaCLE = (MWNumericArray)ObjektNormaCLE.VypocetNormyCLE(ArgumentUb,
ArgumentMeE, ArgumentOmegabeE, ArgumentEpsiloneE, ArgumentKeE, ArgumentMuE,
ArgumentOmegabuE, ArgumentEpsilonuE, ArgumentKuE);
NováVýslednáNormaCLE = (double)VýslednáNormaCLE;
```

Výsledek procedury čili výstupní parametr je dán předpisem zobecněným předpisem, tedy:

v elevaci

```
return (NováVýslednáNorma<Norma>E);
```

v azimutu

```
return (NováVýslednáNorma<Norma>A);
```

Výstupní parametr `NováVýslednáNorma<Norma>E`, respektive `NováVýslednáNorma<Norma>A` je datového typu double a jedná se o globální proměnnou, předávanou do obsluhy události prvku čelního panelu. Výše uvedená konstrukce platí pro všech 14 procedur. Při každém volání funkce, respektive procedury se vstupními parametry, dbáme na správnou deklaraci datových typů, kdy připouštíme možnost validní implicitní konverze přetypováním.

Volání procedur na úrovni formuláře

Výše uvedené procedury přímo pracují s instancemi .NET komponenty a výsledky volání ukládají do globálních proměnných, se kterými lze v rámci příslušného jmenného prostoru `Helikoptéra_CE_150` libovolně pracovat, např. zobrazit jejich hodnoty ve vhodném indikačním prvku Windows formuláře (např. TextBox nebo Label). Obecně záleží na tom, zdali konkrétní prvek formuláře umožňuje zápis, čtení nebo obojí.

V případě vstupních parametrů chceme naprogramovat situaci, kdy jednak provážeme ovládací prvky číselného typu (např. NumericUpDown) nebo textového typu (např. TextBox) se vstupními parametry funkcí pro výpočet H∞ norem v elevaci a azimutu a jednak nějakým akčním ovládacím prvkem (např. Button) vstupní parametry potvrzením předáme, následně si zavoláme příslušnou proceduru, respektive sadu procedur, a výsledky zobrazíme.

Prototypy rutin obsloužení událostí Click akčních ovládacích prvků typu tlačítko (Button) jsou následující, tedy:

pro výpočet elevačních norem

```
private void VýpočetElevačníchNorem_Click(object sender, EventArgs e)
```

pro výpočet azimutových norem

```
private void VýpočetAzimutovýchNorem_Click(object sender, EventArgs e)
```

Provázání ovládacích prvků číselného (`PusovníkNapětí` – prvek typu TrackBar) a textového (uživatelská komponenta NumEdit odvozená z prvku typu TextBox) typu s převedením na společný datový typ `double`:

```
double HodnotaProUb = (this.PosuvníkNapětí.Value)/(-1000);
double HodnotaProMeE = Convert.ToDouble(this.ParametrMeE.Text);
double HodnotaProOmegabeE = Convert.ToDouble(this.ParametrOmegabeE.Text)
double HodnotaProEpsiloneE = Convert.ToDouble(this.ParametrEpsiloneE.Text);
double HodnotaProKeE = Convert.ToDouble(this.ParametrKeE.Text);
double HodnotaProMuE = Convert.ToDouble(this.ParametrMuE.Text);
double HodnotaProOmegabuE = Convert.ToDouble(this.ParametrOmegabuE.Text);
double HodnotaProEpsilonuE = Convert.ToDouble(this.ParametrEpsilonuE.Text);
double HodnotaProKuE = Convert.ToDouble(this.ParametrKuE.Text);
```

Pro úplnost dodejme, že hodnota proměnné `HodnotaProUb` je ve [V], pouze vstupní hodnotu zadáváme v [mV], a proto dělíme číslovkou 1000. Zobecněný prototyp volaných funkcí pro výpočet jednotlivých norem:

```
NováVýslednáNorma<Norma>E = this.Výpočet<Norma>ENormy(<Parametry v elevaci>);
NováVýslednáNorma<Norma>A = this.Výpočet<Norma>ANormy(<Parametry v azimutu>);
```

Zobecněné zobrazení výsledků v textovém formátu – prvek Label:

```
this.PopisekHodnoty<Norma>E.Text = NováVýslednáNorma<Norma>E.ToString();
this.PopisekHodnoty<Norma>A.Text = NováVýslednáNorma<Norma>A.ToString();
```

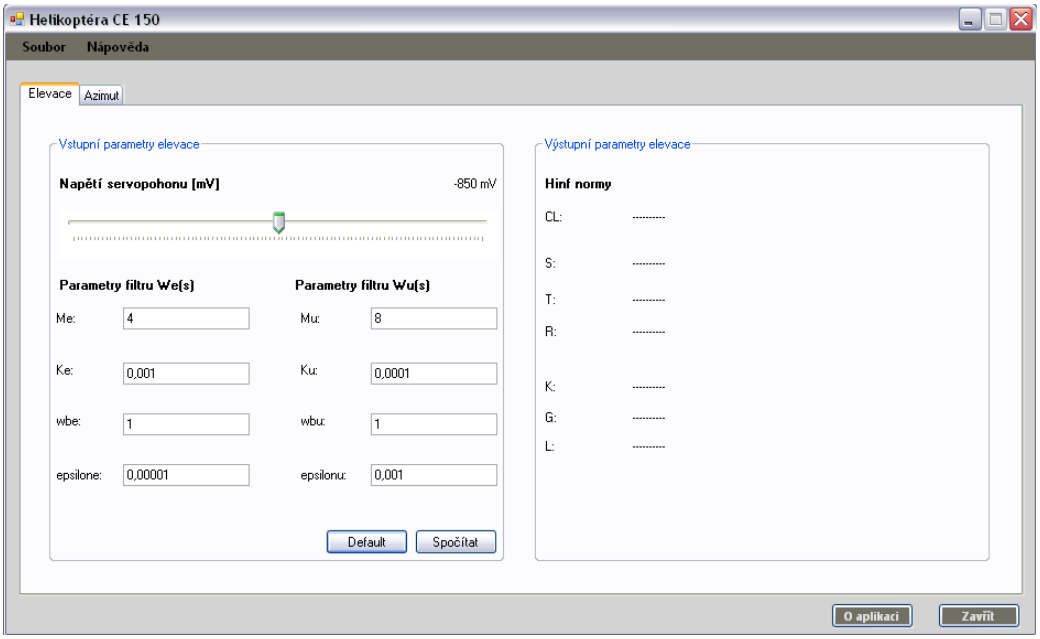
5.2.4 Graficko-uživatelské rozhraní aplikace

Struktura rozhraní

Graficko-uživatelské rozhraní aplikace obsahuje dvě záložky grafického prvku formuláře typu Tabcontrol:

- záložka Elevace
- záložka Azimut

Aplikace je vytvořena v jazyce C# na platformě .NET Framework 3.5 a určena pro výpočet H∞ norem systémů regulátorů a rozšířených soustav v elevaci a azimutu. Pro lepší orientaci uživatele jsou obě záložky pojaty stejným stylem a rozmístěním grafických prvků, rozdíl spočívá pouze v přítomnosti prvku posuvník pro nastavení napětí na servopohonu v [mV]. Při startu vypadá aplikace Helikoptéra CE 150 následovně:



Obr. 5.2.3a: formulář aplikace Helikoptéra CE 150 při jejím startu

Menu aplikace (prvek MenuStrip) má následující hierarchickou strukturu, tedy:

- **Soubor** podpoložka Konec *klávesová zkratka CTRL+Q (Quit)*
- **Nápověda** podpoložka Nápověda *klávesová zkratka F1*
  podpoložka O aplikaci *klávesová zkratka CTRL+A (About)*

Na obr. 5.2.3a je znázorněn hlavní formulář aplikace při startu, kdy v obou záložkách (Elevace, Azimut) jsou nastaveny defaultní hodnoty všech parametrů. Uživatel tudíž může buď okamžitě kliknout na tlačítko Spočítat (aplikace obsahuje dvě tato tlačítka – pro každou záložku), nebo Default (aplikace obsahuje dvě tato tlačítka – pro každou záložku), přičemž se znovu inicializují již nastavené parametry a žádná norma se nevypočítá, neboť událost výpočtu se odpaluje právě tlačítkem Spočítat.

Validní hodnoty a rozsahy vstupních parametrů			
Označení veličiny	**Rozsah hodnot**	**Defaultní hodnota**	**Korekční hodnota**
U_B	-0,90 až -0,80	-0,850	----------
M_{eE}, M_{eA}	0,00001 až 10,0000	4,000	0,0001
K_{eE}, K_{eA}	0,00001 až 0,99000	0,001	0,0100
ω_{beE}, ω_{beA}	0,00001 až 1,00000	1,000	0,0001
ε_{eE}, ε_{eA}	0,00001 až 1,00000	0,00001	0,0001
M_{uE}, M_{uA}	0,00001 až 10,0000	8,000	0,0001
K_{uE}, K_{uA}	0,00001 až 0,99000	0,0001	0,0010
ω_{buE}, ω_{buA}	0,00001 až 1,00000	1,000	0,0001
ε_{uE}, ε_{uA}	0,00001 až 1,00000	0,001	0,0001

Tab. 5.3.1: validní hodnoty a rozsahy vstupních parametrů v elevaci a azimutu

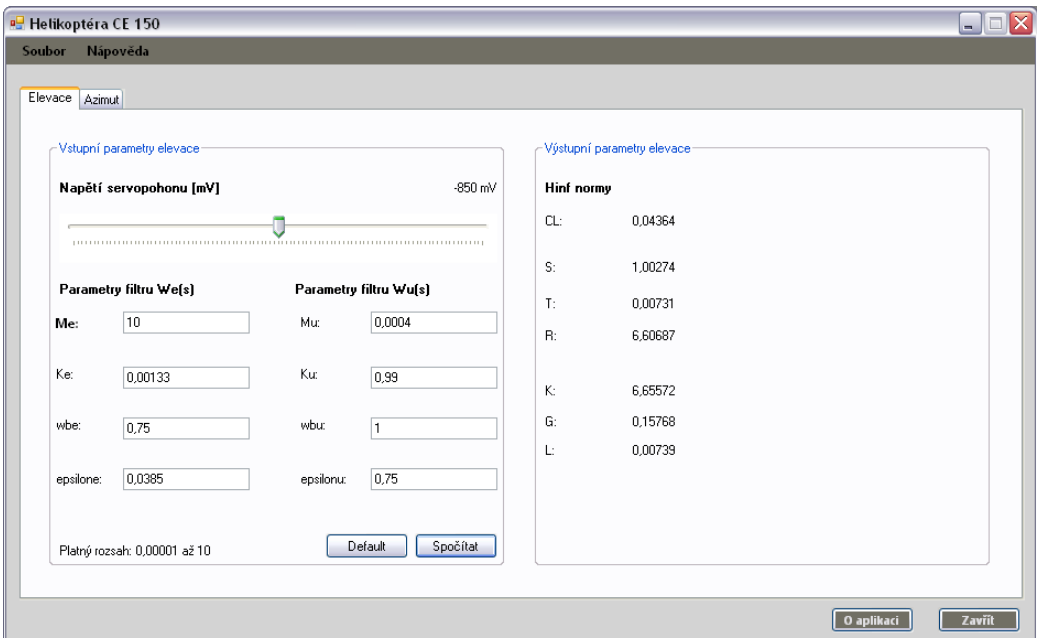
Ověřování validity vstupních dat

Základem úspěšného výpočtu jsou bezpochyby správně zadaná vstupní data. U hodnotových formulářových prvků číselného typu je situace poměrně snadná, neboť validní hodnotový rozsah lze omezit atributy Minimum a Maximum, čemuž odpovídá posuvník napětí (prvek TrackBar). U hodnotových formulářových prvků textového typu je situace začíná poněkud komplikovat, neboť počet nesprávně zadaných hodnot (úmyslně či neúmyslně) je vysoký. V takovém případě lze využít několika řešení:

- složitě ověřovat a pomocí systému podmínek zadanou hodnotu testovat
- využít externí komerční komponenty, nejlépe číselného typu (např. NumericEdit nástroje NI Measurement Studio .NET)
- využít derivátu stávajících komponent, tj. uživatelská úprava standardní komponenty formulářů Windows, dědící atributy standardní komponenty

Aplikace nakonec pro ověřování validity vstupních dat využívá kombinaci uživatelsky definované komponenty odvozené z prvku TextBox a testování podmínek ve formě interního omezení takové hodnoty. Prakticky to znamená, že budou-li hodnoty některých vstupních parametrů mimo svůj rozsah, aplikace automaticky tuto hodnotu nastaví na defaultní hodnotu, odpovídající rozsahu daného parametru, a počítá s ní, navíc ji také zobrazí. Rovněž nelze akceptovat situaci, kdy není parametr vůbec zadán; v takovém případě se v události TextChanged testuje obsah zadávaného textu, tj. je-li zjištěn prázdný řetězec nebo znak 0, okamžitě se políčko vyplní validní hodnotou. Navíc při pohybu kurzoru myši nad daným políčkem se odpaluje událost MouseMove, zobrazující v levém spodním rohu popisek o validním rozsahu parametru. Uživatelská komponenta typu NumEdit navíc neumožňuje zadávat písmena, speciální znaky nebo desetinnou tečku.

Další problém nastává v případě, kdy opravdu nechceme, aby byl daný parametr nulový; v tomto případě se to týká všech parametrů, neboť přenosové funkce váhových filtrů $\widehat{W}_e(s)$ a $\widehat{W}_u(s)$ mají předem stanovený tvar, striktně vyžadující jednu nulu v čitateli přenosu a jeden pól ve jmenovateli přenosu. Proto při zadání nuly do jakéhokoliv parametru se nastaví předem definovaná nenulová hodnota v závislosti na váze, jakou v přenosové funkci představují.



Obr. 5.2.3b: záložka Elevace po výpočtu H∞ norem při konfiguraci $U_B = -0{,}85\ [V]$, $M_e = 10$, $M_u = 0{,}004$, $K_e = 0{,}00133$, $K_u = 0{,}99$, $\omega_{be} = 0{,}75\ [rad \cdot s^{-1}]$, $\omega_{bu} = 1\ [rad \cdot s^{-1}]$, $\varepsilon_e = 0{,}0385$, $\varepsilon_u = 0{,}75$

Na obr. 5.2.3b je proveden výpočet H∞ norem pro reálný systém v elevaci (viz podkapitola 3.3.2) pro srovnání výsledných norem. Je zřejmé, že výsledky se při zadané konfiguraci vstupních parametrů shodují.

Možnosti nápovědy

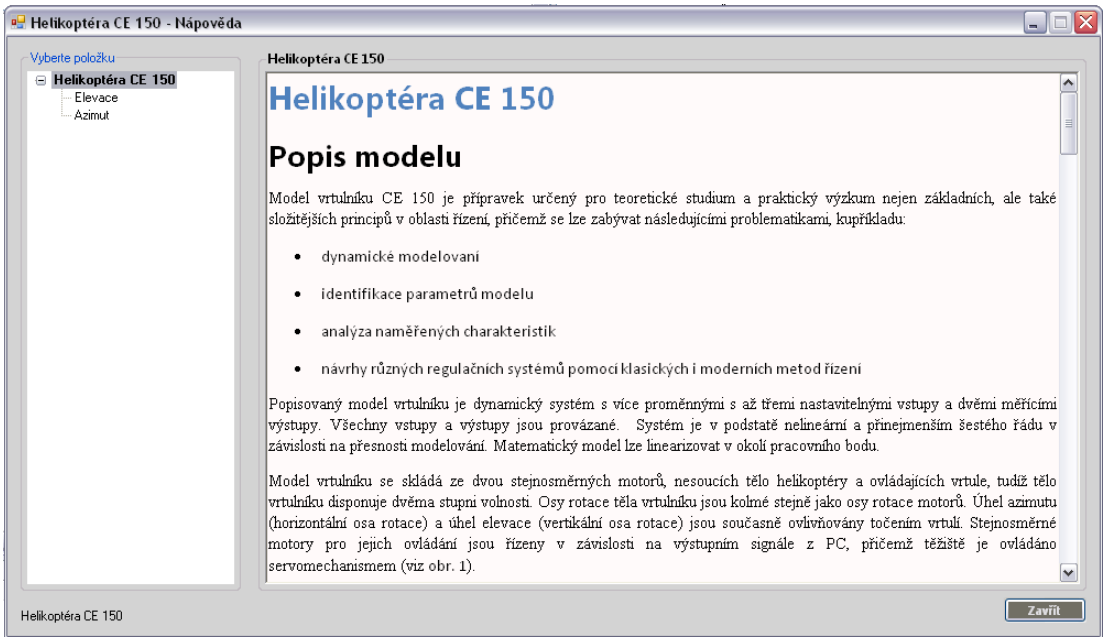
Každá aplikace by měla mít standardně implementovanou nápovědu, obsahující aspoň základní pokyny, jak s aplikací pracovat. Aplikace Helikoptéra CE 150 obsahuje jednoduchou nápovědu obsahující následující strukturu:

- **Helikoptéra CE 150** podpoložka Elevace *základní informace o problematice elevace*
  podpoložka Azimut *základní informace o problematice azimutu*

Formulář nápovědy lze spustit dvěmi způsoby:

- ***Menu -> Nápověda -> Nápověda***
- standardní klávesová zkratka F1

Formulář nápovědy aplikace Helikoptéra CE 150 je následující, tedy:



Obr. 5.2.3c: formulář nápovědy k aplikaci Helikoptéra CE 150

Formátovaný text k jednotlivým položkám je napsán v MS Wordu a uložen jako ****.rtf*** soubor. Komponenta RichTextBox pomocí metody LoadFile a definováním jejích argumentů (název souboru a režim formátování souboru) takový *.rtf soubor načte a zobrazí. V případě složitě formátovaného zdrojového ****.rtf*** souboru (barevné obrázky, formátovaný text) jej RichTextBox nemusí uspokojivě zobrazit.

6 Závěr

Cílem této diplomové práce je návrh H∞ robustních regulátorů pro laboratorní model vrtulníku CE 150 pomocí prostředí MATLAB&Simulink, přičemž nástroj MATLAB Builder for .NET, respektive MATLAB Builder NE, slouží k implementaci robustní regulace na platformu .NET Framework.

První kapitola je věnována komplexnímu modelování systému vrtulníku, tj.:

- *fyzikální modelování*
- *identifikace modelu*
- *linearizace modelu*
- *porovnání matematického modelu s reálným modelem*

Ve fyzikálním modelování byl uveden detailní matematický popis všech částí modelu vrtulníku, tj. od modelování dynamiky v elevaci a azimutu přes soustavy stejnosměrných motorů a vrtulí až po statický senzorický subsystém. Výsledek je reprezentován teoretickým model.

V identifikaci byly provedeny určité úpravy teoretického modelu na základě fyzikální interpretace vzájemných vazeb tak, aby bylo možno s modelem snáze pracovat, respektive jej popsat. Velký důraz je kladen na význam těžiště v závislosti na přiloženém napětí servopohon U_B coby neurčitosti pro robustní řízení a také na detailní odvození ofsetů IRC senzorů pro vyhodnocování úhlu elevace a úhlu azimutu. Výsledek je reprezentován empirickým modelem.

Linearizace tvoří další stupeň zjednodušení empirického modelu, aby bylo možno popsat dynamiku systému (soustavy motor-vrtule, mechanické části v elevaci a azimutu) popsat lineárními přenosovými funkcemi. Hlavní důraz je kladen na odvození linearizovaného modelu v elevaci; v případě azimutu se linearizační proces zjednodušuje, ovšem za cenu uvažování vlivu vazby elevace-azimut.

Porovnání matematického modelu s reálným modelem demonstruje přesnost matematického popisu vůči chování reálného modelu v elevaci a v azimutu.

Druhá kapitola se věnuje problematice robustního řízení, kde uvádíme nezbytný matematický aparát pro normy a neurčitosti. Hlavní důraz je kladen na návrh robustního regulátoru metodou H∞ se zaměřením na problém smíšené citlivosti (MSP), z níž vychází koncepce návrhu pro naši úlohu.

Koncepce návrhu robustního řízení předpokládá dva SISO robustní regulátory - jeden pro elevaci, druhý pro azimut. Z uvedeného vyplývá otázka autonomnosti řízení dvou veličin v křížové vazbě řešená použitím korekčního členu v již vazbě zmíněné vazbě elevace-azimut. V případě návrhu robustního regulátoru pro elevaci byly vytvořeny dva návrhy – pro matematický model, reprezentovaný odvozenou přenosovou funkcí, a pro reálný model, přičemž se ukázala očekávaná rozdílnost mezi matematickým modelem a reálným systémem, odstraněná vhodným nastavením váhových filtrů v systému robustního regulátoru a rozšířené soustavy. V případě azimutu byl vytvořen matematický návrh řízení coby demonstrace, že zvolená koncepce modifikovaného problému smíšené citlivosti lze skvěle funguje také na ryze nominálních systémech, tj. na systémech bez uvažování jakékoliv neurčitosti.

V kapitole s názvem Filozofie .NET jsme se seznámili se základní strukturou, terminologií a filozofií této v mnoha směrech pozoruhodné platformy, na níž chceme implementovat algoritmy robustní regulace pomocí jazyka C# a formulářů systému Windows.

Poslední, pátá kapitola se již plně věnuje nástroji MATLAB Builder for .NET, respektive MATLAB Builder NE, a jeho propojení mezi prostředím MATLAB a platformou .NET Framework, kdy onen spojovací článek představuje vygenerovaná uživatelská .NET komponenta.

Druhá část kapitoly se zabývá praktickou implementací vygenerované .NET komponenty do formulářové aplikace s názvem Helikoptéra CE 150, kdy detailně uvádíme postup, jak lze z M-souboru vygenerovat komponentu, např. pro hlavní programovací jazyk platformy .NET Framework – C#; výhoda tohoto spojení spočívá v distribuovatelnosti aplikace, kdy stačí na cílový stroj nainstalovat pouze běhové prostředí MCR (MATLAB Runtime Compiler). Samotná aplikace slouží k výpočtu klíčových H∞ norem systému v elevaci a v azimutu.

Přínos této práce bych shrnul do tří bodů, tedy:

- ***ucelený, názorný a transparentní matematický popis modelu vrtulníku vhodný pro jeho další využití např. ve výuce předmětů na Katedře měřicí a řídicí techniky, zaměřených na aplikovanou kybernetiku a programování na platformě .NET***
- ***hlubší poznání problematiky robustního řízení pro SISO systémy se zaměřením na metodu H∞ a modifikovaný problém smíšené citlivosti***
- ***hlubší poznání problematiky perspektivního propojení prostředí MATLAB s platformou .NET Framework pomocí nástroje MATLAB Builder for .NET***

Návrhů na vylepšení této práce by se našlo určitě mnoho, kupříkladu:

- *v případě modelování lze zdokonalit a rozvinout identifikační metody (např. úloha optimalizace parametrů) nebo metodiku exaktní linearizace s využitím ljapunovských funkcí*
- *v případě robustního řízení přidání parametrických nebo strukturovaných neurčitostí a zkoumání jejich vlivu na robustní stabilitu v elevaci nebo azimutu, případně H∞ regulace MIMO systému*
- *v případě interakce mezi prostředím MATLAB a platformou .NET vytvořit také propojení se Simulinkem, popř. vytvářet webové aplikace pro dálkové řízení modelu*

Model vrtulníku CE 150 byl již mnohokrát zpracován, např. v [3] byly uvedeny návrhy PID regulace, LQG regulace a adaptivního řízení MIMO systému vrtulníku. Identifikaci parametrů empirického modelu lze nalézt v [5], přičemž [1] tvoří základní studijní materiál této úlohy.

Literatura

[1] **Horáček P.:** *CE 150 Helicopter Model*, výukový manuál k modelu vrtulníku, Praha, 2002; <http://www.ing.unisi.it/~control/cmr/altro/heli_ce150_manual.pdf> [cit. 6. 5. 2009]

[2] **Kotlík B. a kol.:** *Matematické, fyzikální a chemické tabulky*, Havlíčkův Brod, FRAGMENT, 2003; ISBN: 80-7200-521-9

[3] **Hoč M.:** *Helicopter in Virtual Space*, diplomová práce, Praha, ČVUT, 2008; <http://epubl.ltu.se/1653-0187/2008/087/LTU-PB-EX-08087-SE.pdf > [cit. 6. 5. 2009]

[4] **Novák P.:** *Rotační inkrementální senzory*, článek v časopise AUTOMA, 10 / 2002; <http://www.odbornecasopisy.cz/download/au100232.pdf > [cit. 6. 5. 2009]

[5] **Karer G., Zupančič B.:** *Modelling and Identification of a Laboratory Helicopter*, Lublaň, 2003; <http://msc.fe.uni-lj.si/Papers%5Mathmod06_Karer.pdf > [cit. 6. 5. 2009]

[6] **Ivan J.:** *Matematika 1*, Bratislava, ALFA, 1983

[7] **Srovnal V.:** *Navrhování a realizace regulátorů*, výuková skripta, Ostrava, VŠB-TUO, 2004

[8] **Gu D.-W., Petkov P. Hr., Konstantinov M. M.:** *Robust Control Design with MATLAB*, Londýn, Springer-Verlag, 2005; ISBN-10: 1852339837

[9] *Norma (matematika)*, Wikipedie; <http://cs.wikipedia.org/wiki/Norma_vektoru> [cit. 6. 5. 2009]

[10] **Šebek M.:** *Robustní řízení*, soubor přednášek, Praha, FEL ČVUT, 2003; <http://polyx.com/_Robust/slides/ROR11-norma+citlivost.pdf> [cit. 6. 5. 2009]

[11] **Balátě J.:** *Automatické řízení*, Praha, BEN – Technická literatura, 2004; ISBN: 978-80-7300-148-3, EAN 9788073001483

[12] **Šulc B., Vítečková M.:** *Teorie a praxe návrhu regulačních obvodů*, Praha, ČVUT, 2004; ISBN: 80-01-03007-5

[13] *H-infinity Performance*, MathWorks MATLAB R2008a, HELP [offline]

[14] **Sename O.:** *Robust Control – Analysis and Design*, soubor přednášek, Grenoble, GIPSA-lab, 2008; <http://www.lag.ensieg.inpg.fr/sename/Robust_Control.pdf> [cit. 6. 5. 2009]

[15] **Troelsen A.:** *C# a .NET 2.0 profesionálně*, Brno, ZONER Press, 2006; ISBN: 80-86815-42-0

[16] **Phan J.:** *MATLAB C# Book Second Edition*, Colorado State University, LePhan Publishing, 2004; ISBN: 0-9725794-4-3

Přílohy

Adresářová struktura přiloženého DVD je následující:

- **C# a .NET platform:**
  Složka obsahuje projekt s aplikací Helikoptéra CE 150 se všemi podpůrnými soubory.

- **Literatura:**
  Složka obsahuje některé elektronické zdroje uvedené v odkazovém rejstříku a další materiály týkající se převážně problematiky robustního řízení, modelu helikoptéry CE 150 a nástroje MATLAB Builder for .NET.

- **MATLAB .NET Builder:**
  Složka obsahuje projekt ***VypocetNoremHinfRegulatoru*** se všemi podpůrnými soubory.

- **MATLAB & Simulink:**
  Složka obsahuje všechny použité M-soubory nejen pro výpočet H∞ norem, ale také další M-soubory týkající se modelu helikoptéry CE 150, a to včetně demonstračního MDL-souboru od firmy Humusoft s. r. o.

- **Obrázková část diplomové práce:**
  Složka obsahuje nejen všechny obrázky obsažené v textu diplomové práce, ale také ostatní obrázky, a to včetně zdrojových souborů (obrázky v PowerPointu).

- **Software:**
  Složka obsahuje instalace programů MathWorks MCR 7.8 a Microsoft .NET Framework 3.5; rovněž obsahuje dynamicky linkovanou knihovnu ***MWArray.dll*** pro verzi MATLAB Builder .NET 2.0.

- **Textová část diplomové práce:**
  Složka obsahuje kompletní text této diplomové práce.