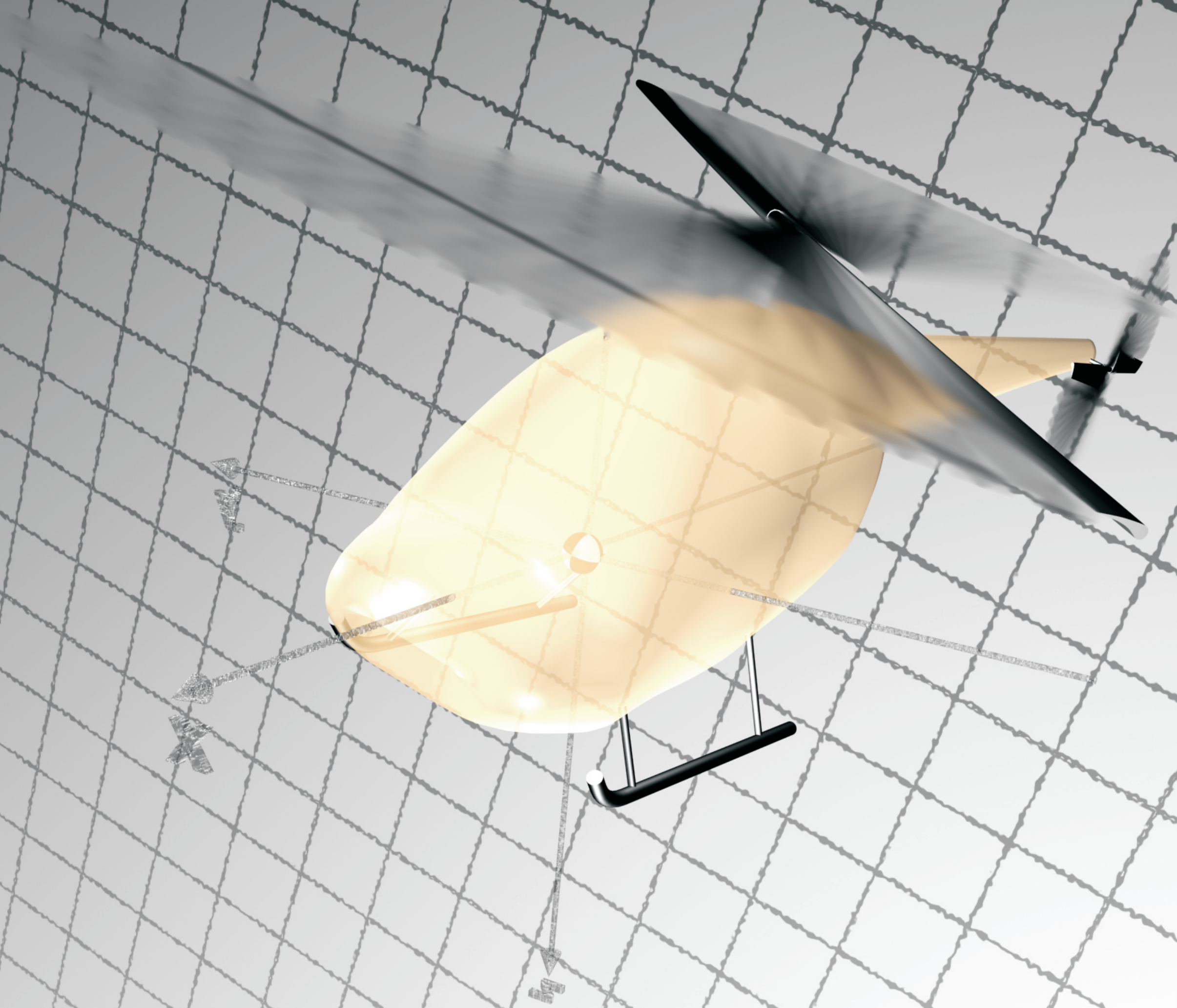


Autonomous

hover of a small scale electric

Helicopter

- Towards Helicopter Aided Mapping of Crops

Mikael Svenstrup
Kasper Winther

Intelligent Autonomous Systems
Aalborg University 2006/2007
Master Thesis

AALBORG UNIVERSITY

TITLE:
Autonomous Hover of an
Electric Helicopter

PROJECT SEMESTER:
Master Thesis,
September 2006 - June 2007

PROJECT GROUP:
1032e

GROUP MEMBERS:
Mikael S. Svenstrup
Kasper Winther

SUPERVISOR:
Anders la Cour-Harbo

NUMBER OF COPIES: 7

NUMBER OF PAGES IN MAIN REPORT: 165

TOTAL NUMBER OF PAGES: 210

APPENDED DOCUMENTS:
1 CD-ROM

FINISHED: 7th June 2007

ABSTRACT:

Helicopter Aided Mapping Of Crops (HAMOC) is a project, which aims at making a small scale electric helicopter able to obtain imagery of a crop field. This report considers the first steps towards this, by the use a Corona 120 electric helicopter. The project goal is to make the helicopter hover autonomously in the laboratory.
The project has been divided into four main parts; hardware implementation, modelling, state estimation, and control development. Regarding hardware, it has been chosen to control the helicopter by an external computer, and use an external power supply. The computer is interfaced to a servoboard on the helicopter by a serial connection. An existing helicopter model has been described, adapted to the Corona 120 helicopter, and the parameters have been determined. The 12 rigid body states of the helicopter have been estimated using image processing of camera data and an inertial measurement unit (IMU), which as fused in an extended Kalman filter. Four decoupled *PID* controllers have been developed to control the helicopter (for the z, y and x axes, and for the yaw angle, respectively). Hereafter, more advanced controllers have been researched.
The hardware works as expected, and it is possible to control the helicopter from the external computer. The developed controllers are able to control the nonlinear model in a simulation. At the end of the project period, autonomous control of the altitude of the real helicopter has been reached, and preliminary tests of a lateral controller have also been done. A helicopter crash has damaged some of the hardware, and has thus prevented further tests of the horizontal controllers, but it is expected that autonomous hover is close to be obtained. It is believed that the developed subsystems form a solid basis for further work on the project.

Table of Contents

Preface

This report is the documentation of a master project under the master specialization in Intelligent Autonomous Systems at Aalborg University. The thesis serves as a basis for the final examination, which takes place June 25, 2007. Associate Professor Anders la Cour-Harbo has been supervising the project, since it's beginning in September 2006.

The master project is an external project, as the project proposal is made in cooperation with Danish Institute of Agricultural Sciences (DIAS), Department of Agroecology. The project focuses on Helicopter Aided Mapping Of Crops (HAMOC).

Acknowledgements goes to:
- Jesper Waagepetersen, Research Director at DIAS, for his interest in the project.
- Oticon for supporting the authors and the project financially.
- Eric Johnson, Lockheed Martin Assistant Professor of Avionics Integration, School of Aerospace Engineering, Georgia Institute of Technology, for sharing his knowledge and experience within the field of unmanned aerial vehicles.

For references, the Harvard method is used. Figures, tables and equations are numbered consecutively inside each chapter.

Below each equation, most of the symbols included in the equation are explained, each explanation starting with a ►. Exceptions occur if a symbol has been explained within the same section. This notation is not only for the convenience of the reader, it also makes it possible to read and hopefully understand the equations, if read out of context.

The report can be found as a PDF-file at the enclosed CD:
[CD-ROM, 2007, `autonomous_helicopter.pdf`].
A start up guide to the system in lab is found in Appendix G on page 205. A description of the contents of the CD-rom can be found in Appendix A on page 173.

Aalborg University, 2007

Mikael S. Svenstrup

Kasper Winther

Nomenclature

Chapter Contents

Many different symbols and acronyms are used in this report. Some are general with well-known meaning, and others are specific for this project. For convenience, these are listed in this section. They are divided in four categories: First, the frame names are defined. Then all the variables and constants used throughout the report are explained. Last, the acronyms are listed.

Frames

Frame	Frame Name
$\{E\}$	Earth Frame
$\{B\}$	Body Frame
$\{H\}$	Hub Frame
$\{T\}$	Tail Rotor Frame

Table 1: *Table of frames.*

Variables

Table 2 lists all the variables used in this report. The first column is the symbol for the variable. They are placed in alphabetical order with the greek letters first. The second column is the physical meaning of the variable. The third column is the unit of the variable. All variables are given in SI-units.

Variable	Interpretation	Unit
Greek capital letters		
${}^{E}\mathbf{\Xi} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$	Position vector of $\{B\}$ in $\{E\}$	$[\,\mathrm{m}]$
${}^{E}\mathbf{\Xi}_{cam} = \begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix}$	The position output from the IPC	$[\,\mathrm{m}]$
${}^{B}\mathbf{\Xi}_{tr} = \begin{bmatrix} x_{tr} \\ y_{tr} \\ z_{tr} \end{bmatrix}$	Position vector of $\{T\}$ in $\{B\}$	$[\,\mathrm{m}]$
$\ddot{\mathbf{\Xi}}_{IMU} = \begin{bmatrix} \ddot{x}_{IMU} \\ \ddot{y}_{IMU} \\ \ddot{z}_{IMU} \end{bmatrix}$	The acceleration output from the IMU	$[\,\mathrm{m/s^2}]$
${}^{E}\mathbf{\Theta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}$	Attitude vector of $\{B\}$ in $\{E\}$	$[\,\mathrm{rad}]$
$\mathbf{\Theta}_{cam} = \begin{bmatrix} \phi_{cam} \\ \theta_{cam} \\ \psi_{cam} \end{bmatrix}$	The attitude output from the IPC	$[\,\mathrm{rad}]$
$\dot{\mathbf{\Theta}}_{IMU} = \begin{bmatrix} \dot{\phi}_{IMU} \\ \dot{\theta}_{IMU} \\ \dot{\psi}_{IMU} \end{bmatrix}$	The angular velocity output from the IMU	$[\,\mathrm{rad/s}]$
Ψ	Blade revolution angle	$[\,\mathrm{rad}]$
Ω_{mr}	Rotation velocity of the main rotor	$[\,\mathrm{rad/s}]$
Ω_{tr}	Rotation velocity of the tail rotor	$[\,\mathrm{rad/s}]$

Continued on next page

Table 2: *Table of variables.*

Continued from previous page

Variable	Interpretation	Unit
Greek lower case letters		
α	Blade angle of attack relative to air inflow angle	[rad]
β	Flapping angle	[rad]
θ_b	Blade pitch	[rad]
$\theta_{mrk,fcam}$	The angle between the initial position and the current position of the side marker as seen from the front camera	[rad]
$\theta_{mrk,hcam}$	The angle between the initial position and the current position of the hcam marker as seen from the helicam	[rad]
$\theta_{mrk,scam}$	The angle between the initial position and the current position of the side marker as seen from the side camera	[rad]
λ_{mr}	Inflow main rotor	[m/s]
λ_{tr}	Inflow tail rotor	[m/s]
μ_x	Advance ratio in the x-direction	[·]
μ_y	Advance ratio in the y-direction	[·]
μ_z	Advance ratio in the z-direction	[·]
$\boldsymbol{\tau}_w$	Torque provided by the wires attached to the helicopter	[Nm]
$\phi_{mrk,fcam}$	The angle between horizontal and the front marker center seen in the image plane.	[rad]
$\phi_{mrk,hcam}$	The angle between horizontal and the helicam marker center seen in the image plane.	[rad]
$\phi_{mrk,scam}$	The angle between horizontal and the side marker center seen in the image plane.	[rad]
ϕ_r	Inflow angle	[rad]
Roman capital letters		
A_1	Longitudinal swash plate angle of main rotor	[rad]
$A_{1,mr}$	Longitudinal blade pitch angle of main rotor	[rad]
B_1	Lateral swash plate angle of main rotor	[rad]
$B_{1,mr}$	Lateral blade pitch angle of main rotor	[rad]
B_e	Expected blue value of a pixel	[·]
B_p	Blue value of a pixel	[·]
C_L	Lift coefficient	[·]
C_T	Thrust coefficient	[·]
D	Drag of a rotor blade	[·]
$D_{pix,fcam}$	The distance to the front marker position relative to when the helicopter is in zero position	[pixels]
$D_{pix,scam}$	The distance to the side marker position relative to when the helicopter is in zero position	[pixels]

Continued on next page

Table 2: *Table of variables.*

Continued from previous page

Variable	Interpretation	Unit
D	Drag on a rotor blade	
D_{fp}	Drag on the front plane	
D_{tf}	Drag on the tail fin	
D_{tp}	Drag on the tail plane	
D_x	Drag on the fuselage	
D_y	Drag on the fuselage	
D_z	Drag on the fuselage	
$\boldsymbol{F}_{mr}$	Force provided by the main rotor	
$\boldsymbol{F}_w$	Force provided by the wires attached to the helicopter	[N]
G_e	Expected green value of a pixel	[·]
G_p	Green value of a pixel	[·]
L	Lift	[N]
R_e	Expected red value of a pixel	[·]
R_p	Red value of a pixel	
S_{lat}	Control input to lateral pitch servo motor	[·]
S_{lon}	Control input to longitudinal pitch servo motor	[·]
S_{mr}	Control input to main rotor DC-motor	[rad/s]
S_{tr}	Control input to tail pitch servo motor	[rad/s]
T_{mr}	Thrust of main rotor	[N]
T_p	Pixel threshold to determine if the pixel is a candidate to be on a circle in an image	
T_{tr}	Thrust of tail rotor	[N]
V_b	Air velocity relative to blade	[m/s]
U	Air velocity relative to $\{B\}$	[m/s]
Roman lower case letters		
a_0	Coning angle	[rad]
a_1	Longitudinal flapping of main rotor blade	[rad]
$a_{1,sb}$	Longitudinal flapping of stabilizer bar	[rad]
b_1	Lateral flapping of main rotor blade	[rad]
$b_{1,sb}$	Lateral flapping of stabilizer bar	[rad]
$d_{l,fcam,scam}$	The (minimum) distance between the lines from each of the wallcams towards the CM	[cm]
$\boldsymbol{l}_{fcam}$	The line, obtained by the front camera, passing through the CM of the helicopter	
$\boldsymbol{l}_{scam}$	The line, obtained by the side camera, passing through the CM of the helicopter	
p_{fcam}	Parameter for the line from the frontcam to the marker	[·]
p_{scam}	Parameter for the line from the sidecam to the marker	[·]

Continued on next page

Table 2: *Table of variables.*

Continued from previous page

Variable	Interpretation	Unit
${}^{E}\boldsymbol{p}_{mrk,fcam}$	Position of the front marker relative to CM, given in $\{E\}$ (dependent on the attitude $\boldsymbol{\Theta}$)	[m]
${}^{B}\boldsymbol{p}_{mrk,hcam}$	Position of the floor marker relative to CM, given in $\{B\}$ (dependent on the attitude $\boldsymbol{\Theta}$ and position $\boldsymbol{\Xi}$)	[m]
${}^{E}\boldsymbol{p}_{mrk,hcam}$	Position of the floor marker relative to CM, given in $\{E\}$ (dependent on the attitude $\boldsymbol{\Theta}$ and position $\boldsymbol{\Xi}$)	[m]
${}^{E}\boldsymbol{p}_{mrk,scam}$	Position of the side marker relative to CM, given in $\{E\}$ (dependent on the attitude $\boldsymbol{\Theta}$)	[m]
q	Weighting factor in the Kalman filter	[·]
$r_{pix,mrk}$	The pixel radius of a marker	
$\boldsymbol{u}$	The control input vector	
$\boldsymbol{v}$	Normal distributed measurement noise	
v_i	Inflow air velocity of the main rotor	[m/s]
$v_{i,tr}$	Inflow air velocity of the tail rotor	[m/s]
${}^{E}\boldsymbol{v}_{mrk,fcam}$	Unit vector pointing from the front camera towards the front marker	[m]
${}^{B}\boldsymbol{v}_{mrk,hcam}$	Unit vector pointing from the helicam towards the floor marker	[m]
${}^{E}\boldsymbol{v}_{mrk,scam}$	Unit vector pointing from the side camera towards the side marker	[m]
$\boldsymbol{x}$	The state vector	
$\hat{\boldsymbol{x}}$	The estimated state vector	
$\boldsymbol{w}$	Normal distributed process noise	
$x_{p,fcam}$	x co-ordinate of the pixel position of the center of the marker in the image from the front camera	[pixels]
$x_{p,hcam}$	x co-ordinate of the pixel position of the center of the marker in the image from the heli camera	[pixels]
$x_{p,scam}$	x co-ordinate of the pixel position of the center of the marker in the image from the side camera	[pixels]
$y_{p,fcam}$	y co-ordinate of the pixel position of the center of the marker in the image from the front camera	[pixels]
$y_{p,hcam}$	y co-ordinate of the pixel position of the center of the marker in the image from the heli camera	[pixels]
$y_{p,scam}$	y co-ordinate of the pixel position of the center of the marker in the image from the side camera	[pixels]
z	The measurement vector	
Calligraphed letters		
$\mathcal{D}$	Matrices in flapping equation 5.25	
$\mathcal{E}$	Matrices in flapping equation 5.25	

Continued on next page

Table 2: *Table of variables.*

Continued from previous page

Variable	Interpretation	Unit
$\mathcal{G}$	Matrices in flapping equation 5.25	
$\mathcal{J}$	Matrices in flapping equation 5.25	
$\mathcal{K}$	Matrices in flapping equation 5.25	
$\mathcal{K}_k$	The Kalman gain	
$\hat{\mathcal{P}}$	Prediction error covariance matrix	
${}^{B}_{E}\mathcal{R}$	The co-ordinate transformation matrix mapping from the earth frame to the body frame	

Table 2: *Table of variables.*

Constants

Table 3 is a list of all the constants used in this the report. For further information about how the constants are derived, the reader is referred to Section 5.5. The first column in Table 3 is the symbol for the constant. They are placed in alphabetical order with the greek letters first. The second column is the physical meaning of the constant. The third column is the value and unit of the constant. All constants are given in SI-units.

Constant	Interpretation	Value
Greek capital letters		
$\Omega_{mr,max}$	Maximum angular velocity of the main rotor	$190\ \mathrm{rad/s}$
$\Omega_{mr,min}$	Minimum angular velocity of the main rotor	$0\ \mathrm{rad/s}$
Greek lower case letters		
$\alpha_{pix,wall}$	The angle spanned by one pixel by the wallcams	$1{,}22 \cdot 10^{-3}$ rad
$\alpha_{pix,hcam}$	The angle spanned by one pixel by the helicam	$2{,}31 \cdot 10^{-3}$ rad
$\alpha_{view,wall}$	The horizontal width of the view of the cameras at a distance of 1 m	0,82 m
$\alpha_{view,hcam}$	The horizontal width of the view of the helicam at a distance of 1 m	0,76 m
${}^{E}\boldsymbol{\kappa}_{fcam}$	Position of front camera in earth frame	$\begin{bmatrix} 1{,}76 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{fp}$	Position of CM of front plane in $\{B\}$	$\begin{bmatrix} 0{,}08 \\ 0{,}01 \\ 0{,}06 \end{bmatrix}$ m

Continued on next page

Table 3: *Table of constants.*

Continued from previous page

Constant	Interpretation	Value
${}^{B}\boldsymbol{\kappa}_{hcam}$	Position of helicam in $\{B\}$	$\begin{bmatrix} -0{,}06 \\ -0{,}01 \\ 0{,}08 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{hub}$	Position of hubframe origin in $\{B\}$	$\begin{bmatrix} -0{,}01 \\ -0{,}01 \\ -0{,}11 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{IMU}$	Position of IMU in $\{B\}$	$\begin{bmatrix} 0{,}04 \\ 0{,}01 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{mrk,fcam}$	Position of the front marker given in $\{B\}$	$\begin{bmatrix} 0{,}19 \\ -0{,}01 \\ 0{,}02 \end{bmatrix}$ m
${}^{E}\boldsymbol{\kappa}_{mrk,hcam}$	Position of the marker on the floor given in $\{E\}$	$\begin{bmatrix} 0 \\ 0 \\ 1{,}02 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{mrk,scam}$	Position of the side marker given in body frame	$\begin{bmatrix} 0{,}13 \\ -0{,}04 \\ 0{,}02 \end{bmatrix}$ m
${}^{E}\boldsymbol{\kappa}_{scam}$	Position of side camera in earth frame	$\begin{bmatrix} 0{,}00 \\ -1{,}46 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{tf}$	Position of tail fin CM in body frame	$\begin{bmatrix} -0{,}47 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{tp}$	Position of tail plane CM in $\{B\}$	$\begin{bmatrix} -0{,}30 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{tr}$	Position of tail rotor frame in $\{B\}$	$\begin{bmatrix} -0{,}49 \\ -0{,}04 \\ -0{,}02 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{w}$	Position of to where the wires are attached to the helicopter given the in $\{B\}$	$\begin{bmatrix} 0{,}23 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
ζ_A	Damping ratio of lateral pitch servo	0,85 [·]
ζ_B	Damping ratio of longitudinal pitch servo	0,85 [·]
ζ_{mr}	Damping ratio of main rotor	1 [·]
ζ_{tr}	Damping ratio of tail pitch servo	0,85 [·]
θ_0	Main rotor collective pitch	0,35 rad
$\theta_{0,sb}$	Collective pitch of stabilizer bar	0,2 rad

Continued on next page

Table 3: *Table of constants.*

Continued from previous page

Constant	Interpretation	Value
$\dot{\theta}_{tr,max}$	Rate limit of tail pitch servo	$10{,}9\ ^{\text{rad}}/_{\text{s}}$
$\dot{\theta}_{A,max}$	Rate limit of lateral pitch servo	$10{,}9\ ^{\text{rad}}/_{\text{s}}$
$\dot{\theta}_{B,max}$	Rate limit of longitudinal pitch servo	$10{,}9\ ^{\text{rad}}/_{\text{s}}$
θ_{tw}	Main rotor blade twist	$-0{,}29$ rad
$\theta_{tr,offset}$	Collective pitch offset of tail rotor	$0{,}28$ rad
ξ_{fcam}	The angle between horizontal and the origin of $\{E\}$ seen from the front camera	$\arctan\left(\frac{-{}^{E}\kappa_{fcam,z}}{{}^{E}\kappa_{fcam,x}}\right)$ [rad]
ξ_{scam}	The angle between horizontal and the origin of $\{E\}$ seen from the side camera	$\arctan\left(\frac{-{}^{E}\kappa_{scam,z}}{{}^{E}\kappa_{scam,y}}\right)$ [rad]
ρ	Density of air	$1{,}29\ ^{\text{kg}}/_{\text{m}^3}$
$\sigma^2_{pos,IPC}$	Noise variance on the three position measurements from the IPC	$0{,}0025[\text{m}^2]$
$\sigma^2_{att,IPC}$	Noise variance on the three attitude measurements from the IPC	$0{,}01[\text{rad}^2]$
$\sigma^2_{acc,IMU}$	Noise variance for the acceleration measurements.	$1[^{\text{m}^4}/_{\text{s}^2}]$
$\sigma^2_{rot,IMU}$	Noise variance for the rotation measurements.	$[0{,}25\ ^{\text{rad}^2}/_{\text{s}^{\text{s}}}]$
$\dot{\psi}_{ref,max}$	Input saturation of tail pitch servo	$1{,}5\ ^{\text{rad}}/_{\text{s}}$
$\dot{\psi}_{ref,min}$	Input saturation of tail pitch servo	$-1{,}5\ ^{\text{rad}}/_{\text{s}}$
$\omega_{n,A}$	Undamped natural frequency of lateral pitch servo	$200\ ^{\text{rad}}/_{\text{s}}$
$\omega_{n,B}$	Undamped natural frequency of longitudinal pitch servo	$200\ ^{\text{rad}}/_{\text{s}}$
$\omega_{n,mr}$	Undamped natural frequency of main rotor	$5\ ^{\text{rad}}/_{\text{s}}$
$\omega_{n,tr}$	Undamped natural frequency of tail pitch servo	$200\ ^{\text{rad}}/_{\text{s}}$
Roman capital letters		
A_{fp}	Area of front plane	$0{,}0791\ \text{m}^2$
A_{tf}	Area of tail fin	$0{,}0042\ \text{m}^2$
A_{tp}	Area of tail plane	$0{,}0042\ \text{m}^2$
A_x	Area of body in x-axis direction	$0{,}0072\ \text{m}^2$
A_y	Area of body in y-axis direction	$0{,}0302\ \text{m}^2$
A_z	Area of body in z-axis direction	$0{,}0833\ \text{m}^2$

Continued on next page

Table 3: *Table of constants.*

Continued from previous page

Constant	**Interpretation**	**Value**
C_d	Drag coefficient for main rotor blades	0,008 [·]
$C_{d,tr}$	Drag coefficient for tail rotor blades	0,008 [·]
C_{ls}	Lift curve slope for main rotor blade	$7{,}6\,\mathrm{rad}^{-1}$
$C_{ls,sb}$	Lift curve slope for stabilizer bar	$3\,\mathrm{rad}^{-1}$
$C_{ls,tr}$	Lift curve slope for tail rotor blade	$2{,}5\,\mathrm{rad}^{-1}$
D_p	Colour distance between a pixel and an expected value	
G_1	Gear ratio between DC motor and main rotor	11,3 [·]
G_2	Gear ratio between main rotor and tail rotor	2,1 [·]
I_b	Inertia of main rotor blade	$0{,}001009\,\mathrm{kg}\cdot\mathrm{m}^2$
$I_{b,sb}$	Inertia of stabilizer bar	$0{,}000084\,\mathrm{kg}\cdot\mathrm{m}^2$
I_x	Moment of inertia of the body about x-axis	$0{,}004415\,\mathrm{kg}\cdot\mathrm{m}^2$
I_y	Moment of inertia of the body about y-axis	$0{,}024916\,\mathrm{kg}\cdot\mathrm{m}^2$
I_z	Moment of inertia of the body about z-axis	$0{,}026528\,\mathrm{kg}\cdot\mathrm{m}^2$
K_A	DC gain of lateral pitch servo	0,43 [·]
K_B	DC gain of longitudinal pitch servo	0,35 [·]
K_b	Bell factor	0,41 [·]
K_{gyro}	Gain from yaw velocity to gyro output	$-0{,}67\,\mathrm{s}/\mathrm{rad}$
K_h	Hiller factor	0,59 [·]
K_{mr}	DC gain of main rotor	1 [·]
K_s	Spring force constant for main rotor blade	$1{,}9\,\mathrm{N\cdot m}/\mathrm{rad}$
K_{tf}	Factor compensating for that only a part of the tail fin is covering the tail thrust area	0,97 [·]
K_{tr}	DC gain of tail pitch servo	0,48 [·]
M	Helicopter mass	1,02 kg
M_w	Mass of attached wires	$0{,}10\,\mathrm{kg}/\mathrm{m}$
R	Main rotor radius	0,38 m
R_i	Inner radius of stabilizer bar	0,04 m
R_o	Outer radius of stabilizer bar	0,12 m

Continued on next page

Table 3: *Table of constants.*

Continued from previous page

Constant	**Interpretation**	**Value**
R_{tr}	Radius of tail rotor	0,09 m
$S_{lat,max}$	Input saturation of lateral pitch servo	1 [·]
$S_{lat,min}$	Input saturation of lateral pitch servo	−1 [·]
$S_{lon,max}$	Input saturation of longitudinal pitch servo	1 [·]
$S_{lon,min}$	Input saturation of longitudinal pitch servo	−1 [·]
$S_{tr,max}$	Input saturation of tail pitch servo	1 [·]
$S_{tr,min}$	Input saturation of tail pitch servo	−1 [·]
Ts	Sampling time	0,005 s
Roman lower case letters		
b	Number of main rotor blades	2 [·]
b_{tr}	Number of tail rotor blades	2 [·]
c	Main rotor blade chord	0,05 m
c_{sb}	Stabilizer bar chord	0,05 m
c_{tr}	Tail rotor blade chord	0,03 m
d_{fp}	Drag coefficient for front plane	1,0 [·]
d_{tf}	Drag coefficient for tail fin	1,2 [·]
d_{tp}	Drag coefficient for tail plane	0,5 [·]
d_x	Longitudinal drag coefficient for body	1,0 [·]
d_y	Lateral drag coefficient for body	1,2 [·]
d_z	Vertical drag coefficient for body	1,0 [·]
e	Main rotor hinge offset	0,05 m
g	Gravitational acceleration	9,82 $^{m}/_{s^2}$
m_b	First mass moment of main rotor blade	0,0053 kg · m
r_{mrk}	The radius of a marker	1,5 cm
$res_{hcam,h}$	The horizontal video resolution of the helicam (number of pixels)	320
$res_{hcam,v}$	The vertical video resolution of the helicam (number of pixels)	240
$res_{wall,h}$	The horizontal video resolution of the wallcams (number of pixels)	480
$res_{wall,v}$	The vertical video resolution of the wallcams (number of pixels)	640

Continued on next page

Table 3: *Table of constants.*

Continued from previous page

Constant	Interpretation	Value
$x_{off,fcam}$	Pixel x-position of the front marker when the helicopter is in zero position.	[pixels]
$x_{off,hcam}$	Pixel x-position of the floor marker when the helicopter is in zero position.	[pixels]
$x_{off,scam}$	Pixel x-position of the side marker when the helicopter is in zero position.	[pixels]
$y_{off,fcam}$	Pixel y-position of the front marker when the helicopter is in zero position.	[pixels]
$y_{off,hcam}$	Pixel y-position of the floor marker when the helicopter is in zero position.	[pixels]
$y_{off,scam}$	Pixel y-position of the side marker when the helicopter is in zero position.	[pixels]
z_{floor}	The distance from the origin of $\{E\}$ to the ground level	1,02 m
Calligraphed letters		
$\mathcal{I}$	Inertia tensor of the helicopter	
$\mathcal{Q}$	Covariance matrix of the process noise, $\boldsymbol{w}$	
$\mathcal{R}$	Covariance matrix of the measurement noise, $\boldsymbol{v}$	

Table 3: *Table of constants.*

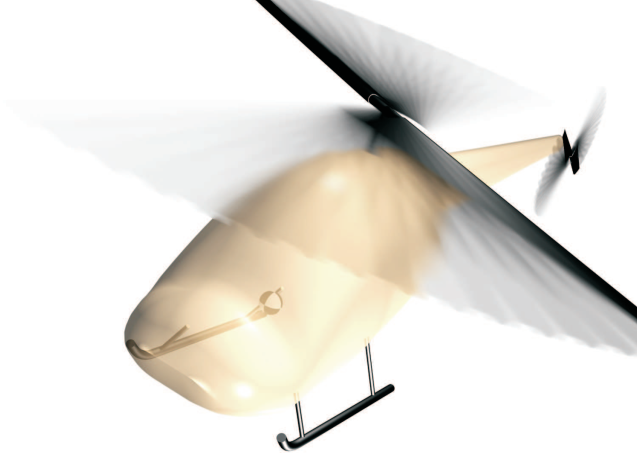
Acronyms

General acronyms used throughout this report are listed in Table 4.

Acronym	meaning
CC	Control Computer
CG	Center of Gravity
CHT	Circular Hough Transform
CM	Center of Mass
DOF	Degrees Of Freedom
EKF	Extended Kalman Filter
IMU	Islamic Movement of Uzbekistan (or Inertial Measurement Unit)
IPC	Image Processing Computer
OS	Operating System
PWM	Pulse Width Modulation
RC	Remote Control
RGB	Red, Green, Blue
ROI	Region Of Interest
ROO	Region Of Operation

Table 4: *Table of acronyms.*

Main Report



1

INTRODUCTION

Chapter Contents

1.1 Unmanned Aerial Vehicles

An Unmanned Aerial Vehicle (UAV) is some kind of aircraft, which is capable of flying without an on-board human pilot. It may be operated manually from a ground station, or it may be autonomous in the sense, that it is capable of fulfilling different tasks without human interaction, by means of advanced feedback control systems. The latter is what this project is about.

Research in UAVs has mainly been for military applications, such as surveillance, reconnaissance or target finding behind enemy lines. An example of one of the less successful UAV projects is SAGEMS “Sperwer-DK” (“Tårnfalken” in Danish), which is a 300 kg, propeller-driven airplane, which were meant for providing information about military targets or movements behind enemy lines. Unfortunetaly, it never left ground, due to mechanical and electrical malfunctions [Marfeldt, 2005, p. 12-13]. However, other countries are using UAV’s in military operations around the globe, and according to the weekly magazine “ingeniøren” [Holm, 2006, p. 22-23] it will become a huge industry in the next decade. At the moment, Saab Aerosystems have more than a hundred persons developing their new UAV project: an autonomous helicopter called “Skeldar”. This project is not only military, it is also possible to use Skeldar for civilian purposes, such as traffic surveillance or inspection of power lines.

At Georgia Institute of Technology, Atlanta, extensive research is done in different civilian applications for UAVs. At the moment they are developing a first responder for preliminary suveys of an area to provide situation overview for policemen, ambulance crew or fire fighters before they arrive at the scene. The helicopter could for example give fire fighters valuable information about the fire place, and thereby minimizing the damage and maybe even save human lives.

The above mentioned applications are only a few examples of what UAVs may be used for in the future. This UAV project has its base in farming.

1.2 Helicopter Aided Mapping of Crops

Research Centre Foulum, Danish Institute of Agricultural Sciences, is doing extensive research in efficient and environmentally safe farming. A key part of this research is intimate knowledge of growth conditions of crops. Today this knowledge is obtained in a cumbersome and expensive way, using satellite or aircraft imagery or mobile ground based sensor systems. However, much better research data can be obtained by taking multispectral images of the field from a helicopter. Because of the manoeuvrability of the helicopter, it will be possible to take many small images of a field, and combine them to one big image of the whole field afterwards. In that way, the resolution of the whole field image can be increased by decreasing the altitude in which the helicopter is photographing, because this reduces the area covered by each image. The price will be an increased number of pictures to combine. Furthermore, it is possible to gather the imagery within a very limited time span and at arbitrary times of the day. This means that it is possible to gather the imagery of the same field at different times of year (e.g. each day), while maintaining the same angle of sunlight on each image. Gathering image data of a field every day throughout a growing season with the same angle of sunlight, will give very precise data of the crop growth. This application has been named Helicopter Aided Mapping Of Crops (HAMOC).

1.3 Initiating Problem

The idea of this project is to use a small model helicopter for mapping of crops. This helicopter must be able to carry out the whole operation from take-off to landing without human interaction. A ground station connected to the helicopter by a wireless link serves as a user interface. This means that the procedure for an operation can look like this:

1. The operator calculates the desired time and date of the operation.
2. The operator enters the exact boundaries of the field in GPS co-ordinates, as well as the desired altitude from which the images are to be taken, into a ground station computer.
3. The ground station generates photographing locations and a trajectory for the helicopter to follow.
4. The operator initiates the operation by turning on the helicopter at the desired time and date.
5. The ground station connects to the helicopter, and the trajectory information is uploaded.
6. The helicopter takes off and starts taking pictures.
7. On the ground station the operator must be able to follow the progress of the operation on-line. For example, the screen can show the whole field and be continuously updated when images are relayed to the ground station.
8. When done and all the images are transferred to the ground station, the helicopter lands at the take off point.

9. The ground station stitch the images together to create one image of the whole field.
10. The operator turns off the helicopter, and can now analyze the entire image saved by the ground station.
11. For security reasons, it must be possible for the operator at all times to both cancel the mission using the ground station, and turn off the helicopter.

The final project goal is therefore to obtain a fully autonomous system to gather imagery of a field. This can be accomplished by dividing the project into several different sub blocks, which have to be considered.

Hardware implementation on the helicopter: The helicopter has to carry an on-board computer as well as a camera. Different sensors are needed, e.g. accelerometers, magnetometer, gyroscopes, etc. It must also be considered how to ensure stable power supply to the hardware on the helicopter.

Trajectory generation: When the field boundary is known, the system must be able to calculate a trajectory of flight, and the positions where to take the images.

Modelling, state estimation and control: In order to make the helicopter follow the calculated trajectory autonomously with a given precision, some kind of feedback control will have to be implemented. Due to this, it is necessary to implement different sensors on the helicopter, to estimate the current state, and to have a model of the helicopter, i.e. a mathematical description of the input-output relations and cross-couplings.

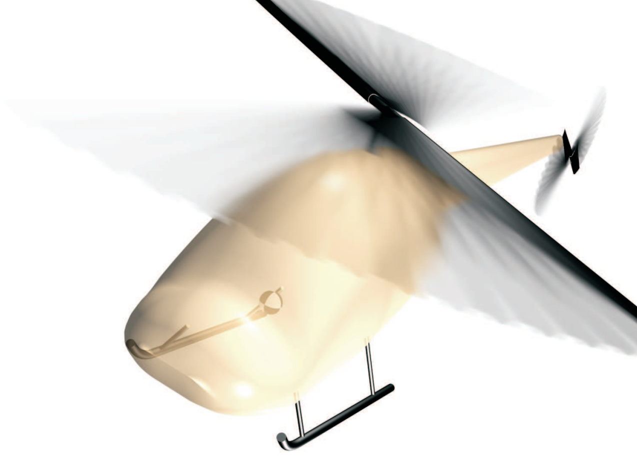
Position determination: It is necessary to know the absolute position of the helicopter relative to earth, as the helicopter must be able to follow the given trajectory. Furthermore, the helicopter has to take images from certain calculated points above the field. A GPS may be used for this purpose.

Image processing: To get one high resolution image of the whole field as wanted, it is necessary to splice all the small images, taken by the helicopter. This may be done off-line, when the helicopter has gathered all imagery of the field.

Ground station: The ground station will function as the user interface to the system, enabling the user to set up and start the image gathering. It also gives the user information about the status of the helicopter during operation.

Communication: The ground station must be able to communicate both ways. Before take-off, data must be uploaded to the on-board computer, and while flying the user must be able to follow the behaviour of the helicopter and to switch to manual control, if necessary (e.g. for safety reasons).

This project will be the first phase of reaching a fully operational HAMOC system. Not all the mentioned tasks will be considered in this report. The next chapter elaborates on the specific framework for this master project.



2

PROJECT FRAMEWORK

Chapter Contents

In this chapter, the framework of this master project is established. Some former helicopter projects at Aalborg University are presented in order to determine the starting point for the HAMOC project. Then the goal and limitation of this specific project is defined and some specific objectives are outlined.

2.1 UAVs at Aalborg University

In the previous two years, research and study projects in developing autonomous helicopters has been ongoing at the Department of Electronic Systems, Automation and Control. The helicopter acquired for the present project is the third.
The first helicopter, a Futura SE Model (Figure 2.1) was used in 2004/2005 for two master projects. One project [Mustafic et al., 2005] modelled the helicopter using first principles modelling techniques. The model was never verified in test flight, but a non-linear controller (feedback linearization) was implemented in simulation. It showed impossible to stabilize the model with this controller structure. The other project [Jensen and Nielsen, 2005] was focusing mainly on robust control, developing two different types of controllers and comparing their performance and robustness. This project succeeded in stabilizing the helicopter in hover simulation.

Figure 2.1: *The Futura SE helicopter. (Photo: Anders la Cour-Harbo)*

In 2005 the Futura SE Model was replaced by a Bergen Industrial Twin Model (Figure 2.2). This helicopter was used in an 8th semester project in the spring 2005, which led to two master projects in 2005/2006. The intention with the master projects was to develop a UAV for participation in the International Aerial Robotics Competition in Georgia, summer 2006 (IARC06). One project [Hald et al., 2006] focused on modelling and control. Again, a first-principles modelling technique was used, and the model was implemented fully in C and linearized in hover equilibrium. The controller was able to stabilize the model in a close range around hover, but only in a simulation environment. The controller was not implemented and tested on the Bergen helicopter.
The aim of the other master project [Holmgaard et al., 2006] was implementation of the hardware and software platform, as well as developing the navigation system for the UAV. The hardware and software platform was tested in test flights, and it showed fully operational. An optimal flight path was calculated on the basis of a map of the IARC area, and using GPS the navigation system was able to follow this path.
As there were still a lot of work to do before the UAV is operational, the Bergen helicopter did not enroll in IARC this time. However, the work was not in vain, as parts of it is utilized in an ongoing Ph.D. project. Since 2004 Ph.D.-student Morten Bisgaard has been working with modelling and controlling a helicopter with a slung load [Bisgaard, 2005]. The application in his project is mine detection, using a UAV with a mine detection device hanging underneath the helicopter, and thereby minimizing the risc of mines detonating

Figure 2.2: The Bergen Industrial Twin helicopter (Photo: Troels Lund).

under operation. During this project, the model for the Bergen helicopter has been further developed and implemented in the simulation environment, using S-functions in Simulink supported by 3D-graphics using irrlicht. Furthermore, a camera is mounted on the helicopter with the lens pointing vertically down. This is used for position detection of the slung load.
Besides from these projects directly involving the helicopters, there are a number of projects, which have concerned different aspects of UAV theory and practice, concurrently. The communication protocol between the UAV and a ground station is one example. Furthermore, there has been projects dealing with other kinds of UAVs, e.g. autonomous airplanes and a so called "Four Rotor Dragan Flyer".

2.2 Foundation for this Project

It is apparent, that there is a solid foundation in UAV research at the department, which can be utilized in this project. This foundation comprises the following aspects:

- Modelling: a model for the Bergen helicopter is designed and implemented in Simulink using S-functions. This model will be a basis for the helicopter model used in this project [Bisgaard, 2005], [Hald et al., 2006].
- Simulation environment: a popup window visualizes the helicopter in 3 dimensions, when running a simulation in Simulink. This is useful when the controllers are tested, not for documenting the test, but to get a visual idea of how the helicopter behaves [Bisgaard, 2005].
- Hardware: even though the hardware setup on the helicopter in this project is not the same as on the Bergen helicopter platform, there will be similarities which can be re-used, and a lot of experience about what not to do.
- Sensors: knowledge of the different types of sensors will be valuable in this project. The implemented sensors on the Bergen helicopter are: an IMU, a magnetometer and a GPS.
- Image processing: the algorithm used for slung load detection in Bisgaard [2005] can serve as a basis for position detection of the helicopter in this project.

- Trajectory generation: the navigation system on the Bergen helicopter may be reused for calculating an optimal flight path for the helicopter in this project.

Last, the fact that extensive work with UAVs has been conducted in the previous years at Dept. of Electronic Systems has provided valuable practical knowledge and knowhow. A disadvantage with the two first helicopters, were that they could not be tested in the lab. Due to the size of the new electric helicopter, this has become possible, and it significantly decreases the time needed for testing. Despite this fact, it is not possible for two persons to complete all the tasks mentioned in the initiating problem in nine months. To keep the project manageable, it is necessary to omit different parts of the project in favour of others.

2.3 Project Delimitation

Figure 2.3: *The Corona-LMH 120 electric helicopter (Photo: Lars Horn).*

For the purpose of this project a Corona 120 electric model helicopter from *LiteMachines* has been bought (Figure 2.3). The final goal is formulated as follows:

The purpose of this project is to make a small-scale electric helicopter capable of hovering autonomously in the laboratory.

This goal will be a major achievement on the way to a fully operational HAMOC, due to the fact that a helicopter is a highly complex (non-linear) system, which has an unstable equilibrium point in hover. Using a small lightweight helicopter with fast dynamics does not make the task easier. It is therefore also decided *not* to spend time implementing an on-board computer and power supply on the helicopter. An external computer and power supply will be used, meaning that external cables will actually have to be connected to the helicopter, when active. This procedure will also make it easy to test new software. Further, when a fully operational system is obtained, the software should be easily portable to an on-board computer.

2.4 Objectives

To fulfil the goal, a number of tasks must be accomplished. These tasks have been grouped in four different objectives, listed below. It has been given high priority to develop all systems instead of producing exhaustive solutions to each system.

Objective 1: Hardware Implementation The hardware platform for the system must be designed and implemented so that the helicopter is fully operational. This objective includes the following tasks:

1.1 Consider hardware system structure.
1.2 Sensor selection and implementation.
1.3 Implement actuator controlling devices.
1.4 Set up system computers.
1.5 Choose and design interfaces between hardware parts.
1.6 Provide proper power supply for the whole system.

Objective 2: Modelling A model for the Corona 120 helicopter must be developed. To do this, the existing model for the Bergen Twin is used as basis. This leaves three tasks to be done:

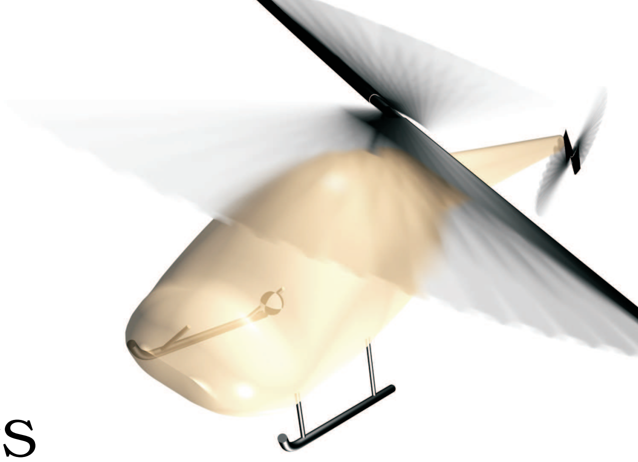
2.1 Adapt the existing model to the Corona 120. The main difference from the Bergen model is that the rotor rotates the opposite way.
2.2 Verify the model. Tests will be carried out to show that the obtained model is applicable for the Corona 120.
2.3 Measure or estimate the parameters for the Corona 120. Directly measurable quantities such as masses and lengths will be measured using appropriate instruments, and all other parameters are calculated or estimated, based on some test setup.

Objective 3: State Estimation To be able to control the helicopter, it is necessary to estimate the states. Instead of using GPS as a position sensor, the position and attitude is estimated by means of image processing of the imagery from webcameras filming the helicopter. By tracking markers on the helicopter from two different and known points in space, it should be possible to determine the position and attitude of the helicopter (as each camera has a two dimensional image as output). The HAMOC application has a camera mounted on the helicopter, pointing downwards for taking images of the crops. In this project the helicopter camera will be dedicated for increasing the accuracy of the position and attitude output. This objective includes the following tasks:

3.1 Design markers to mount on the helicopter and the floor.
3.2 Develop image processing software to track the markers.
3.3 Combine the redundant information from three cameras to one position and attitude estimate.
3.4 Combine this estimate with IMU information to obtain a state estimate.

Objective 4: Control Design and Implementation A controller has to be developed for hover. With this controller implemented, the helicopter must remain steady in hover, suppressing model errors and disturbances. The tasks of this objective are:

4.1 Investigate different controllers and design a feasible controller for hover.
4.2 Implement and test the controller on the model in the simulation environment.
4.3 Test the controller on the real helicopter.
4.4 Investigate and design advanced controllers.



3

Definition of Concepts

Chapter Contents

In this chapter all the concepts used in this report are introduced. Basically the same definitions, concepts and notation as in [Hald et al., 2006] are used.

3.1 Typographic Conventions

Symbol	Rule	Example
Scalar	Upper- or lowercase, italic	a, A
Frame denotation in text	Uppercase, italic, in curly brackets	$\{A\}$
Vector	Upper- or lowercase, bold	$\boldsymbol{v}$, $\boldsymbol{F}$
Vector in frame $\{A\}$	The frame designation as superscript before the vector	${}^{A}\boldsymbol{v}$
Matrix	Uppercase, calligraphed	$\mathcal{M}$
Rotation matrix from $\{A\}$ to $\{B\}$	A as subscript and B as superscript before the matrix	${}^{B}_{A}\mathcal{R}$

Table 3.1: *Table of typographic standards.*

For more intuitive readability of figures, torque vectors are drawn as the direction of the resulting force around a point, instead of the torque vector itself (see Figure 3.1).

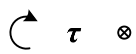


Figure 3.1: *The left arrow arc illustrates how the right torque vector τ (pointing into the paper) is illustrated in figures.*

3.2 Helicopter Concepts

In the following subsections different general terms and concepts will be introduced.

3.2.1 Frame Definitions

Four different frames are used to describe the model of the helicopter. All frames are defined using a right-hand three dimensional Cartesian co-ordinate system. Figure 3.2 and Figure 3.3 show all the frames described in this section.

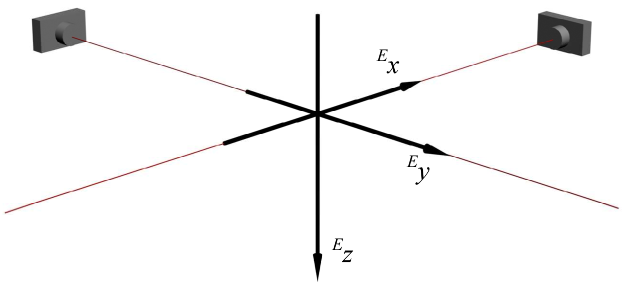


Figure 3.2: *The earth frame is aligned with the center axes of the frontcam and sidecam.*

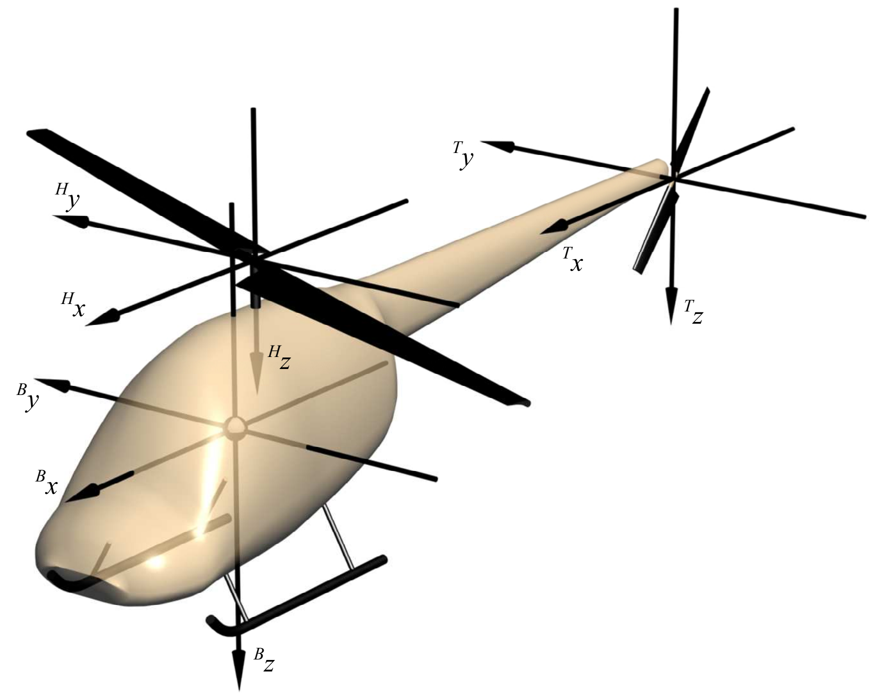


Figure 3.3: *The frames attached to the helicopter are the body frame, the hub frame and the tail rotor frame.*

The Earth Frame $\{E\}$ is used as reference when describing the position of the helicopter. The origin is located in the point where the two center axes of the frontcam and sidecam are intersecting. This point is where the CM of the helicopter is supposed to be located when hovering. The ^{E}x-axis is coinciding with the center axis of the frontcam, and pointing towards the camera. The ^{E}y-axis is coinciding with the center axis of the sidecam and pointing away from the camera. The ^{E}z-axis is pointing downwards. The ground is assumed to be perfectly level and horizontal within the range of the helicopter.

The Body Frame $\{B\}$ has its origin in the CM of the helicopter, and follows the fuselage (the body) of the helicopter. When the helicopter is standing on level ground, the ^{B}x-axis is horizontal and parallel with the metal frame/plate dividing the helicopter in a left and a right side. It is pointing towards the front of the helicopter. The ^{B}y-axis is horizontal and pointing towards the right-hand side of the helicopter. The ^{B}z-axis is pointing downwards. The position of the helicopter is given as the body frame origin position in the earth frame, and the attitude of the helicopter is given as the rotation of the body frame relative to the earth frame in 3-2-1 euler angles. Notice that when the helicopter is hovering steady, the position of the helicopter should be $^{E}\mathbf{\Xi}_{hover} \approx [0\ 0\ 0]^T$, and the axes of $\{B\}$ and $\{E\}$ will point in approximately the same direction, $^{E}\mathbf{\Theta}_{hover} \approx [0\ 0\ 0]^T$. The exact position of the helicopter where $\{B\}$ and $\{E\}$ are coinciding, will be denoted as *the zero position*.

The Hub Frame $\{H\}$ is used to describe the main rotor blade motion. It is oriented as the body frame, and it has origin in the main rotor rotation shaft, and in a height corresponding to where the blades are mounted in the hinge.

Tail Rotor Frame $\{T\}$ is used to describe the tail rotor blade motion. It is oriented as the body frame and has its origin in the center of the tail rotor rotation.

Lateral Plane is a plane parallel to the ${}^{B}y{}^{B}z$-plane.

Longitudinal Plane is a plane parallel to the ${}^{B}x{}^{B}z$-plane.

3.2.2 Attitude Definitions

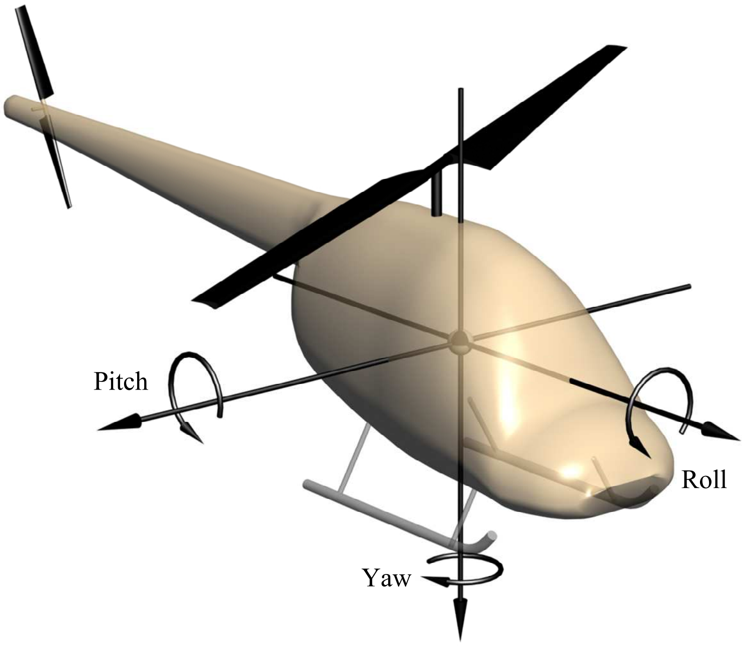


Figure 3.4: *Definition of roll, pitch and yaw directions.*

Roll describes the angular motion about the ${}^{B}x$-axis. Positive roll motion is a clockwise rotation when the helicopter is seen from the rear. A roll denotes an angle, and a roll motion denotes an angular velocity.

Pitch is the angular motion about the ${}^{B}y$-axis. Positive pitch is a clockwise rotation when the helicopter is seen from the left. A pitch denotes an angle, and a pitch motion denotes an angular velocity.

Yaw is the angular motion about the ${}^{B}z$-axis. Positive yaw is a clockwise rotation when the helicopter is seen from above. A yaw denotes an angle, and a yaw motion denotes an angular velocity.

3.2.3 Actuator Terms

A helicopter is controlled by means of four different input, three input to control the main rotor and one to control the tail rotor. On a remote controlled model helicopter, these input are actuated through servo motors and DC motors.

Rotor Velocity Normally a helicopter has a fixed angular velocity of the main and tail rotor. A speed controller (called a governor) is taking care of maintaining this fixed angular velocity and suppressing disturbances like aerodynamic drag on the rotor blades. On some small scale model helicopters, there is no governor, and the angular velocity is used to control the lift thrust from the main rotor directly. This is the case in this project, where the angular velocity is called $\dot{\Psi}$. The main rotor is powered by a DC motor.

Collective pitch On a helicopter, lift thrust is normally controlled by pitching all the main rotor blades equally. This is called collective pitch. However, some small scale helicopters has fixed collective pitch, and then the lift thrust is controlled by the angular velocity of the main rotor. This is the case in this project.

Cyclic pitch The lift thrust vector can be tilted, such that it is not parallel with the main rotor shaft axis. This is done by making the pitch of the main rotor blades depend on the angular position of the blade, Ψ. This is called cyclic pitch, as the rotor blade pitch is a periodic function of the position of the blade with a 2π cycle. Two input are necessary to control cyclic pitch, namely lateral pitch and longitudinal pitch. By combining these two input it is possible to tilt the lift thrust vector towards any direction. The actuators controlling cyclic pitch are servo motors.

Lateral pitch is the pitch of the main rotor blades making the helicopter perform a roll motion. Due to a 90° phase lag, the extremum of the lateral pitch angle is actually where the blade is parallel to the ${}^{B}x$-axis.

Longitudinal pitch is the pitch of the main rotor blades making the helicopter fly forwards or backwards. Due to a 90° phase lag, the extremum of the longitudinal pitch angle is actually where the blade is parallel to the ${}^{B}y$-axis.

Tail Pitch The tail rotor thrust is controlled by applying collective pitch to the tail rotor. A servo motor similar to the cyclic pitch actuators is controlling the tail pitch.

3.2.4 Main Rotor Terms

Some of the important terms regarding the rotor hub is explained in this section.

Blade position is called Ψ. For $\Psi = 0\,\mathrm{rad}$ the blade is parallel to the ${}^{B}x$-axis and pointing towards the tail of the helicopter. The angular position increases when the blade is rotating clockwise as seen from above. The Corona 120 has a counterclockwise main rotor direction, so $\dot{\Psi}$ is negative, as illustrated in Figure 3.5.

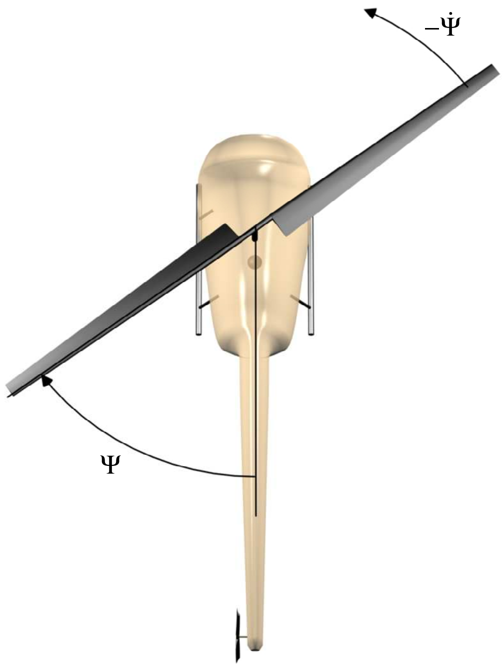


Figure 3.5: Definition of main rotor blade position.

Flapping angle The main rotor blades does not only perform a rotation around the hub. They also bend up and down depending on the angular velocity and the pitch of the blade. This motion is called flapping, and it occurs because the rotor blades are not totally rigid. The flapping motion is very important to describe, as it is because of the flapping that the pilot is able to control the helicopter. However, the true flapping is to complex to describe mathematically, so it is simplified to variable, β, which is the angle in a virtual hinge on the blade, and the blade is modelled as stiff (rigid). The virtual hinge has a spring force constant, K_s, such that β has a stable equilibrium in 0 rad. The simplified principle is illustrated in Figure 3.6.

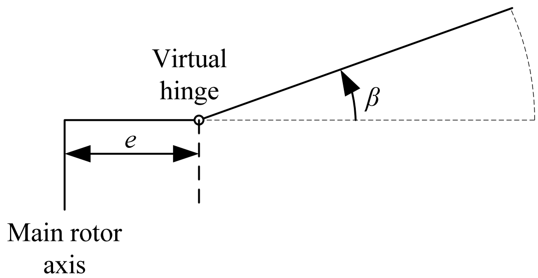


Figure 3.6: The definition of the flapping angle β. The rotor blade is modelled having a virtual hinge with a spring constant. This hinge is located at distance e from the hub.

Coning When a main rotor blade rotates a full revolution in hover, the area that it sweeps is shaped like a cone, as illustrated in Figure 3.7. This is due to the flapping of the blade, and because gravity is pulling down in the helicopter.

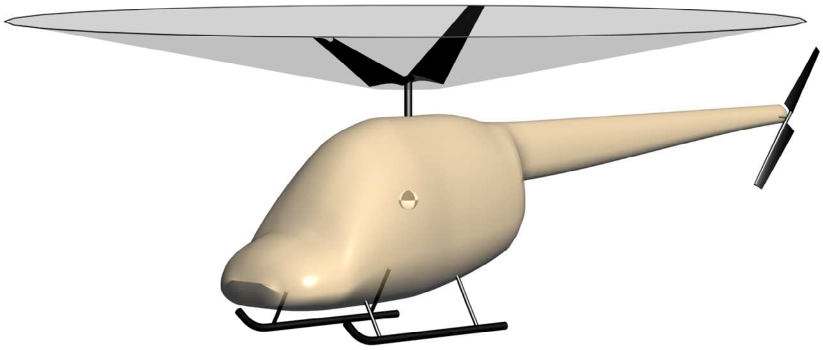


Figure 3.7: Due to flapping the area swept by a rotating main rotor blade is not necessarily flat. Taking the virtual flapping hinge into considerations, the area may be shaped as a cone.

Hub Plane (HB) is the plane perpendicular to the main rotor shaft axis and positioned where the rotor blades are attached to the shaft.

Tip Path Plane (TPP) is defined as the plane spanned by the tips of the blades when they rotate, and it is also the plane having the lift thrust vector as normal. When there is no flapping, the TPP is coinciding with HP.

Swash plate The swash plate is the mechanism which transfers the collective and cyclic pitch input from the actuators to the main rotor blades. It consists of a non-rotating plate attached to the body of the helicopter, and a rotating plate attached to main rotor hub. The two plates are connected to each other through a ball bearing, such that they are always parallel. When tilting the non-rotating plate by applying cyclic pitch, the rotating plate will tilt the same angle, and the cyclic pitch input is then transferred to the rotor blades. See Figure 3.8.

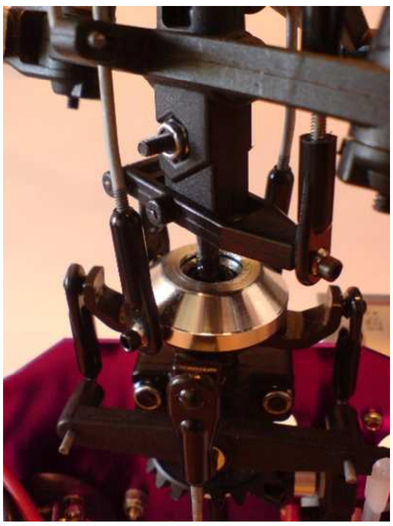


Figure 3.8: The swash plate on the Corona 120.

Stabilizer bar The smaller a helicopter is, the faster the dynamics of the helicopter motion becomes. On most small scale helicopters, it is necessary to slow down the dynamics, since it is too fast for the pilot to control. This is done by means of a stabilizer bar, see Figure 3.9. Through the mechanical construction, the stabilizer bar provides some fraction of the cyclic pitch to the main rotor blades. This contribution depends on the flapping of the stabilizer bar. Due to the inertia in the stabilizer bar, it works like a negative feedback system, thus counteracting the fast dynamics of the main rotor.

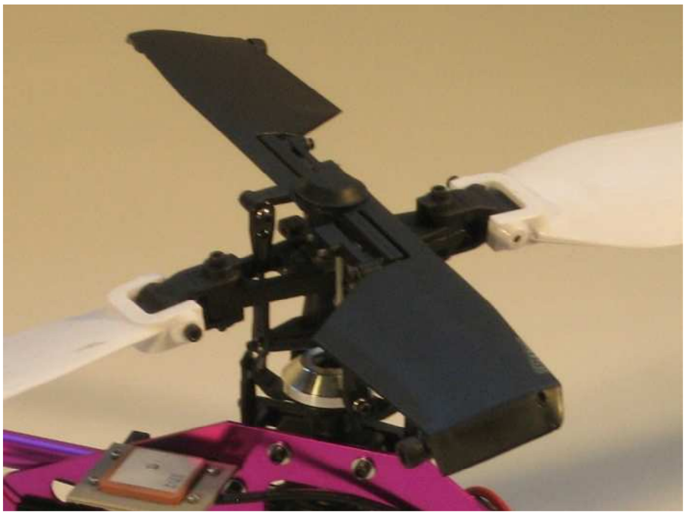


Figure 3.9: *The stabilizer bar on the Corona 120.*

Bell/Hiller gain The fraction of cyclic pitch provided by the swash plate directly is called the Bell gain, and the fraction provided by the stabilizer bar is called the Hiller gain. The Bell/Hiller gain depends on the mechanical construction of the hub. Figure 3.10 shows a blockdiagram of the Bell/Hiller gain concepts.

3.2.5 Hub Angle Definitions

Figure 3.11 shows the hub and the different angles related to this. All angles are defined with respect to the body frame, and they are following the right-hand rule. In Table 3.2 the angles are described further.

Variable	Plane	Angle definition	Direction
A_1	Lateral	Swash plate	From swash plate to a line perpendicular to the rotor shaft axis
B_1	Longitudinal	Swash plate	From swash plate to a line perpendicular to the rotor shaft axis
a_1	Longitudinal	Blade flapping	From hub plane to tip path plane
b_1	Lateral	Blade flapping	From hub plane to tip path plane
$a_{1,sb}$	Longitudinal	Stabilizer bar flapping	From hub plane to tip path plane
$b_{1,sb}$	Lateral	Stabilizer bar flapping	From hub plane to tip path plane

Table 3.2: *Definitions of angles related to the hub (see Figure 3.11).*

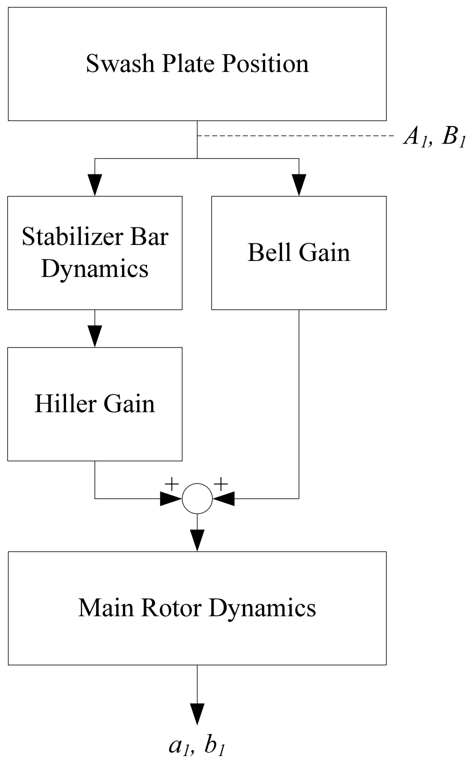
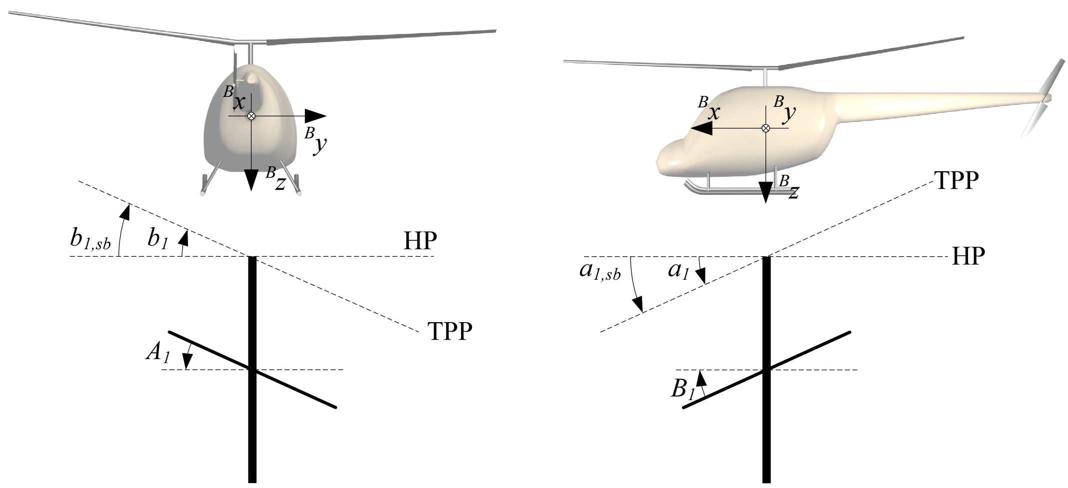


Figure 3.10: *Blockdiagram showing how the main rotor blade pitch is a combination of the direct input from the swash plate, and the flapping of the stabilizer bar.* A_1*,* B_1*,* a_1 *and* b_1 *are defined in the next section.*



(a) The helidopter seen from the rear end. Note that in this position A_1 is negative due to the orientation of the arrow.

(b) The helicopter seen from the left-hand side. Note that in this position a_1 and $a_{1,sb}$ is negative due to the orientation of the arrows.

Figure 3.11: *Illustration of hub angle directions. See Table 3.2 for explanation of variables.*

3.2.6 Cyclic Input Effects

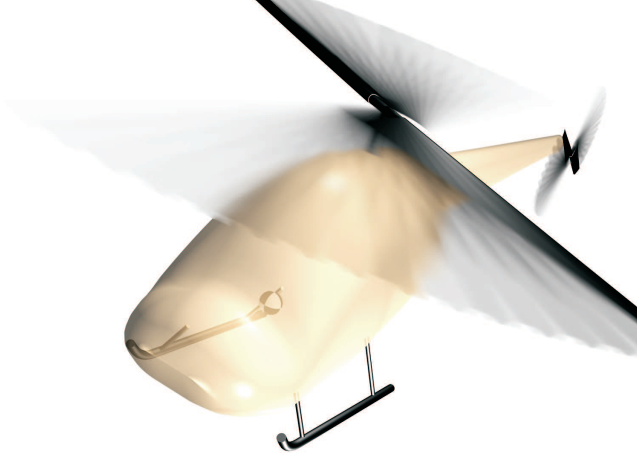
Table 3.3 shows how the helicopter reacts to different cyclic input. The first column is the cyclic input, S_{lat} and S_{lon}. Second column shows the swash plate angles, A_1 and B_1. Third column shows at what position the main rotor pitch angle, θ_b, reaches its maximum value. Fourth column shows at what position the flapping angle, β is maximum. Note that β is always maximum 90° after the pitch angle was maximum. Fifth column relates the two flapping angle variables, a_1 and b_1, to the input. Sixth column gives the resulting attitude motion, and the last column gives the resulting translatory velocity.

Cyclic input	**Swash plate angle**	θ_b **max at**	β **max at**	**Flapping**	**Attitude motion**	**Translatory motion**
$S_{lat} > 0$ $S_{lon} = 0$	$A_1 > 0$ $B_1 = 0$	$\Psi = 0°$	$\Psi = 270°$	$b_1 < 0$ $a_1 = 0$	${}^B\dot{\phi} < 0$	${}^B\dot{y} < 0$
$S_{lat} < 0$ $S_{lon} = 0$	$A_1 < 0$ $B_1 = 0$	$\Psi = 180°$	$\Psi = 90°$	$b_1 > 0$ $a_1 = 0$	${}^B\dot{\phi} > 0$	${}^B\dot{y} > 0$
$S_{lat} = 0$ $S_{lon} > 0$	$A_1 = 0$ $B_1 > 0$	$\Psi = 90°$	$\Psi = 0°$	$b_1 = 0$ $a_1 < 0$	${}^B\dot{\theta} < 0$	${}^B\dot{x} > 0$
$S_{lat} = 0$ $S_{lon} < 0$	$A_1 = 0$ $B_1 < 0$	$\Psi = 270°$	$\Psi = 180°$	$b_1 = 0$ $a_1 > 0$	${}^B\dot{\theta} > 0$	${}^B\dot{x} < 0$

Table 3.3: *Table of various effects, due to certain cyclic input.*

4

HARDWARE



Chapter Contents

In this chapter the overall hardware setup is described. The chapter begins with an overview of the whole system, after which a detailed explanation of each block, and their interfaces is given.

4.1 System Overview

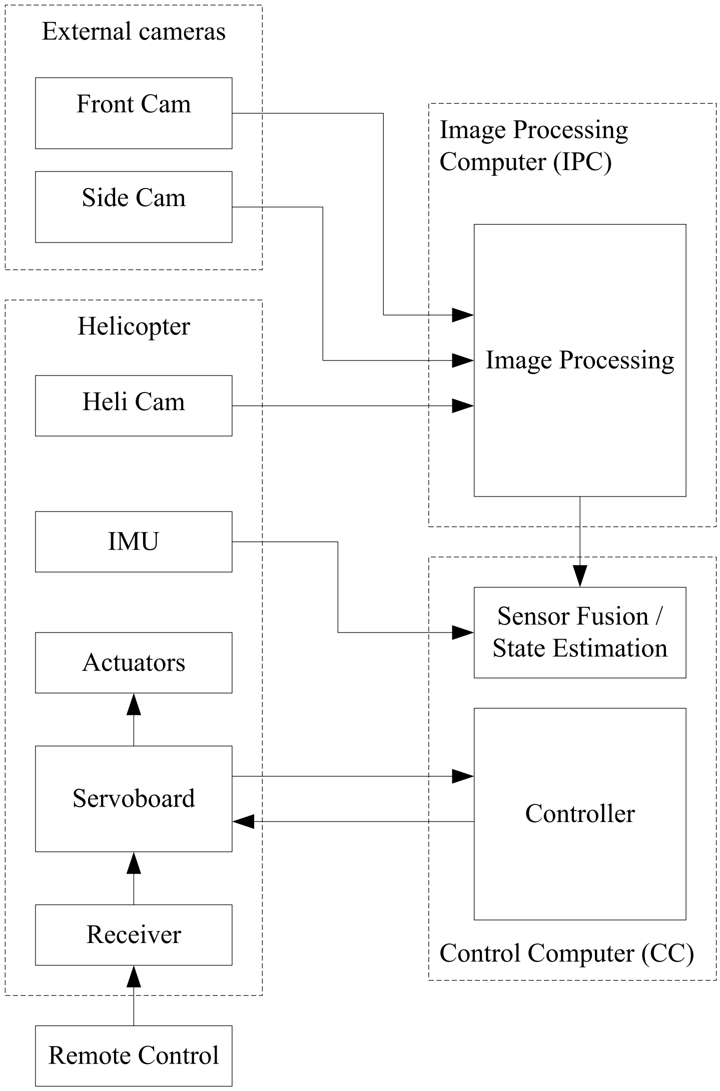


Figure 4.1: *An overview of the whole system.*

It is chosen to divide the system in four main blocks:

- The helicopter with sensor and actuator hardware implemented: inertial measurement unit (IMU), on-board helicopter camera (helicam) and servoboard.
- The two external cameras.
- The image processing computer (IPC).
- The control computer (CC).

Figure 4.1 shows these four blocks, and how they are interconnected. It is chosen to exclude the remote control from the main system, as it is not a part of the feedback loop, when the helicopter is flying autonomously. However, it is necessary when controlling the helicopter manually. The switch between manual and autonomous flight is controlled by a switch on the remote control, as depicted on Figure 4.2.

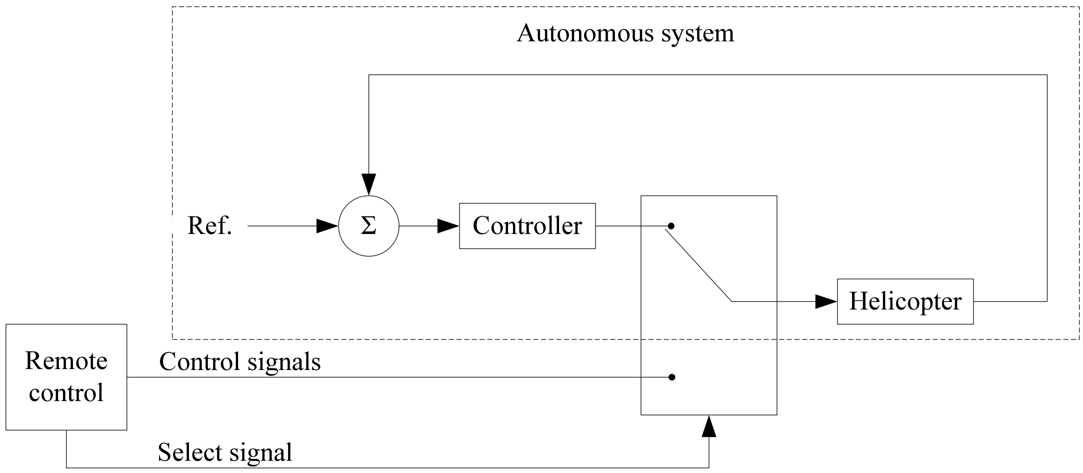


Figure 4.2: *A schematic of how the remote control is used to switch between manual and autonomous flight.*

4.2 Camera Setup

Position determination of the helicopter is enabled by the use of three cameras. One camera is mounted on the helicopter - this is called the helicam (denoted as $(\cdot)_{hcam}$ in equations). Two external cameras are mounted perpendicular to each other, as shown in Figure 4.3. These are called frontcam and sidecam (denoted as $(\cdot)_{fcam}$ and $(\cdot)_{scam}$ in equations, respectively). The main reason for choosing this type of position sensor, is that it does not require any hardware considerations at all, we just have to attach the helicam to the helicopter, the frontcam and sidecam to the wall, and then place markers on the helicopter and the floor. Furthermore, the algorithm tracking the markers has been developed, and should be feasible for this application with only a little adaptation. In addition, the accuracy of the measured position is expected to be high, compared to the simplicity and price of the camera setup. Figure 4.4 shows the lab, with the cameras mounted on the wall.

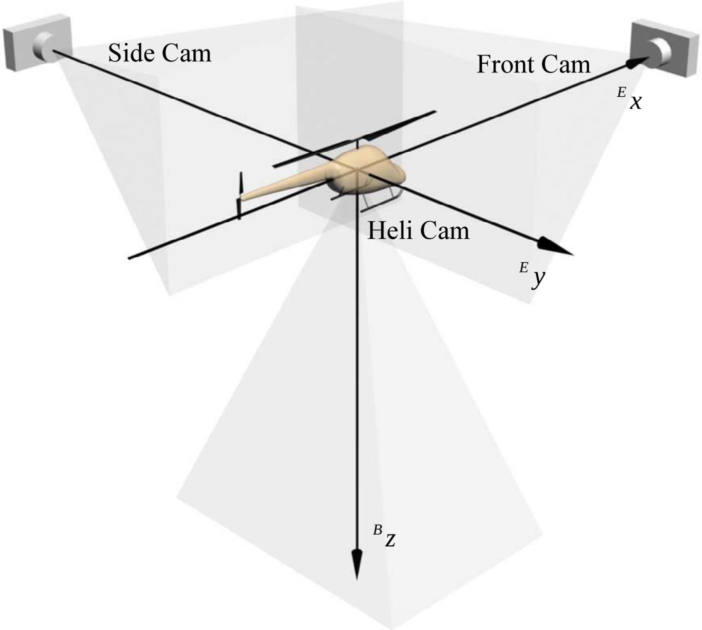


Figure 4.3: The camera setup. The two external cameras (front- and sidecam) are pointing in directions perpendicular to each other. The camera mounted on the helicopter is pointing in the $^B z$ direction. The directions of the three cameras are then perpendicular to each other, when the helicopter is hovering in the zero position.

Figure 4.4: The lab where the flight tests take place. The computers controlling the helicopter is seen in the right side. The pilot is placed to the left, such that the helicopter has the rear end towards him. The frontcam is mounted on the wall in front of the helicopter, and the sidecam is mounted on the wall below the window.

The helicam is pointing in the direction of the $^B z$-axis, such that it is filming vertically downwards when the helicopter is in hover. The two external cameras are placed perpendicular to each other, and pointing horizontally towards a centerspot, where the origin of the body frame is supposed to be situated in hover. That is, when the helicopter is in

steady hover, the three cameras will be pointing in directions perpendicular to each other, and the two external cameras will have the helicopter in view. In this position, $\{E\}$ and $\{B\}$ will be coinciding, and the helicopter is in zero position. In $\{E\}$, the position of the frontcam and sidecam are as follows:
Frontcam:

$$^{E}\boldsymbol{\kappa}_{fcam} = \begin{bmatrix} 1{,}76 & 0 & 0 \end{bmatrix}^T \qquad [\,\mathrm{m}] \qquad ,$$

Sidecam:

$$^{E}\boldsymbol{\kappa}_{scam} = \begin{bmatrix} 0 & -1{,}46 & 0 \end{bmatrix}^T \qquad [\,\mathrm{m}] \qquad .$$

The helicam position is given in the body frame:

$$^{B}\boldsymbol{\kappa}_{hcam} = \begin{bmatrix} -0{,}08 & -0{,}01 & 0{,}06 \end{bmatrix}^T \qquad [\,\mathrm{m}] \qquad .$$

The floor level is a plane parallel to the $^{E}x^{E}y$-plane in $^{E}z = 102\,\mathrm{cm}$. This value will be denoted as z_{floor}.
All the cameras are conventional USB web-cams, recording compressed video with a frame rate up to 30 frames pr. second. The frontcam and sidecam are a Philips SPC900, having a resolution of 640×480 pixels, and the helicam is a Philips SPC610NC, having a resolution of 320×240 pixels.
The intention with this setup is to provide sensor information of both the helicopters absolute position as well as it's attitude. This information can then be fusioned with the data from the IMU to gain a higher accuracy on the measurements. This is done by placing markers on the helicopter and the ground beneath it, and then by means of image processing determine the position of these marks.

4.3 The Helicopter

The acquired helicopter is a Corona 120 electric powered model helicopter from LiteMachines. Figure 4.5 shows an image of the out-of-the-box assembled helicopter.

Figure 4.5: *The Corona 120 model helicopter.*

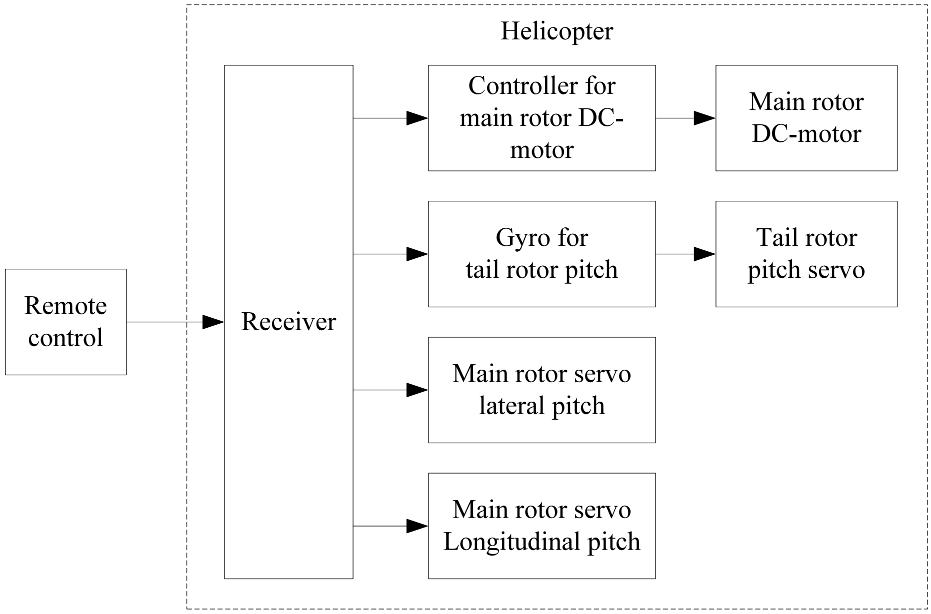


Figure 4.6: *Block diagram of the helicopter.*

Figure 4.6 shows a blockdiagram of the electronical parts of the helicopter. It has 4 actuators:

- A DC-motor for the main rotor.
- A servo for the longitudinal pitch of the main rotor.
- A servo for the lateral pitch of the main rotor.
- A servo for the collective pitch of the tail rotor.

In manual flight, these actuators are controlled by the receiver on the helicopter, which again is controlled by the remote control. The left joystick longitudinal motion controls main rotor angular velocity, and the lateral motion controls the tail rotor pitch. The right joystick longitudinal motion controls longitudinal pitch of the main rotor, and the lateral motion controls lateral pitch of the main rotor. This is an underactuated system controlling the six degrees of freedom (6-DOF) motion of the helicopter, as seen on Figure 4.7.

In between the main rotor DC-motor and the receiver, a controller is taking care of converting the PWM signal from the receiver to a control signal for the motor. Further, the controller acts as a 5 V voltage regulator for power supply to the rest of the system.

Due to the fast dynamics of the yaw motion of the helicopter, a one axis gyro is implemented by the manufacturer on the helicopter to control the tail rotor pitch. In that way, the pilot controls the yaw rate, rather than the pitch directly.

The power supply for the helicopter is a battery pack, which can only last for a few minutes. For the use in this project, it has been replaced with an external power supply as written in the delimitation in Section 2.3 on page 10. Besides from the longer lasting power, it also reduces the weight of the helicopter significantly and makes room for attaching the necessary hardware to the helicopter frame. The outer jacket of the two

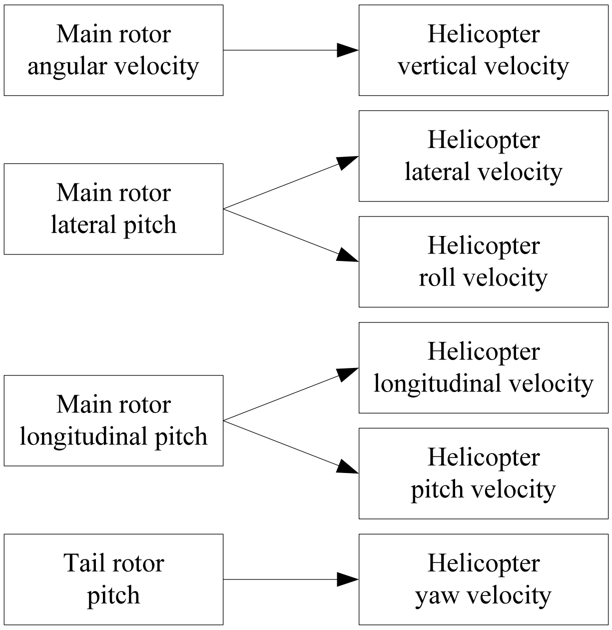


Figure 4.7: *Diagram showing how the actuators are affecting the velocity and attitude of the helicopter. Cross couplings are not shown.*

power cables are made of soft silicone to keep them flexible, and thereby minimizing the disturbance when flying.

4.3.1 Servoboard

In order to make the helicopter fly autonomously it is necessary to alter the system, such that a computer can control the actuators. For this purpose a servoboard has been designed. A block diagram of this is depicted in Figure 4.8.

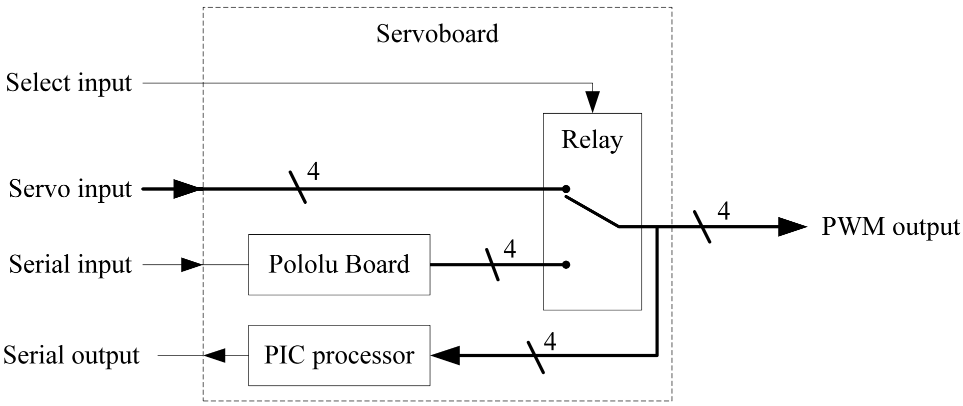


Figure 4.8: *Block diagram of the servoboard.*

The servoboard takes a serial input and converts it to PWM output signals for the three servo motors and the main motor controller. This is done by the *Pololu Micro Serial Servo Controller* [Pololu, 2005] acquired from Pololu Robotics and Electronics. Furthermore, a PIC-processor is doing the reverse processing - converting the PWM output signals to a serial signal, which is transmitted back to the computer. This serial signal can be used to

read the output of the servoboard, regardless of whether the helicopter is in manual og autonomous mode. This is useful when testing the system.
The switching from manual to autonomous flight is taken care of by a relay connected to a select input on the receiver. This input is controlled by a switch on the remote control. The switch is chosen to be one of the spring loaded buttons, such that the pilot must actively press and hold the button during autonomous flight. When the button is released, the helicopter is controlled manually again.
A schematic diagram of the print layout of the servoboard can be found at [CD-ROM, 2007, `literature/hardware/servoboard_diagram`], and the serial protocol is defined in [CD-ROM, 2007, `literature/hardware/servoboard_protocol`].

4.3.2 Inertial Measurement Unit

The inertial measurement unit (IMU) is acquired from O-Navi, and is of type Falcon/MX, see [CD-ROM, 2007, `literature/hardware/datasheet_IMU.pdf`] and [CD-ROM, 2007, `literature/hardware/falcon_imu_guide.pdf`]. An image of the board is shown in Figure 4.9. It measures linear acceleration in all three directions in the range from $-2g$ to $+2g$, where g is the gravitional acceleration (9,82 $^{m}/_{s^2}$), and it measures angular velocity about all three axes in the range from $-150\,°/s$ to $+150\,°/s$ ($\approx -2{,}6\,^{rad}/_{s}$ to $2{,}6\,^{rad}/_{s}$). Notice that the IMU also measures the gravitational acceleration, i.e. if the z-axis of the accelerometer is aligned with ${}^{E}z$, and the IMU is not accelerating relative to the earth frame, the output of the IMU will be $-1g$. This can be regarded as an offset, such that vertically, the range is $-1g$ to $+3g$, with downwards as positive direction.

Figure 4.9: *The IMU board.*

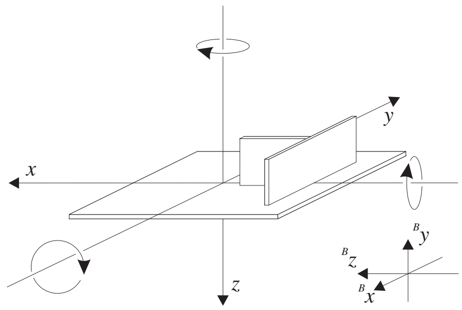


Figure 4.10: *The directions of linear accelerations and angular velocities of the IMU. The body frame axes is shown as well.*

Figure 4.10 shows the directions of the linear accelerations and angular velocities of the IMU, relative to the board. A C program is implemented to convert the serial data from the IMU to six scalar values of linear accelerations and angular velocities, relative to the body frame of the helicopter. Due to the orientation of the IMU when it is attached to the helicopter, it is necessary to rotate the output by multiplying by the following rotation matrix, in order to align the accelerations with the body frame axes:

$$ {}^{B}_{IMU}\mathcal{R} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \quad . \tag{4.1}$$

As the IMU co-ordinate origin is not coincident with the CM of the helicopter, it is necessary to compensate for the offset distances. These are given in the body frame:

$$ {}^{B}\boldsymbol{\kappa}_{imu} = \begin{bmatrix} 0 & 2 & 0 \end{bmatrix}^T \qquad [\,\mathrm{cm}] \qquad . \tag{4.2} $$

When using the IMU, problems with the acceleration data has occured. When the main rotor is rotating, vibrations make the output saturate. This is illustrated in Figure 4.11, where the three acceleration output signals of the IMU is plotted together with the main rotor angular velocity. During the whole test, the helicopter does not leave the ground (this will first happen at a main rotor velocity of around 140 rad/s). It can be seen that both the $\ddot{y}$ and the $\ddot{z}$ measurements becomes erroneous, when the rotor is started up. It is assumed that this is because of too large vibrations, since the angular velocity data still is valid, though very noisy. This problem can be solved by switching it with another IMU, which has a wider measuring range. A $\pm 10g$ IMU has been ordered for this purpose, but has unfortunately not arrived yet.

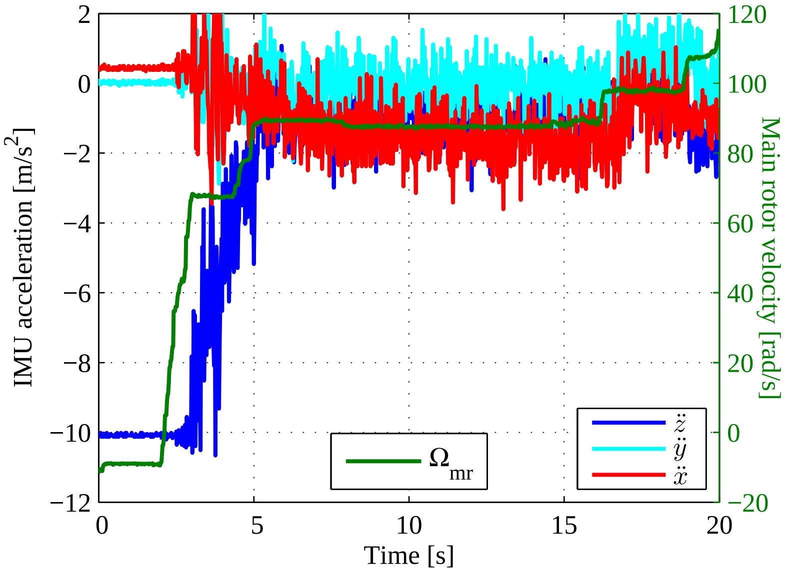


Figure 4.11: The IMU saturates because of vibrations at a main rotor velocity of around 60 rad/s, which is less than the half the velocity needed to take off.

4.3.3 Hardware Implementation on the Helicopter

The servoboard, the IMU and the helicam is mounted on a single aluminium plate, which again is mounted on the helicopter. This makes it easy to disassemble the system in order to revert to the original manual controlled helicopter. Figure 4.12 shows a block diagram after the implementation of the hardware.

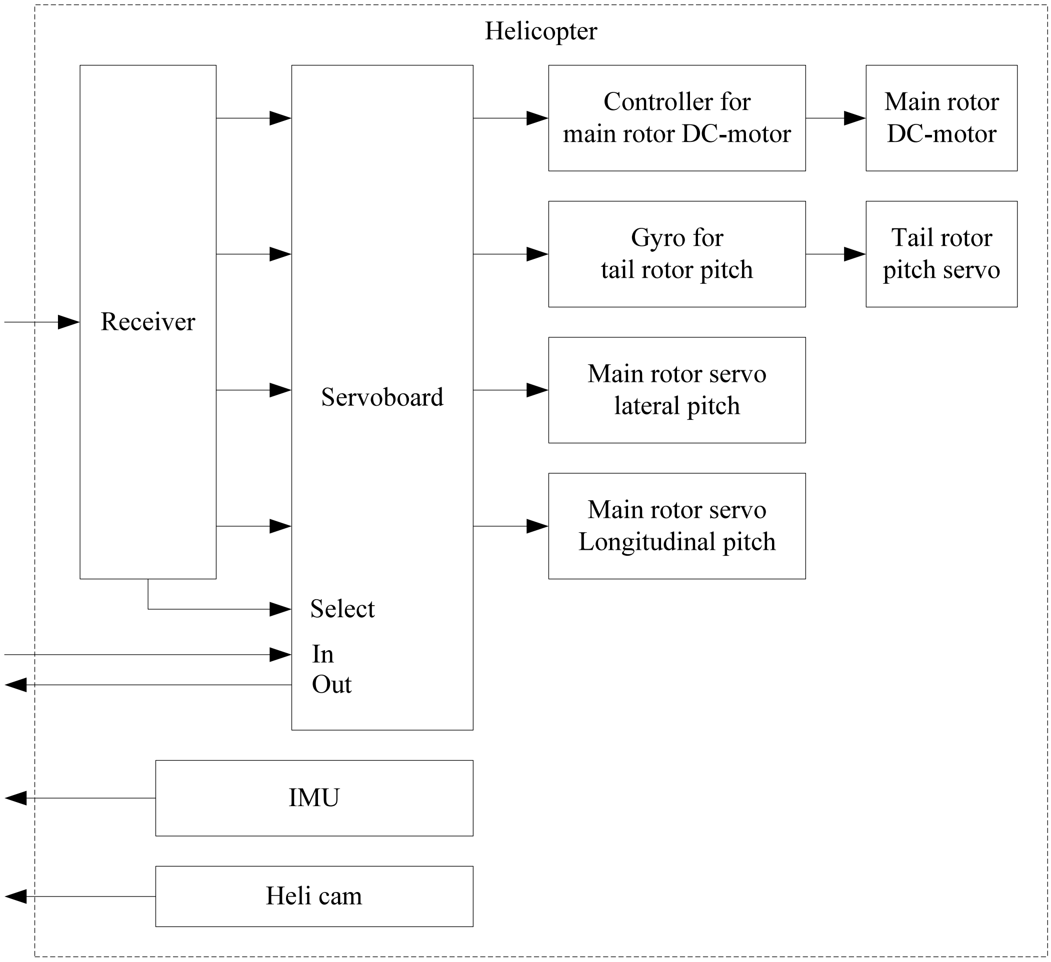


Figure 4.12: *The physical system including the servoboard. The power supply is not included in this diagram.*

As it shows, it is necessary to connect the helicopter and a computer by a number of cables:

- The servoboard has a serial interface; receive, transmit and a ground.

- The IMU also has a serial interface, but only transmit and ground.

- The helicam uses a standard USB connection, with four wires: data in, data out, +5 V and ground. Furthermore, these four wires are shielded, which is the reason why the camera are powered from the USB connection and not from the main power cable.

Further two main power cables, +12 V and ground, are connected.
The following images (Figure 4.13 to Figure 4.21) show how the hardware is implemented on the helicopter.

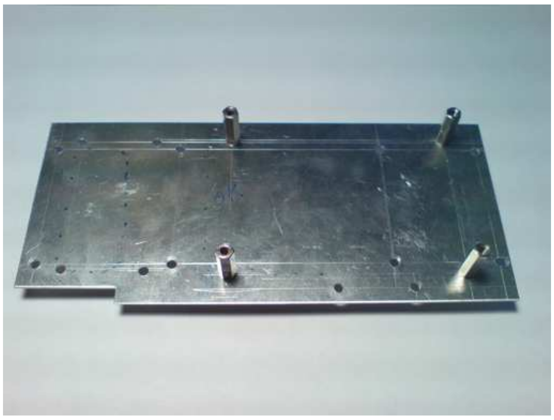


Figure 4.13: *The aluminium plate. It has a thickness of* 1 mm.

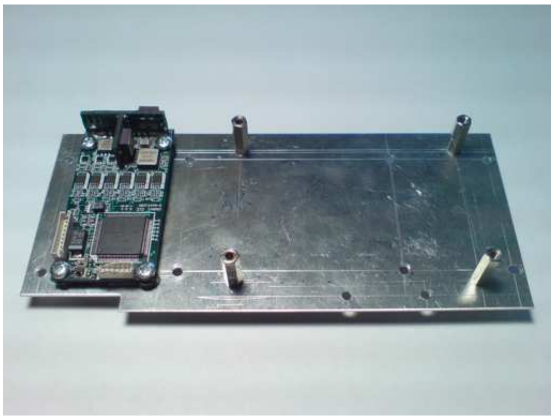


Figure 4.14: *The IMU mounted on the alu plate.*

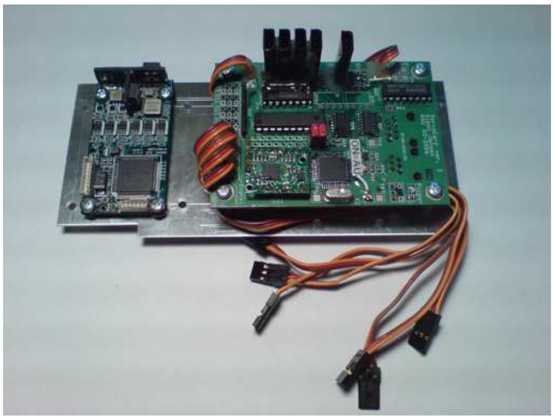


Figure 4.15: *The servoboard mounted on the alu-plate together with the IMU.*

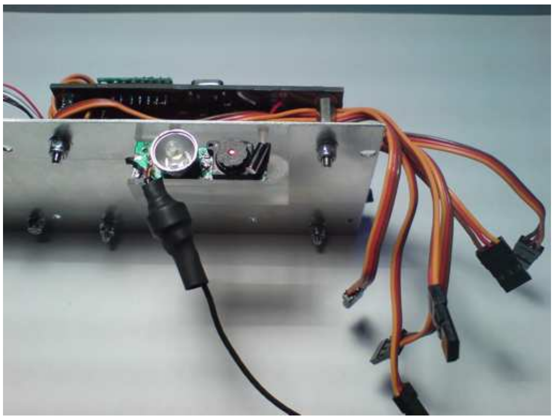


Figure 4.16: *The helicam mounted underneath the alu plate.*

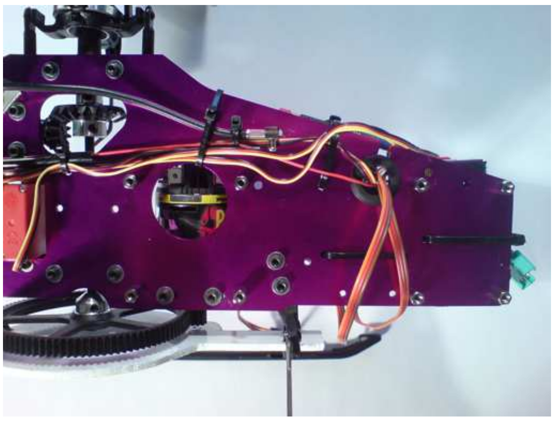


Figure 4.17: *The helicopter frame before mounting the alu plate.*

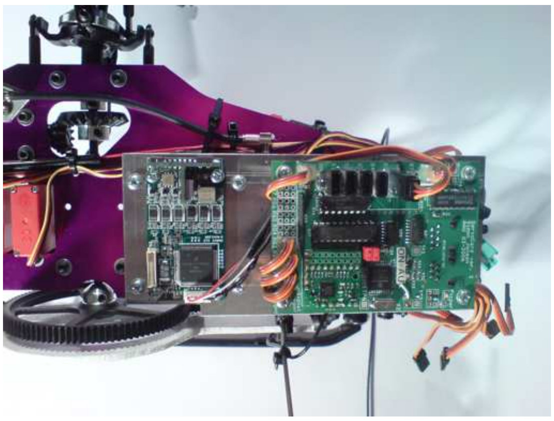


Figure 4.18: *The alu plate mounted on the right-hand side of the helicopter frame.*

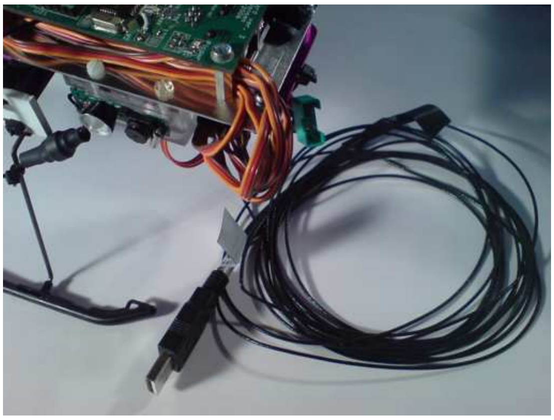


Figure 4.19: *The helicam mounted underneath the helicopter frame.*

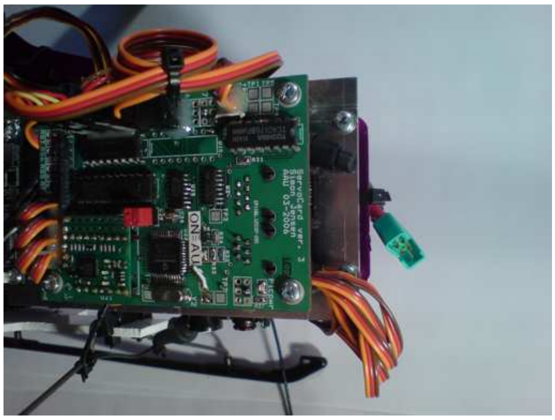


Figure 4.20: *The connectors for the helicopter placed in the front of the frame. The green plug is the main power supply. The black round plug is for the IMU, and the RJ-45 plug is for the servoboard.*

Figure 4.21: *The helicopter with all hardware implemented.*

4.4 Power Supply

The power supply for the helicopter is an external 12 V battery. From this battery, all the components on the helicopter are supplied, except the helicam, which is powered directly through the USB wire. However, due to the fact that the servo motors are consuming a lot of current (periodic 50 Hz peaks), it has been necessary to split the power supply in two separate circuits on the helicopter. This is done by inserting a +5 V voltage regulator (LM7805), which ensures that the servoboard and gyro voltage does not drop critically, when the servos are drawing a lot of current.

The PWM signal from the receiver is only +3 V even though the receiver has a 5 V voltage supply. Noise causes this signal to vary with more than ±0,5 V, so to avoid problems with the CMOS levels a comparator is implemented between the receiver and the servoboard. This ensures that the signals received by the servoboard does not drop below the triggering level.

Figure 4.22 shows a diagram of the power supply for the helicopter.

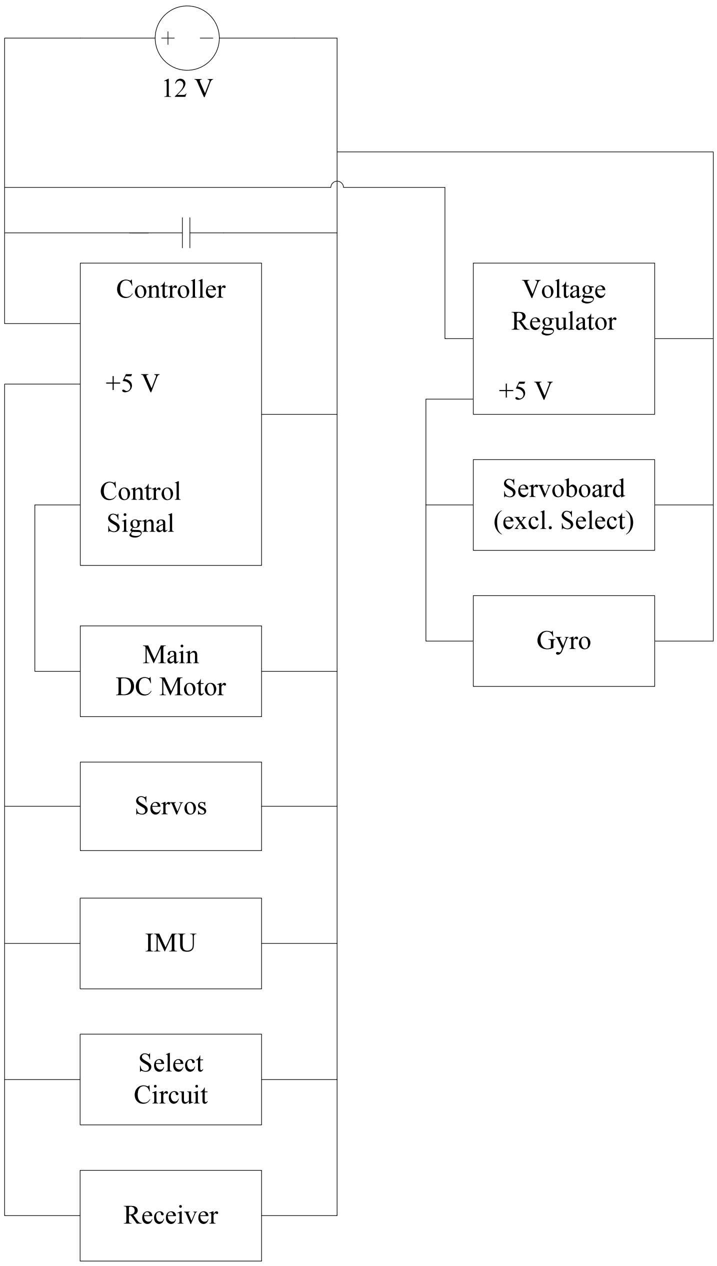


Figure 4.22: *The power supply for the helicopter.*

4.5 Computer Setup

Two computers are used for controlling the helicopter: a Control Computer (CC) and an image processing computer (IPC). The CC is a Linux computer running MATLAB SIMULINK, and the IPC is running Windows XP and MATLAB SIMULINK. The image processing is running seperately in order to save processing power for the CC. The reason for choosing Windows as operating system for the IPC, is that the provided drivers for the cameras are designed for Windows, and the cameras are not supported by Linux.

4.5.1 Image Processing Computer (IPC)

The three USB cameras are all connected to the IPC. In Simulink, the imagery from the cameras can be grabbed, by the use of a video input block. This block returns the imagery in RGB-form, i.e. for each frame a 640×480 matrix for each colour, red, green and blue. The image processing can then be implemented in C using S-functions in Simulink. The output should be an absolute position of the helicopter in three dimensions as well as it's attitude. The computer transmits this data to the CC over a serial connection.

4.5.2 Control Computer (CC)

Besides receiving telemetry data from the IPC, the CC receives data from the helicopter over two serial lines: one from the IMU and one from the servoboard. It transmits the control signals to the helicopter through a third serial line.
The IMU data and the image processing data is fusioned, in order to obtain higher accuracy on the telemetry of the helicopter. As the input from the IMU is linear acceleration and the input from the IPC is absolute position, it will be possible to estimate position, velocity and acceleration seperately through this sensor fusion block.
The software on the CC is mainly implemented in S-functions for MATLAB and SIMULINK. By doing this, most of the code will be written in C or C++, and will therefore be easily portable when later implementing the whole system on an on-board computer.

4.6 Partial Conclusion

The hardware platform described in this chapter has been a quite large part of the project, and considerably amounts of time has been spent, doing error detection and correction. The problems encountered was mostly electrical, but also a single mechanical flaw with the tail rotor gears was solved. Some of the electrical problems included:

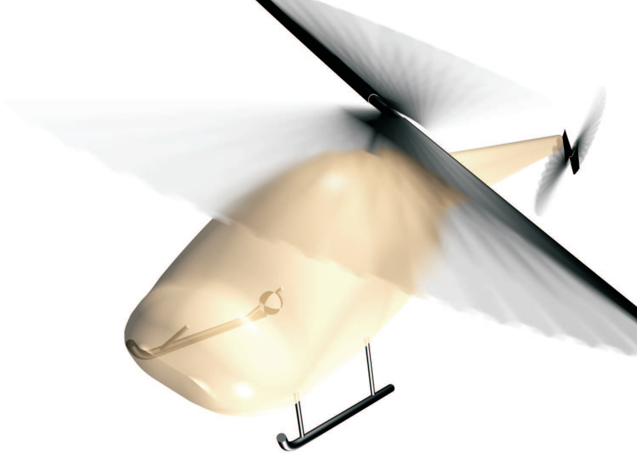
- the helicam, which was initially dismantled in order to reduce weight and to attach the webcam PCB directly to the frame of the helicopter. The cable was very weak in the soldering on the PCB, and it did not withstand the vibrations when flying. Eventually, we shifted the camera with a new one, which was not dismantled.

- the gyro, which was powered through the servo cable. The servos draw a lot of current, and this makes this supply drop down to a level, where the gyro shut off. The solution was to move the gyro power supply to the voltage regulater output seperated from the servo supply.

- the servoboard had the same problems as the gyro with the power.

- the IMU, which saturates due to the vibrations of the helicopter.

Except for the IMU, all the problems were solved, and the hardware is working. Optimization is still possible in several ways, but it is concluded that the system is ready for software implementation, as it is.



5

MODELLING

Chapter Contents

This chapter describes the modelling of the helicopter. The model derived in Hald et al. [2006] is used, and therefore an overview of this model is given first. Even though the model is a generic model, some modifications are needed to adapt it to the Corona 120 used in this project. These modifications are explained in the following sections. After the model has been verified, the parameters necessary for the Corona helicopter are determined.

5.1 Model Overview

The purpose of this project is not to do comprehensive modelling of the helicopter. Therefore an existing helicopter model is used. The existing model is an almost generic helicopter model, for which only parameters for the specific helicopter needs to be inserted. This section serves as an overview of this model, not a derivation, hence formulas are presented and explained rather than derived. Some minor modifications, described in section 5.2 on page 59, are needed to adapt the existing model to the Corona 120. Therefore, the reader should be aware that the formulas and explanations in this section does not necessarily apply for the Corona 120 helicopter, but for the larger Bergen Industrial Twin Helicopter. For in depth information and derivation of the present model, please see Hald et al. [2006] and Bisgaard [2005], and for general helicopter model theory used to derive the model, see Prouty [1990]. The entire state vector used is identical to the one used in Hald et al. [2006]:

$$
\boldsymbol{x} = \begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \psi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ a_0 \\ a_1 \\ b_1 \\ \dot{a}_0 \\ \dot{a}_1 \\ \dot{b}_1 \\ a_{1,sb} \\ b_{1,sb} \\ \dot{a}_{1,sb} \\ \dot{b}_{1,sb} \\ \theta_0 \\ A_1 \\ B_1 \\ \theta_{tr} \\ \dot{\theta}_0 \\ \dot{A}_1 \\ \dot{B}_1 \\ \dot{\theta}_{tr} \end{bmatrix}
\begin{array}{ll}
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{E}\boldsymbol{\Xi} & \text{Position of the helicopter of the helicopter given in \{E\}} \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{E}\boldsymbol{\Theta} & \text{Attitude of the helicopter of the helicopter given in \{E\}} \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{B}\dot{\boldsymbol{\Xi}} & \text{Translatory velocity of the helicopter given in \{B\}} \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{B}\dot{\boldsymbol{\Theta}} & \text{Angular velocity of the helicopter given in \{B\}} \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = \boldsymbol{a}_{mr} & \text{Flapping angles of the main rotor} \\
\left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = \dot{\boldsymbol{a}}_{mr} & \text{Flapping velocities of the main rotor} \\
\left.\begin{matrix} \\ \\ \end{matrix}\right\} = \boldsymbol{a}_{sb} & \text{Flapping angles of the stabilizer bar} \\
\left.\begin{matrix} \\ \\ \end{matrix}\right\} = \dot{\boldsymbol{a}}_{sb} & \text{Flapping velocities of the stabilizer bar} \\
\left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} = \boldsymbol{\Upsilon} & \text{Actuator positions} \\
\left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} = \dot{\boldsymbol{\Upsilon}} & \text{Actuator velocities}
\end{array}
\tag{5.1}
$$

5.1.1 General Model Structure

Basically the model consists of three main blocks as shown in Figure 5.1:

1. Actuator Dynamics:
   - The input to the rotor blade dynamic model is generated by the actuators. To simplify the model, the actuator dynamics are considered unaffected by the dynamics of the rest of the helicopter and can therefore be derived separately.
2. Rotor blade dynamics:
   - Description of the motion of the rotor blades and stabilizer bar. This is the most complex part of the modelling, since the rotor blade dynamics are largely affected by aerodynamic forces. The rotor blade dynamics is split into two parts.
   - The first part describes how a rotor blade flaps due to the aerodynamic forces on the blade.
   - These flapping angles are used in the second part to calculate the force and torques provided by the rotor blades.
3. Force and torque on the rigid body:
   - The forces and torques generated by the rotor blades are acting in the hub of the helicopter. The forces and torques can be seen as acting independently and hence, be split up. Further, the whole helicopter body can be treated as a rigid body because there are no moving parts. Again this part is divided in two.
   - First all the forces and torques calculated in the rotor blade dynamics are summed.
   - Then it is calculated how this affects the motion of the rigid helicopter body.

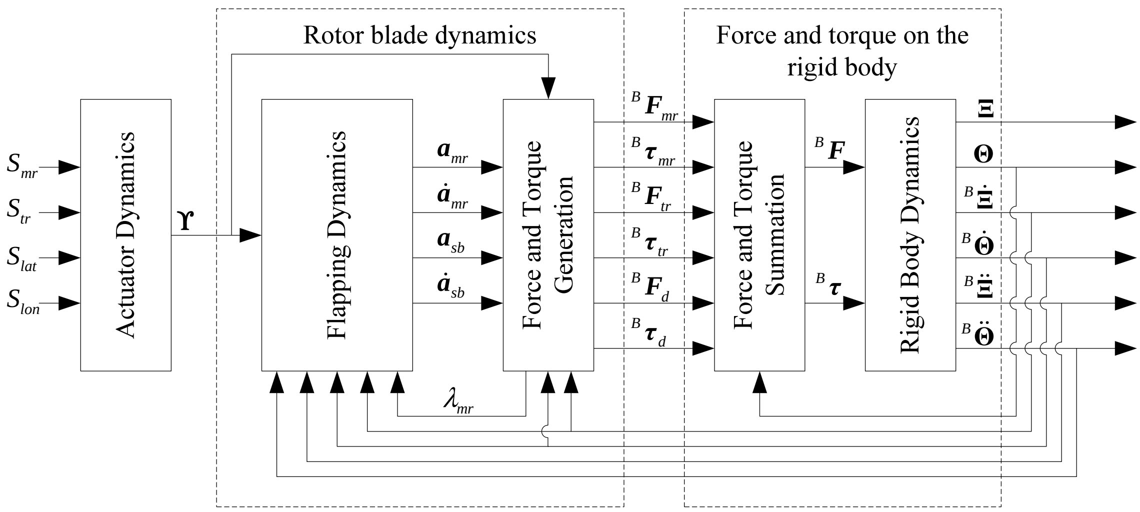


Figure 5.1: *The overall model structure consists of three major blocks, which can be divided into smaller blocks.*

The model is very extensive, and it can be difficult to get an overview of the connection and dependencies of the different parts. To get another perspective on how this model is derived, a reverse argumentation is used (e.g. why it is needed to know the flapping when the force is calculated). The following explanations and illustrations are simplified in the way that the outer feedback loops (position, velocity and acceleration - translatory and angular) shown in Figure 5.1, are removed. This means that the following describes a one step iteration of the model.
The wanted output from the model is the motion of the helicopter, translatory and rotational. To calculate this, it is as shown in Figure 5.2 necessary to know the forces and torques acting on the helicopter. These forces and torques comes from four different parts, namely the main rotor, the tail rotor, gravity and the wind drag on the fuselage. The effect of the stabilizer bar is in this relation negligible, and is only used together with the actuator positions to model the blade pitch. The outer boxes on the figure illustrates the connection with Figure 5.1. The effect from each of the four is described in the following.

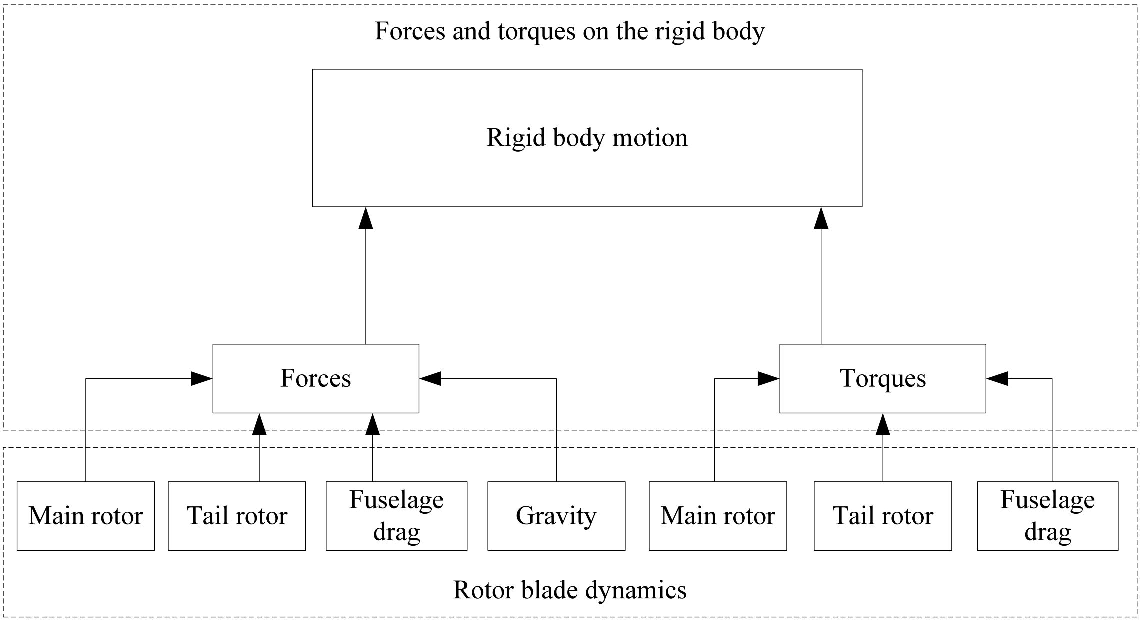


Figure 5.2: *This figure illustrates that to determine the motion of the rigid helicopter body, it is necessary to know the forces and torques acting on it. These forces and torques comes from the main rotor, the tail rotor, gravity, and drag on the fuselage.*

Main Rotor The derivation structure of the forces and torques originating from the main rotor is shown in Figure 5.3 on the next page. Note that this figure should not be seen as the computational structure of the model. It is rather an overview over the path to take to obtain the final equations for forces and torques. For example the blade element lift is never calculated in the model implementation, but used integrated in e.g. the formulas for forces and torques. It can be seen on the Figure, that to calculate the force, it is necessary to know the effect from each blade element, i.e. how much lift is generated and how much drag is generated. The same is the case for the torque generation, but further the flapping angles are needed because a torque arises from the spring effect in the hinge of the blade. The blade drag can be calculated from the rotor blade velocity and the blade characteristics.

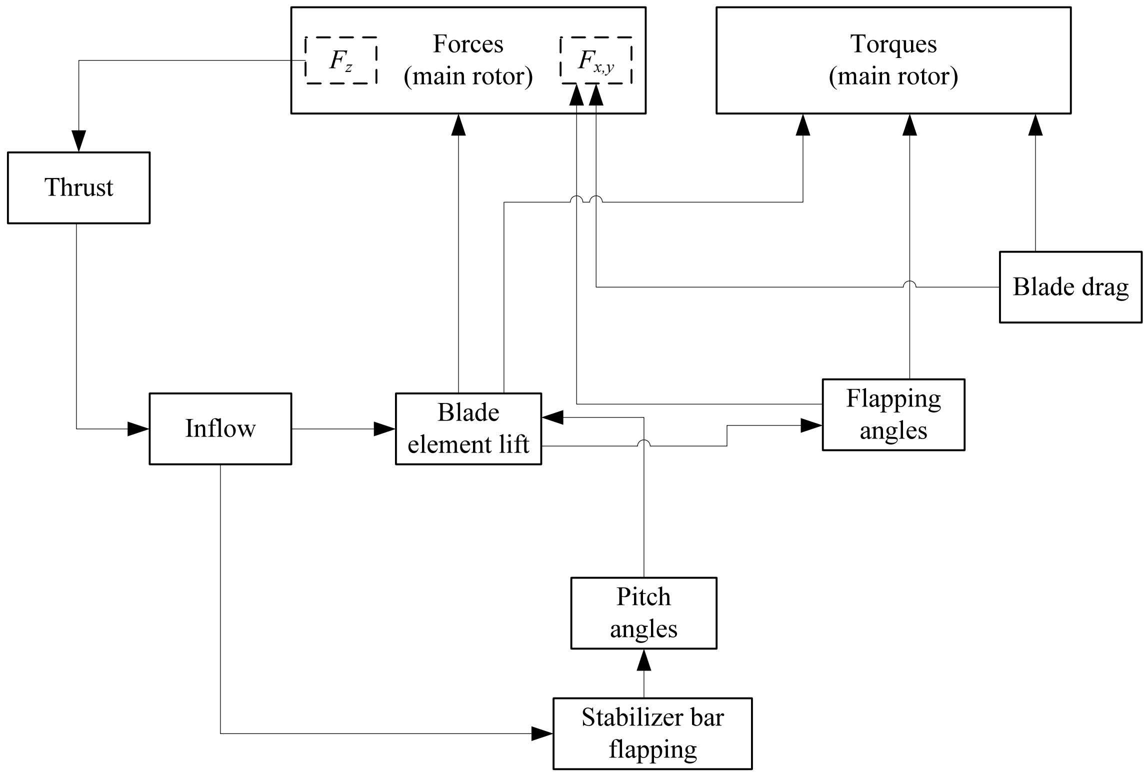


Figure 5.3: Structure of the model to derive the forces and torques originating from the main rotor. It is seen that a central part of the derivation of the model is to calculate the lift provided by a blade element, and that this in fact depend on the force provided by the blade itself. Note that this Figure should not be seen as the computational structure of the model, but rather as an overview of how to obtain equations to calculate the forces and torques.

The flapping is caused by the lift force on each blade can be calculated. The blade lift force can be used to calculate all the torques acting on the blade, and by summation of all the torques, the resulting flapping torque on each blade The most difficult part of the helicopter modelling is to derive the lift of each blade, since this is caused by the aerodynamic forces on the blade. The blade lift can be calculated using blade element theory, which means that a small element of the blade is analyzed and the effect on this element is integrated over the whole blade. The input to this block (as seen in Figure 5.3) is the inflow, which roughly describes the vertical velocity of the air through the hub plane, and of course the pitch of the blade.

The inflow is derived from the thrust, which is the vertical component of the force generated by the main rotor. The force is dependent on the blade element lift, which is again dependent on the inflow. This forms a recursive loop in the model, and when the model is iterated it takes some iterations for the inflow to settle in a steady state.

To calculate the pitch angles of the blade, the actuator input as well as the flapping of the stabilizer bar is used. The flapping of the stabilizer bar is derived the same way as the flapping of a blade.

Tail Rotor The forces and torques from the tail rotor is derived the same way as for the main rotor, but is simpler in the way that some dynamics existing in the main rotor are neglected.

Fuselage Drag The drag on the fuselage is illustrated in Figure 5.4. It can be seen that this drag is also affected by the inflow provided by the main rotor and tail rotor.

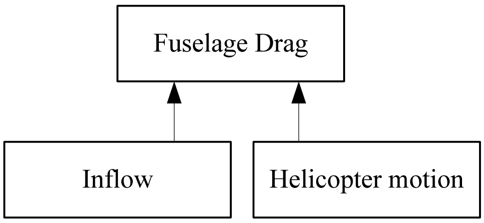


Figure 5.4: *This Figure illustrates that the motion of the helicopter as well as the inflow are needed to calculate the drag on the helicopter fuselage. This means that not only the blade element lift is dependent on the inflow (as shown in Figure 5.3 on the facing page).*

Gravity The gravity effect is the easiest to calculate, since this only depends on the attitude of the helicopter.

In the following two sections two important parts of the derivation of the model equations are described, namely inflow generation and blade element analysis. Hereafter the important major parts of the model, according to Figure 5.1 on page 40, is described in subsequent sections.

5.1.2 Inflow Generation

The above description implies that it is very essential to know how to calculate the lift on each blade (see e.g. Figure 5.3 on the facing page). To do this, it is necessary to know the inflow ratio, so this is the first part of the modelling. It is chosen to model the thrust as uniform over the entire rotor-disc and as a steady state solution, since according to Bisgaard [2005] and [Hald et al., 2006, p.56] this method provides a sufficiently accurate model. The complete derivation of the thrust equations can be found in [Hald et al., 2006, App. B]. The basic idea is that sufficiently high above the rotor, the velocity of the air is zero, and somewhere below the velocity has reached its maximum. The following equations are for the main rotor, but can easily be adjusted to fit the tail rotor. The thrust equation for the main rotor basically describes the mass flow of air through the hub plane (see Figure 5.5 on the next page), which is calculated as the air density multiplied by the area of the hub plane, the induced velocity and the relative air velocity:

$$T_{mr} = 2\rho A v_i U = 2\rho A v_i \sqrt{{}^H\dot{x}^2 + {}^H\dot{y}^2 + ({}^H\dot{z} - v_i)^2} \tag{5.2}$$

- T_{mr} is the magnitude of the thrust generated by the main rotor.
- ρ is the density of the air.
- A is the area of the rotor disk.
- ${}^H\dot{x}$, ${}^H\dot{y}$, ${}^H\dot{z}$ is the velocity of the hub, given in the hub frame {H}.
- v_i is the induced velocity of the air flowing through the rotor disk.
- U is the total air velocity also comprised of the airflow caused by the motion of the helicopter.

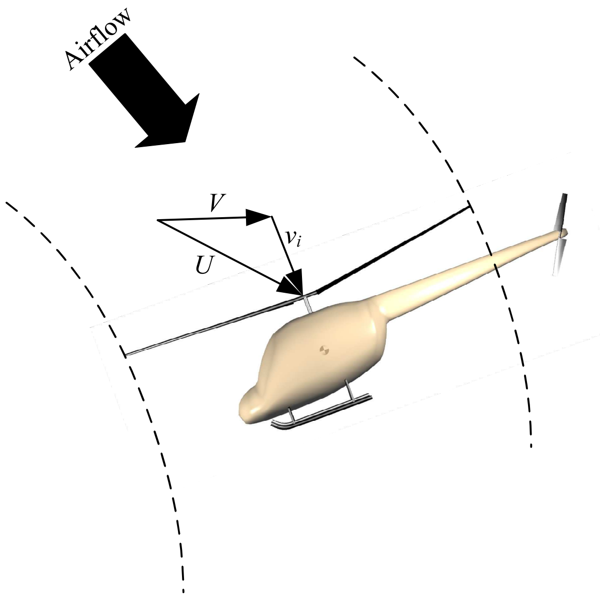


Figure 5.5: The airflow through the main rotor during forward flight. v_i is the induced velocity of air, V is additional airflow caused by the motion of the helicopter and U is the resulting airflow.

By introducing some dimensionless variables, this formula can be rearranged so equations in the following sections are made less complicated.

Inflow ratio:

$$\lambda_{mr} = \frac{{}^{H}\dot{z} - v_i}{\Omega_{mr} R} \quad . \tag{5.3}$$

- λ_{mr} is the dimensionless inflow ratio.
- Ω_{mr} is the rotation velocity of the main rotor blade.
- R is the radius of the rotor blade.

Advance ratios:

$$\mu_x = \frac{{}^{H}\dot{x}}{\Omega_{mr} R} \quad , \quad \mu_y = \frac{{}^{H}\dot{y}}{\Omega_{mr} R} \quad , \quad \mu_z = \frac{{}^{H}\dot{z}}{\Omega_{mr} R} \quad . \tag{5.4}$$

Thrust Coefficient:

$$C_T = \frac{T_{mr}}{\rho A (\Omega_{mr} R)^2} \quad \Leftrightarrow \quad T_{mr} = C_T \rho A (\Omega_{mr} R)^2 \quad . \tag{5.5}$$

Inserting these variables into (5.2) and rearranging, a fourth order equation in λ_{mr} is obtained:

$$\lambda_{mr} = \mu_z - \frac{C_T}{2\sqrt{\mu_x^2 + \mu_y^2 + \lambda_{mr}^2}} \quad . \tag{5.6}$$

The solution to this equation can be found by using the Newton-Raphson method, which is used in Hald et al. [2006]. However, this method has a potential of yielding the wrong solution or not converging. Since Equation (5.6) can be rearranged to a fourth order equation (by squaring), it is possible to solve this equation analytically. This is done instead of using the Newton-Raphson method. The solution produces four roots, but only one of them is the correct solution to the inflow (the other roots arises when squaring). The correct solution is found by inserting each root into the original equation (5.6). Then the one root which produces the smallest error, which might not be zero because of numerical issues, is chosen as the inflow.

5.1.3 Blade Element Analysis

When the inflow is known, it is possible to calculate the lift of each blade using blade element analysis. This is the mathematical most difficult part of the modelling, since this is the part where all aerodynamic effects on the rotor blades are modelled. Basically the idea is to analyze the forces acting on a small element of the blade Δr. If this is done as a function of the distance from the hub, the results can be integrated over the blade length to obtain the whole force and torque provided by the blade. Further to obtain a steady state solution, an integration about a whole blade revolution can be done. Notice that all the equations are calculated for one blade. After setting up the equations, it is possible to multiply by the number of blades, but for example in the calculation of the flapping, the forces acting on the one blade alone is most useful.
Note that in this section, the variables θ_b, θ_0, and ϕ_r are used, and these are not the same as the variables ϕ and θ for the attitude of the helicopter:

- θ_b is the instantaneous pitch of a blade.
- θ_0 is the collective pitch of a blade.
- ϕ_r is the inflow angle relative to a zero pitch angle ($\theta_b = 0$) of a blade.

The force acting on the blade can be split into two perpendicular forces, namely a lift ΔL and a drag ΔD. These can be seen in Figure 5.6. Note that the lift and drag are treated as scalars and not vectors as seen on the Figure.

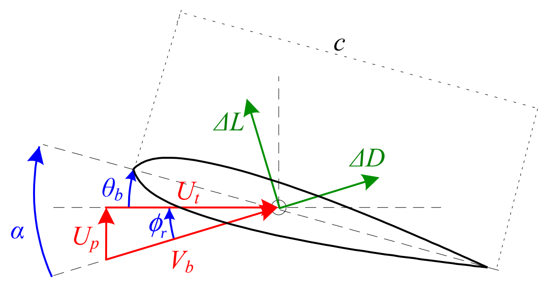


***Figure 5.6:** Illustration of a cross section a blade (a blade element).*

The aerodynamic modelling of the lift on a small element of the blade Δr is rather comprehensive; see for instance: Prouty [1990], Bramwell [1976], Johnson [1994] and Padfield [1996]. The equation relating the blade element to the lift can be expressed as:

$$\Delta L = \frac{\rho}{2} V_b^2 C_L c \cdot \Delta r \tag{5.7}$$

- ΔL is the lift of the small element of the blade.
- Δr is a small element of the blade.
- ρ is the density of the air.

- V_b is the velocity of the air relative to the blade.
- C_L is the local lift coefficient of the blade element.
- c is the length of the blade chord.

The velocity of the air V_b is a function of the translatory and rotational velocity of the helicopter together with the inflow and the rotation velocity of the blade itself. The local lift coefficient C_L can be expressed as a function of the angle of attack α as shown in Figure 5.7. As seen on the figure the lift coefficient can be approximated by the product of the angle of attack α and the lift curve slope C_{ls}:

$$C_L = C_{ls} \cdot \alpha \tag{5.8}$$

$$V_b = f\left({}^{B}[\dot{x},\dot{y},\dot{z},\dot{\phi},\dot{\theta},\dot{\psi}],\lambda_{mr},\Omega,r\right) \tag{5.9}$$

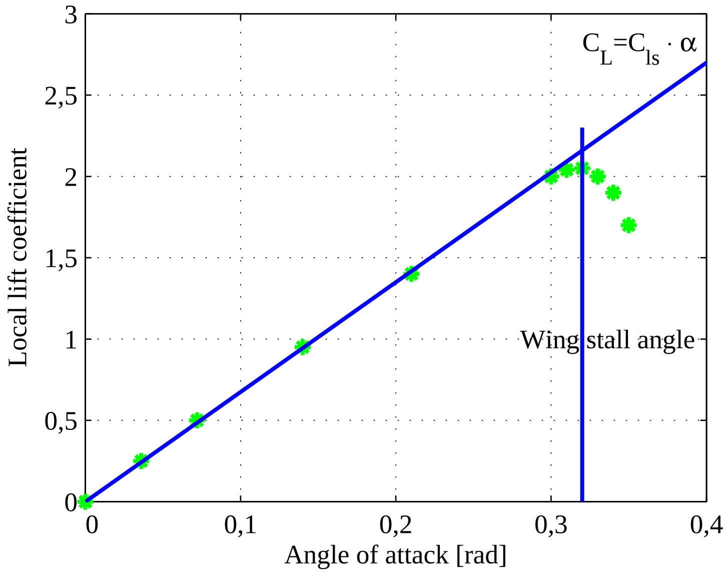


Figure 5.7: A "free hand" sketch of typical measurements of the local lift coefficient for a blade (freely adapted from [Prouty, 1990, p.23]). It is seen that the lift coefficient C_L is nearly proportional to the angle of attack α until the wing stalls.

By using this and the assumption that the tangential velocity of the blade U_t is much greater than the perpendicular U_p, a formula for the lift can be obtained - this is done in [Hald et al., 2006, p. 36-40]. By dividing (5.7) by Δr and letting $\Delta r \rightarrow 0$, the differential quotient $\frac{dL}{dr}$ is obtained. The formula for the magnitude of lift can be achieved by integration from the hinge to the tip of the blade:

$$L = \int_e^R \frac{\rho}{2} C_{ls} V_b^2 \, \alpha \, c \, dr \quad , \tag{5.10}$$

$$\Leftrightarrow L = \frac{\rho}{2} C_{ls} \int_e^R U_t^2 \left(\theta_b + \frac{U_p}{U_t}\right) c \, dr \quad , \tag{5.11}$$

where the blade pitch θ_b is given by

$$
\begin{aligned}
\theta_b &= \theta_0 - A_{1,mr}\cos(\Psi) - B_{1,mr}\sin(\Psi) + \theta_{tw}\frac{e+r}{R} - K_1\beta && (5.12)\\
A_{1,mr} &= A_1K_b - b_{1,sb}K_h && (5.13)\\
B_{1,mr} &= B_1K_b - a_{1,sb}K_h \qquad . && (5.14)
\end{aligned}
$$

- θ_0 is the collective pitch.
- $A_{1,mr}$ is the lateral blade pitch angle.
- $B_{1,mr}$ is the longitudinal blade pitch angle.
- A_1 is the lateral swash plate angle.
- B_1 is the longitudinal swash plate angle.
- θ_{tw} is the twist of the tip of the blade compared to the twist at the hinge.
- e is the hinge offset.
- r is the distance from the hinge.
- R is the main rotor radius
- K_1 is the cross-coupling between the flapping angle and the pitch angle.
- β is the flapping angle of the blade.
- K_b is the bell gain-factor.
- K_h is the hiller gain-factor.
- $a_{1,sb}$ is the longitudinal flapping angle of the stabilizer bar.
- $b_{1,sb}$ is the lateral flapping angle of the stabilizer bar.
- Ψ is the rotational angle of the blade.

The direction of the lift ${}^H\hat{e}_{Lift}$ depends on the inflow angle ϕ_r, the blade flapping angle β, and the revolution angle Ψ:

$$
{}^H\hat{e}_{Lift} = \begin{bmatrix} \sin(\phi_r)\sin(\Psi) + \cos(\phi_r)\sin(\beta)\cos(\Psi) \\ -\sin(\phi_r)\cos(\Psi) + \cos(\phi_r)\sin(\beta)\sin(\Psi) \\ -\cos(\phi_r)\cos(\beta) \end{bmatrix} \qquad . \tag{5.15}
$$

The derivation of the drag is similar to that of the lift, and the result is:

$$
\begin{aligned}
\Delta D &= \frac{\rho}{2}V_b^2 C_d c \cdot \Delta r \qquad , && (5.16)\\
D &= \frac{\rho}{2}\, C_d \int_e^R U_t^2\left(\theta_b + \frac{U_p}{U_t}\right) c\, dr \qquad . && (5.17)
\end{aligned}
$$

- C_d is the drag coefficient for the main rotor blades.

5.1.4 Actuator Dynamics

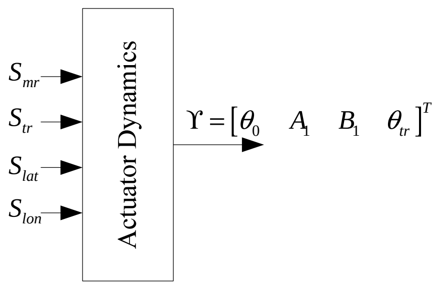


Figure 5.8: *The actuator dynamics block of the model taken from Figure 5.1 on page 40.*

According to Brennan [1997], the servo motors can be well approximated by rate limited second order systems. This means that the actuator block (see Figure 5.8) has the transfer functions including rate limitations

$$H_0 = \frac{\theta_0}{S_{col}} = \frac{K_0 {\omega_{n,0}}^2}{s^2 + 2\zeta_0\omega_{n,0}s + {\omega_{n,0}}^2} \tag{5.18}$$

$$H_A = \frac{\theta_A}{S_{lat}} = \frac{K_A {\omega_{n,A}}^2}{s^2 + 2\zeta_A\omega_{n,A}s + {\omega_{n,A}}^2} \tag{5.19}$$

$$H_B = \frac{\theta_B}{S_{lon}} = \frac{K_B {\omega_{n,B}}^2}{s^2 + 2\zeta_B\omega_{n,B}s + {\omega_{n,B}}^2} \tag{5.20}$$

$$H_{tr} = \frac{\theta_{tr}}{S_{tr}} = \frac{K_{tr} {\omega_{n,tr}}^2}{s^2 + 2\zeta_{tr}\omega_{n,tr}s + {\omega_{n,tr}}^2} \tag{5.21}$$

- S_x is the input to the servo.
- K_x is the DC gain.
- $\omega_{n,x}$ is the undamped natural frequency.
- ζ_x is the damping ratio.

The rate limit is called $\dot{\theta}_{x,max}$ and the input saturation $S_{x,max}$ and $S_{x,min}$.

5.1.5 Flapping Dynamics

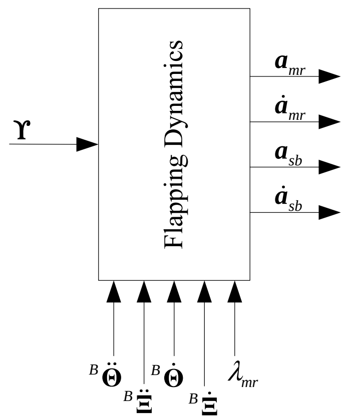


Figure 5.9: *The flapping dynamics block of the model taken from Figure 5.1 on page 40.*

As described in 3.2 on page 14, the rotor blade dynamics or flapping dynamics describe the motion of the rotor blades. The flapping with a flapping angle β is shown in Figure 5.10. It is assumed that only the main rotor and the stabilizer bar exhibits flapping, since the tail rotor is much more stiff, and there are only collective pitch and no cyclic pitch.

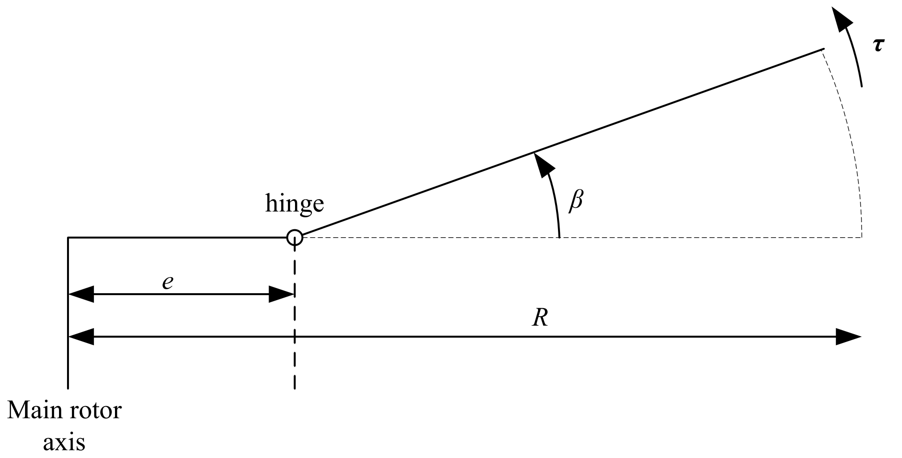


Figure 5.10: *Figure illustrating the flapping β, virtual hinge offset e and the rotor blade radius R*

Since flapping concerns the angle β of the hinge, the motion has only to do with the torque around this point. The torque is defined as a vector, around which the torque acts, and with a length equal to the magnitude of the torque. This means that the torque is always perpendicular to the blade, which again means that the direction of the torque vector given in {B} is $\begin{bmatrix} -\sin(\Psi) & \cos(\Psi) & 0 \end{bmatrix}^T$ if the blade is bending upwards (away from the fuselage) and $\begin{bmatrix} \sin(\Psi) & -\cos(\Psi) & 0 \end{bmatrix}^T$ if the blade is bending downwards (towards the fuselage). However, since the flapping is only dependent on the force exerted on the blade, and not the position, it makes calculations easier if only the magnitude of the torques are considered. In the following the magnitude of the torques are defined with the opposite sign as the force acting on the blade, e.g. if a force presses the blade down (positive force in the z-axis), the torque will be negative.
The torques which are included in the modelling is the following (the sign in the parenthesis denotes the sign of the torque under normal flight operation i.e. hover):

- τ_β, flapping torque, which is the resulting torque (+/−).
- τ_a, aerodynamic torque calculated using blade element calculations (+).
- τ_{cf}, centrifugal torque is the effect that the blade is pulled away from the hub since it rotates (−).
- τ_R, restraint torque coming from the spring like behaviour of the blade (−).
- τ_{ba}, body angular torque is the result of the rotation of the helicopter itself (dependent on $sign(\ddot{\phi},\ddot{\theta})$ and Ψ).
- τ_{bn}, body normal torque is the result of the helicopter body acceleration (same sign as ${}^H\ddot{z}$) .
- τ_{cor}, Coriolis force originating helicopter rotation and the blade element moving at the same time..

$$\tau_\beta = \tau_a + \tau_{cf} + \tau_R + \tau_{ba} + \tau_{bn} + \tau_{cor} \tag{5.22}$$

All the torques can be calculated using blade element theory, and the resulting torque is equal to the product of the angular acceleration, and the inertia of the blade[1]:

[1]Note in Hald et al. [2006] and Bisgaard [2005] the sign of τ_β is opposite, since it in (5.22) is used as the resulting torque.

$$\boldsymbol{\tau}_\beta = \dot{\beta} I_b \qquad . \tag{5.23}$$

The flapping angle β is according to Prouty [1990] well approximated by the first harmonics:

$$\beta = a_0 - a_1 \cos(\Psi) + b_1 \sin(\Psi) \qquad . \tag{5.24}$$

Hence (5.23) can be split in terms of a_0, a_1, b_1, the coning, longitudinal and lateral flapping, and all higher order harmonics discarded. This results in a tree dimensional differential equation of the form:

$$\ddot{\boldsymbol{a}}_{mr} + \mathcal{D}\dot{\boldsymbol{a}}_{mr} + \mathcal{K}\boldsymbol{a}_{mr} = \mathcal{J}\boldsymbol{\Upsilon}_{mr} + \boldsymbol{E}\lambda + \boldsymbol{G} \qquad . \tag{5.25}$$

- $\boldsymbol{a}_{mr} = \begin{bmatrix} a_0 \\ a_1 \\ b_1 \end{bmatrix}$ is the vector describing the flapping of the main rotor.
- λ is the inflow ratio.
- $\boldsymbol{\Upsilon}_{mr} = \begin{bmatrix} \theta_0 \\ A_{1,mr} \\ B_{1,mr} \end{bmatrix}$ is a linear combination of the actuator positions and stabilizer bar according to the bell-hiller gain.
- $\mathcal{D}$,$\mathcal{J}$,$\mathcal{K}$ are 3×3 matrices.
- $\boldsymbol{E}$,$\boldsymbol{G}$ are 3 dimensional vectors.

Stabilizer Bar Flapping

Now, the approximate same procedure can be used to derive differential equations for the flapping of the stabilizer bar. However, the lift of the stabilizer bar is considered so small that it is omitted in the lift equations. Therefore the flapping of the stabilizer bar is only used to slow the dynamics of the helicopter, i.e. slow down the pitch velocity of the main rotor. The pitch is given by Equation (5.12), and after that used in Equation (5.25). Further, since the stabilizer bar is mounted in a free pivot joint on the hub, there is no restraint torque, no hinge offset, and no collective pitch. The torque equation is therefore:

$$\boldsymbol{\tau}_{\beta,sb} = \boldsymbol{\tau}_{a,sb} + \boldsymbol{\tau}_{cf,sb} + \boldsymbol{\tau}_{ba,sb} + \boldsymbol{\tau}_{bn,sb} + \boldsymbol{\tau}_{cor,sb} \qquad . \tag{5.26}$$

The differential equation is the similar to Equation (5.25), but only two dimensional because there is no coning angle. Hence the output is:

$$\boldsymbol{a}_{sb} = \begin{bmatrix} a_{1,sb} \\ b_{1,sb} \end{bmatrix} \qquad . \tag{5.27}$$

5.1.6 Force and Torque on the Rigid Body

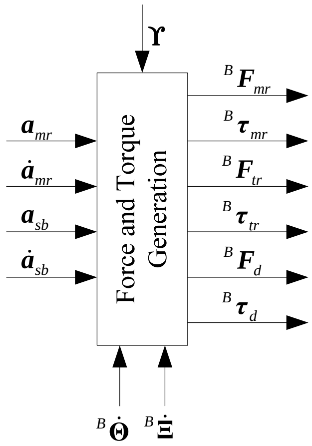


Figure 5.11: *The force and torque generation block of the model taken from Figure 5.1 on page 40.*

In the previous sections the basic equations to derive the motion and the forces acting on the rotor blades were derived. These equations are now used to calculate the total effect on the rotation and translation of the whole helicopter. The helicopter is in this section regarded as a rigid body on which the forces and torques act. The forces and torques originates from the main and tail rotor together with the drag from the air flowing past the fuselage of the helicopter. In the following sections the forces and torques are described for the three sources, respectively.

Main Rotor - Forces

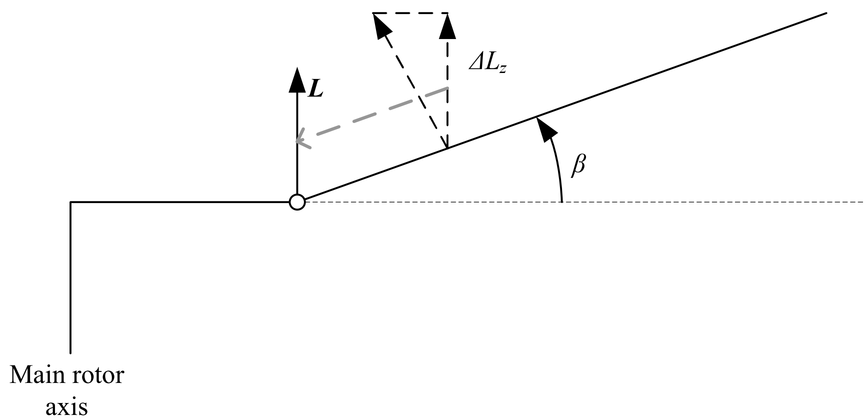


Figure 5.12: *The lift projectet onto the z-axis and translated to operate in the virtual hinge.*

The contribution from the main rotor comes from the lift L and drag D given in Equations (5.11) and (5.17). It can be shown that these two forces can be considered as acting in a virtual hinge as shown in Figure 5.12 where the lift is projected onto the z-axis and moved to the virtual hinge. To calculate the forces acting in the virtual hinge the following steps are taken and exemplified by the calculation of ${}^HF_{z,mr}$:

1. Split L and D into components along the three axes of the hub frame {H}, by using the direction given in Equation (5.15) and illustrated in Figure 5.12 for the lift and a similar formula for the drag.

$$ {}^{H}L_z = -\cos(\phi_r)\cos(\beta)\cdot L \quad , \quad {}^{H}D_z = -\sin(\phi_r)\cos(\beta)\cdot D \qquad . \tag{5.28}$$

2. Add the two forces in each direction, yielding $F_{x,mr}$, $F_{y,mr}$, $F_{z,mr}$ as a function of the inflow angle ϕ_r, the blade flapping angle β, and the revolution angle Ψ.

$$ {}^{H}F_{z,mr,2} = -\cos(\phi_r)\cos(\beta)\cdot L + (-\sin(\phi_r)\cos(\beta))\cdot D \qquad . \tag{5.29}$$

3. Use small angle approximations for ϕ_r and β and the assumption that $|D| \ll |L|$ to reduce complexity of the equations.

$$ {}^{H}F_{z,mr,3} = -L + 0\cdot D \qquad . \tag{5.30}$$

4. Integrate around a whole revolution of the blade, and divide by 2π to get the average magnitude of the force:$\frac{1}{2\pi}\int_0^{2\pi}(\cdot)\,d\Psi$.

$$ {}^{H}F_{z,mr,4} = \frac{1}{2\pi}\int_0^{2\pi} -L d\Psi \qquad . \tag{5.31}$$

5. Multiply by the number of blades b on the helicopter.

$$ {}^{H}F_{z,mr} = \frac{-b}{2\pi}\int_0^{2\pi} L d\Psi \qquad . \tag{5.32}$$

The result is the force vector on the rigid body:

$$ {}^{H}\boldsymbol{F}_{mr} = \begin{bmatrix} {}^{H}F_{x,mr} \\ {}^{H}F_{y,mr} \\ {}^{H}F_{z,mr} \end{bmatrix} \tag{5.33}$$

which is the same as the force given in the body frame ${}^{B}\boldsymbol{F}_{mr}$, since the two frames are aligned.

Main Rotor - Torques

What is left is the torque contribution from the main rotor. This can be split in to three:

1. The torque arising from the fact that the hub is not placed in the center of mass, $\boldsymbol{\tau}_{hub}$.

2. The torque arising from the fact that the forces given in (5.33) are not acting directly in the hub, but in the virtual hinge, denoted $\boldsymbol{\tau}_{aero}$ (see Figure 5.13 on the facing page).

3. The torque directly affecting the hinge $\boldsymbol{\tau}_{res}$ (see Figure 5.13).

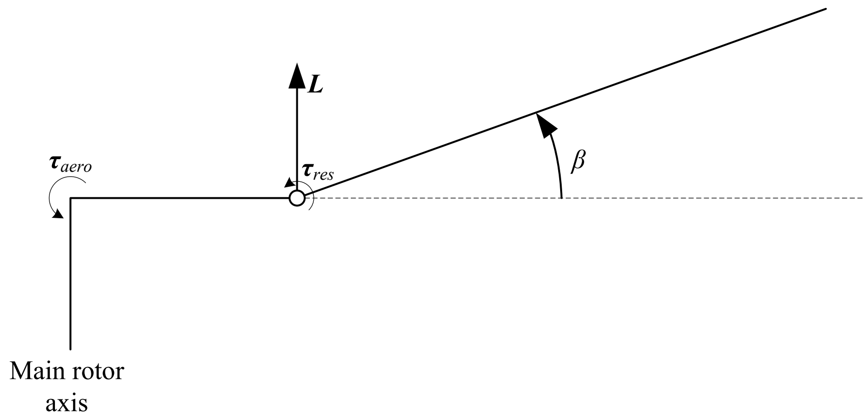


Figure 5.13: *The resulting torques $\boldsymbol{\tau}_{res}$ and $\boldsymbol{\tau}_{aero}$ acting in the hinge and in the hub.*

Torque 1: $\boldsymbol{\tau}_{hub}$ is the cross product of the vector from CM to the hub and the force vector acting in the hub Hald et al. [2006].[2]:

$$ {}^{B}\boldsymbol{\tau}_{hub} = {}^{B}\boldsymbol{\kappa}_{hub} \times {}^{H}\boldsymbol{F}_{mr} = \begin{bmatrix} \kappa_{y,hub}\,{}^{H}F_{z,mr} - \kappa_{z,hub}\,{}^{H}F_{y,mr} \\ \kappa_{z,hub}\,{}^{H}F_{x,mr} - \kappa_{x,hub}\,{}^{H}F_{z,mr} \\ \kappa_{x,hub}\,{}^{H}F_{y,mr} - \kappa_{y,hub}\,{}^{H}F_{x,mr} \end{bmatrix} \tag{5.34} $$

- ${}^{B}\boldsymbol{\kappa}_{hub}$ is the distance from the CM to the hub given in {B}.
- ${}^{H}F$ is the forces provided by the rotor blades.

Torque 2: $\boldsymbol{\tau}_{hub}$ occurs because the forces are acting in the virtual hinges. Here ${}^{H}F_z$ from Equation (5.33) can not be used directly since this is an average around a whole revolution and the torque vector is in a direction perpendicular to the blade. Therefore the expression for the lift in the direction of the z-axis is used together with a small angle approximation for β:

$$ \tau_{aero,x} = b\,e\,L\,\cos(\beta)\sin(\Psi) \approx b\,e\,L\,\sin(\Psi) \quad , \tag{5.35} $$

$$ \tau_{aero,y} = -b\,e\,L\,\cos(\beta)\cos(\Psi) \approx -b\,e\,L\,\cos(\Psi) \quad . \tag{5.36} $$

- b is the number of blades on the helicopter.
- L is the lift generated by a rotor blade.
- e is the distance from the hub to the hinge.

The same can be done for the torque around the ${}^{H}z$-axis. However the lift vector L is instead projected onto the hub plane, which is perpendicular to the z-axis, and a component coming from the drag occurs as well:

$$ \tau_{aero,z} = -b\,(r+e)\,(D\,\cos(\phi_r) - L\,\sin(\phi_r)) \approx -b\,(r+e)\,(D - L\,\phi_r) \quad . \tag{5.37} $$

As with the forces, these torques are integrated around a whole revolution to get the average.

[2]This notation differs a bit from the one used in Hald et al. [2006], where the absolute horizontal and vertical distances (κ_h and κ_v) are used. In this report a vector ${}^{B}\boldsymbol{\kappa}_{hub}$ is used to capture all three axis and a possible negative position offset of an object. This procedure is also used for other ${}^{B}\boldsymbol{\kappa}$ in this thesis.

Torque 3: $\boldsymbol{\tau}_{hub}$ is the restraint torque (from Equation (5.22)) originating from the spring constant in the hinge multiplied by the flapping angle. However in this case it is needed to split the torque into a vector along the x- and y-axis. The coning effect a_0 cancels itself if the average torque is taken about a whole revolution, and so does the longitudinal flapping angle a_1 when calculating the torque around the x-axis. Therefore the restraint torque around the x-axis becomes:

$$\tau_{res,x} = K_s\, b_1\ \sin(\Psi)\ \sin(\Psi) \qquad . \tag{5.38}$$

- $b_1 \cdot \sin(\Psi)$ is the flapping angle.
- $K_s \cdot \sin(\Psi)$ is the spring constant multiplied by the effect caused by the angle of the blade.

Integrating this around a whole revolution to get the average effect and multiplying by the number of blades yields

$$\tau_{res,x} = \frac{b}{2}\, K_s\, b_1 \qquad . \tag{5.39}$$

and similarly for the y-axis

$$\tau_{res,y} = \frac{b}{2}\, K_s\, a_1 \qquad . \tag{5.40}$$

There is no restraint torque around the z-axis, since the hinge is a pivot around this axis. To get the total torque acting on the helicopter, the torques are summed yielding (κ_y is omitted since it is zero and the expression for $\boldsymbol{\tau}_{aero}$ is not inserted because it is rather complex):

$$^{B}\boldsymbol{\tau}_{mr} = \begin{bmatrix} ^{B}\tau_{x,mr} \\ ^{B}\tau_{y,mr} \\ ^{B}\tau_{z,mr} \end{bmatrix} = \begin{bmatrix} \tau_{aero,x} + \frac{b}{2}\, K_s\, b_1 - \kappa_{z,hub}\ {}^{H}F_{y,mr} \\ \tau_{aero,y} + \frac{b}{2}\, K_s\, a_1 + \kappa_{z,hub}\ {}^{H}F_{x,mr} - \kappa_{x,hub}\ {}^{H}F_{z,mr} \\ \tau_{aero,z} + \kappa_{x,hub}\ {}^{H}F_{y,mr} \end{bmatrix} \qquad . \tag{5.41}$$

Tail Rotor

The forces and torques for the tail rotor can be derived in a similar way as for the main rotor using blade element theory. However it is simpler since some of the dynamics does not exist on the tail rotor:

- There is no cyclic pitch.
- The blades are modelled as stiff, hence there is no flapping.
- The thrust is parallel to the ^{T}y-axis.
- The rotor is modelled with no hinge offset.

Hence the whole flapping calculations can be avoided and we end up with equations:

$$^{B}\boldsymbol{F}_{tr} = \begin{bmatrix} ^{B}F_{x,tr} \\ ^{B}F_{y,tr} \\ ^{B}F_{z,tr} \end{bmatrix} = \begin{bmatrix} 0 \\ ^{T}F_{y,tr} \\ 0 \end{bmatrix} \qquad , \tag{5.42}$$

$$^{B}\boldsymbol{\tau}_{tr} = \begin{bmatrix} ^{B}\tau_{x,tr} \\ ^{B}\tau_{y,tr} \\ ^{B}\tau_{z,tr} \end{bmatrix} = \begin{bmatrix} -\kappa_{z,tr}\ {}^{T}F_{y} \\ ^{T}\tau_{y,tr} \\ \kappa_{x,tr}\ {}^{T}F_{y} \end{bmatrix} \qquad . \tag{5.43}$$

- $^{B}\boldsymbol{F}_{tr}$ is the force contribution from the tail rotor.
- $^{B}\boldsymbol{\tau}_{tr}$ is the torque contribution from the tail rotor.
- $^{B}\boldsymbol{\kappa}_{tr}$ is the vector from the CM to the origin of {T} given in {B}.
- $^{T}F_{y,tr}$ is the thrust generated by the tail rotor, and is derived in a similar way as for the main rotor by integrating the lift (see Equation 5.11 on page 46) around a revolution: $\frac{1}{2\pi}\int_0^{2\pi} -dL_{tr}\, d\Psi_{tr}$ (the minus is because the thrust is in the direction $-^{T}\boldsymbol{y}$).

Drag

As an effect of the air flowing past an object, drag arises. This is an effect of both the wake of the rotors and the velocity of the helicopter. In this model drag on the fuselage (D_x, D_y, D_z), the front plane D_{fp}, the tail plane D_{tp} and the tail fin D_{tf} are considered. These are illustrated in Figure 5.14.

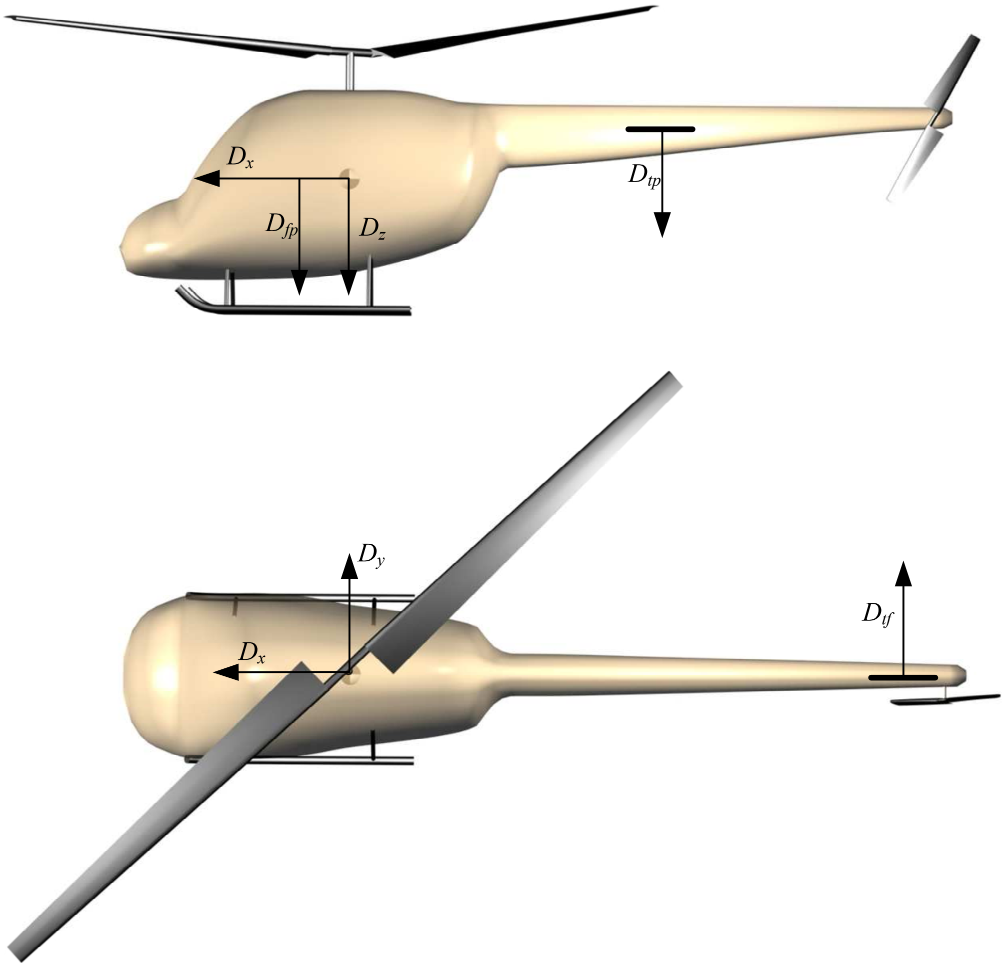


Figure 5.14: The drag affected by the air flowing past the helicopter. The considered drags are fuselage drag (D_x, D_y, D_z), drag on the front plane D_{fp}, drag on the tail plane D_{tp} and drag on the tail fin D_{tf}.

Drag is calculated by using a quadratic function of the velocity of the air (a modified version of [Prouty, 1990, p. 21]):

$$D = \frac{\rho}{2}\, d\, A\, |v_{air}| v_{air} \qquad . \tag{5.44}$$

- ρ is the density of the air.
- d is a drag coefficient.
- A is the area of the object seen in the direction of the air.
- v_{air} the velocity of the air compared to the object - the $|\cdot|$ is to get the right sign on the equation.

Fuselage drag The drag on the fuselage is modelled by using the velocity of the helicopter and the induced air velocity in the ${}^{B}z$ direction. Note that the velocity of the helicopter has the opposite sign of the air velocity:

$$ {}^{B}\boldsymbol{D}_{fu} = \begin{bmatrix} -\frac{\rho}{2}\, d_x\, A_x\, |{}^{B}\dot{x}|{}^{B}\dot{x} \\ -\frac{\rho}{2}\, d_y\, A_y\, |{}^{B}\dot{y}|{}^{B}\dot{y} \\ -\frac{\rho}{2}\, d_z\, A_z\, |{}^{B}\dot{z} - v_i|({}^{B}\dot{z} - v_i) \end{bmatrix} \quad . \tag{5.45} $$

Front plane drag

$$ D_{fp} = -\frac{\rho}{2}\, d_{fp}\, A_{fp}\, |{}^{B}\dot{z}_{fp} - v_i|\, ({}^{B}\dot{z}_{fp} - v_i) \quad . \tag{5.46} $$

- ${}^{B}\dot{z}_{fp} = {}^{B}\dot{z} + \dot{\phi}\kappa_{y,fp} - \dot{\theta}\kappa_{x,fp}$ is the z-axis velocity of the front plane.
- $v_{i,tr}$ is the induced air velocity of the tail rotor.

Tail fin and plane drag are calculated the same way as for the fuselage, however the velocity is a function of both the rotation and velocity of the helicopter:

$$ D_{tf} = -\frac{\rho}{2}\, d_{tf}\, A_{tf}\, |{}^{B}\dot{y}_{tr} - K_{tf}v_{i,tr}|\, ({}^{B}\dot{y}_{tr} - K_{tf}v_{i,tr}) \quad , \tag{5.47} $$

$$ D_{tp} = -\frac{\rho}{2}\, d_{tp}\, A_{tp}\, |{}^{B}\dot{z}_{tp} - v_i|\, ({}^{B}\dot{z}_{tp} - v_i) \quad . \tag{5.48} $$

- K_{tf} is a constant accounting for that the tail rotor does not cover the whole tail fin.
- ${}^{B}\dot{y}_{tr} = {}^{B}\dot{y} - \dot{\phi}\kappa_{z,tr} + \dot{\psi}\kappa_{x,tr}$ is the y-axis velocity of the tail rotor.
- ${}^{B}\dot{z}_{tp} = {}^{B}\dot{z} + \dot{\phi}\kappa_{y,tp} - \dot{\theta}\kappa_{x,tp}$ is the z-axis velocity of the tail plane.
- $v_{i,tr}$ is the induced air velocity of the tail rotor.

Forces and torques are calculated by summing and using the distance to the point where the drag is acting. It is assumed that the fuselage drag acts in the CM:

$$ {}^{B}\boldsymbol{F}_d = \begin{bmatrix} {}^{B}F_{x,d} \\ {}^{B}F_{y,d} \\ {}^{B}F_{z,d} \end{bmatrix} = \begin{bmatrix} D_{x,fu} \\ D_{y,fu} + D_{tf} \\ D_{z,fu} + D_{tp} \end{bmatrix} \quad , \tag{5.49} $$

$$ {}^{B}\boldsymbol{\tau}_d = \begin{bmatrix} {}^{B}\tau_{x,d} \\ {}^{B}\tau_{y,d} \\ {}^{B}\tau_{z,d} \end{bmatrix} = \begin{bmatrix} \kappa_{y,tp}\, D_{tp} - \kappa_{z,tf}\, D_{tf} \\ -\kappa_{x,tp}\, D_{tp} \\ \kappa_{x,tf}\, D_{tf} \end{bmatrix} \quad . \tag{5.50} $$

- K_{tf} is the position of th tail fin in $\{B\}$.

5.1.7 Force and Torque Summation

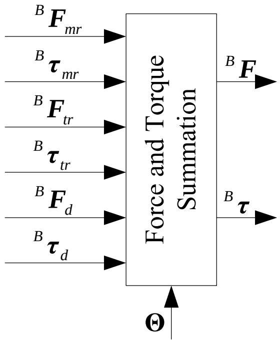


Figure 5.15: *The force and torque summation block of the model taken from Figure 5.1 on page 40.*

To get the total effect on the body of the helicopter, the forces and torques are summed. Concerning the forces, the gravity $^B\boldsymbol{F}_g = {}^B_E\mathcal{R}\,{}^E\boldsymbol{F}_g$ is also added:

$$^B\boldsymbol{F} = {}^B\boldsymbol{F}_{mr} + {}^B\boldsymbol{F}_{tr} + {}^B\boldsymbol{F}_{d} + {}^B\boldsymbol{F}_{g} \tag{5.51}$$

$$^B\boldsymbol{\tau} = {}^B\boldsymbol{\tau}_{mr} + {}^B\boldsymbol{\tau}_{tr} + {}^B\boldsymbol{\tau}_{d} \tag{5.52}$$

5.1.8 Rigid Body Dynamics and Kinematics

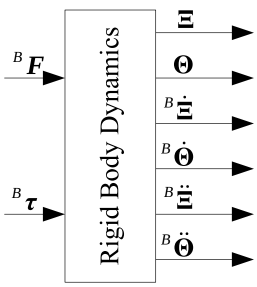


Figure 5.16: *The rigid body dynamics block of the model taken from Figure 5.1 on page 40.*

The total sum of forces and torques on the helicopter is now used to calculate the movement of the helicopter by use of Newton's second law and Euler's rotational equations. The rotation of the helicopter is described by use of 3-2-1 Euler angles, which gives the rotation from the $\{E\}$ to the $\{B\}$ as:

$$^B_E\mathcal{R} = C_x(\phi)\,C_y(\theta)\,C_z(\psi) = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\psi & s\phi c\theta \\ c\phi s\theta c\psi + s\phi s\psi & c\phi s\theta s\psi - s\phi c\psi & c\phi c\theta \end{bmatrix} \quad . \tag{5.53}$$

- $^B_E\mathcal{R}$ is the rotation matrix from $\{E\}$ to the $\{B\}$.
- $C_x(\phi)$ is the direction cosine matrix describing a rotation around the x-axis.
- $C_y(\theta)$ is the direction cosine matrix describing a rotation around the y-axis.
- $C_z(\psi)$ is the direction cosine matrix describing a rotation around the z-axis.

- $c\cdot$ and $s\cdot$ is used as short notation for $\cos(\cdot)$ and $\sin(\cdot)$.

Now since the inertia of the helicopter is constant, then Euler's equation for rotation [Craig, 2005, p. 172] can be used to calculate the angular acceleration.

$$\begin{aligned} {}^{B}\boldsymbol{\tau} &= \mathcal{I}\ {}^{B}\ddot{\boldsymbol{\Theta}} + {}^{B}\dot{\boldsymbol{\Theta}} \times (\mathcal{I}\ {}^{B}\dot{\boldsymbol{\Theta}}) \quad , \\ &\Updownarrow \\ {}^{B}\ddot{\boldsymbol{\Theta}} &= \mathcal{I}^{-1}\left({}^{B}\boldsymbol{\tau} - {}^{B}\dot{\boldsymbol{\Theta}} \times (\mathcal{I}\ {}^{B}\dot{\boldsymbol{\Theta}})\right) \quad . \end{aligned} \tag{5.54}$$

- $\mathcal{I}$ is the inertia tensor of the helicopter.
- $\boldsymbol{\tau}$ is the total torque from (5.52).
- ${}^{B}\ddot{\boldsymbol{\Theta}}$ is the angular acceleration of the helicopter.
- ${}^{B}\dot{\boldsymbol{\Theta}}$ is the angular velocity of the helicopter.

Similarly the translational acceleration can be calculated:

$${}^{B}\ddot{\boldsymbol{\Xi}} = \frac{{}^{B}\boldsymbol{F}}{M} - {}^{B}\dot{\boldsymbol{\Theta}} \times {}^{B}\dot{\boldsymbol{\Xi}} \quad . \tag{5.55}$$

- M is the mass of the helicopter.
- ${}^{B}\boldsymbol{F}$ is the force affecting the helicopter given in $\{B\}$.
- ${}^{B}\dot{\Xi}$ is the velocity of the helicopter given in $\{B\}$.

The reason for the last term in the equation is as follows. It is wanted to find the change of the velocity seen from the body frame of the helicopter. For example, if the gravity is disregarded and the helicopter is performing a roll motion with constant velocity seen from a human perspective (the earth frame), ${}^{E}\dot{y} = constant$. Then after half a revolution ${}^{B}\dot{y}$ would be negative without any force has influenced the motion of the helicopter. This effect can be calculated by the transport theorem:

$$\left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{E} = \left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{B} + {}^{B}\dot{\boldsymbol{\Theta}} \times {}^{B}\dot{\boldsymbol{\Xi}} \tag{5.56}$$

- $\left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{E}$ is the acceleration as seen from $\{E\}$, but given in $\{B\}$ co-ordinates.
- $\left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{B}$ is the acceleration as seen from $\{B\}$, and given in $\{B\}$ co-ordinates.

By rearranging, the result in (5.55) can be obtained:

$$\begin{aligned} &\left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{E} = \left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{B} + {}^{B}\dot{\boldsymbol{\Theta}} \times {}^{B}\dot{\boldsymbol{\Xi}} \\ \Updownarrow \quad & \\ &\frac{{}^{B}\boldsymbol{F}}{M} = \left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{B} + {}^{B}\dot{\boldsymbol{\Theta}} \times {}^{B}\dot{\boldsymbol{\Xi}} \\ \Updownarrow \quad & \\ &\left\{{}^{B}\ddot{\boldsymbol{\Xi}}\right\}_{B} = \frac{{}^{B}\boldsymbol{F}}{M} - {}^{B}\dot{\boldsymbol{\Theta}} \times {}^{B}\dot{\boldsymbol{\Xi}} \\ \Updownarrow \quad & \\ &{}^{B}\ddot{\boldsymbol{\Xi}} = \frac{{}^{B}\boldsymbol{F}}{M} - {}^{B}\dot{\boldsymbol{\Theta}} \times {}^{B}\dot{\boldsymbol{\Xi}} \end{aligned} \tag{5.57}$$

By integration, ${}^{B}\dot{\mathbf{\Theta}}$ and ${}^{B}\dot{\mathbf{\Xi}}$ is obtained, and by use of the rotation matrix ${}^{E}_{B}\mathcal{R} = {}^{B}_{E}\mathcal{R}^{T}$ transformed to $\{E\}$. Finally, the result is integrated again to obtain the global attitude and position $\mathbf{\Theta}$ and $\mathbf{\Xi}$. A summary of all the formulas can be found in [Hald et al., 2006, app. D].

5.2 Modifications

The Corona 120 does not entirely function as the Bergen twin helicopter, which is the basis for the existing model described in the previous section. Most of the differences is accounted for in the parameters, and does not affect the model itself. However, some modifications of the model are needed to adapt it to the Corona 120. The differences in the operation of the two helicopter types concern:

Main Rotor and Stabilizer Bar Rotation The main rotor and stabilizer bar turns the other way around i.e. in a counterclockwise direction when seen from above ($\dot{\Psi}$ is negative).

Tail rotor thrust The tail rotor is mounted on the other side of the tail boom, and hence the thrust vector is pointing in the opposite direction. But the rotor still turns clockwise seen from the right hand side, just as on the Bergen helicopter, so the blade drag provided by the tail rotor blades is the same.

No collective pitch The main rotor thrust is not governed by collective pitch, but only by the angular velocity of the main rotor.

Wire attached The wires, which are attached in the nose of the helicopter, affects the motion of the helicopter.

Stabilizer bar collective pitch The stabilizer bar has collective pitch.

Furthermore a gyro is implemented on the Corona helicopter. This is not different from the Bergen helicopter, and is already implemented in the model. But the gyro is not described in Hald et al. [2006], and will therefore be described as well as the other modifications.
In the following subsections, each change of model is described in detail. Some variables are common for both the existing Bergen model, and for the Corona model. To be able to distinguish between these variables, they are denoted by the subscript B and C, for the Bergen and Corona model, respectively.

5.2.1 Main Rotor and Stabilizer Bar Rotation

This adaptation of the model has proved the most comprehensive to study, since the whole model has to be reviewed. The obvious first impulse is just to change Ψ_B to $-\Psi_C$ in the calculation, because the rotor turns the opposite way. By looking at a simplified pitch angle θ_b only concerning actuator input (obtained from Equation (5.12)), it is easily proved that this method is not correct:

$$\theta_{b,B} = \theta_0 - A_{1,mr}\cos(\Psi) - B_{1,mr}\sin(\Psi) \qquad . \tag{5.58}$$

- $\theta_{b,B}$ is the blade pitch for the Bergen helicopter.

- ► θ_0 is the collective pitch.
- ► $A_{1,mr}$ is the lateral pitch angle.
- ► $B_{1,mr}$ is the longitudinal pitch angle.

For the correct formula, the collective pitch must be the same for the two models. For the last two terms a blade at the same angle for both models ($\Psi_B = \Psi_C$) must have opposite signs. This is because the blade would have exactly the same orientation, but the leading edge of the blade is in the opposite direction (see Figure 5.17). Hence the blade pitch for the Corona helicopter must be:

$$\theta_{b,C} = \theta_0 + A_{1,mr}\cos(\Psi) + B_{1,mr}\sin(\Psi) \quad . \tag{5.59}$$

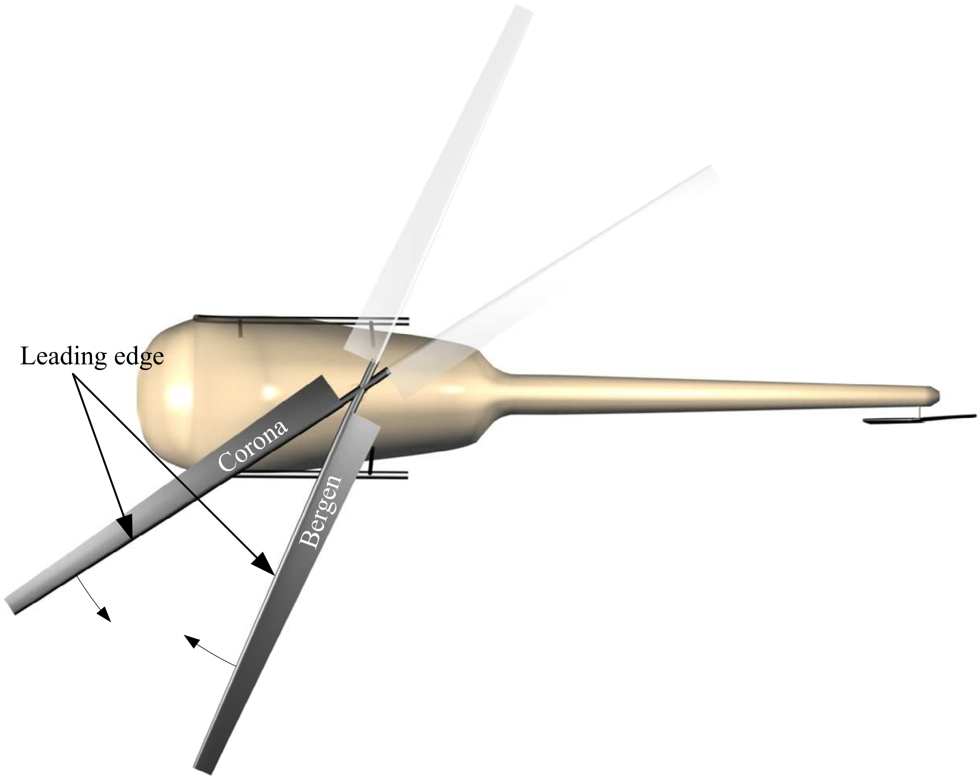


Figure 5.17: *The leading edge of the rotor blade for each helicopter is in the opposite direction.*

If $\Psi_B = -\Psi_C$ is inserted into (5.58) to change the model, this would yield:

$$\theta_0 - A_{1,mr}\cos(-\Psi_C) - B_{1,mr}\sin(-\Psi_C) = \theta_0 - A_{1,mr}\cos(\Psi) + B_{1,mr}\sin(\Psi) \quad . \tag{5.60}$$

Since (5.59) and (5.60) are obviously not equal, it would yield a wrong result just to change the sign of Ψ as suggested.
An interesting property is that the derivative with respect to time is the same for both θ_B and θ_C because $\frac{d\Psi_C}{dt} = -\Omega_{mr}$:

$$\dot{\theta}_{b,C} = \dot{\theta}_{b,B} = \Omega_{mr}A_{1,mr}\sin(\Psi) - \Omega_{mr}B_{1,mr}\cos(\Psi) \quad . \tag{5.61}$$

This means that if it is possible to find a Ψ_0 where $\theta_{b,C}(\Psi_0) = \theta_{b,B}(\Psi_0)$, then the blade pitch of the two helicopters will behave symmetrically about this angle because the rotation is opposite (see Figure 5.18). For example after the blade has turned ν degrees: $\theta_{b,B}(\Psi_0 + \nu) = \theta_{b,C}(\Psi_0 - \nu)$.

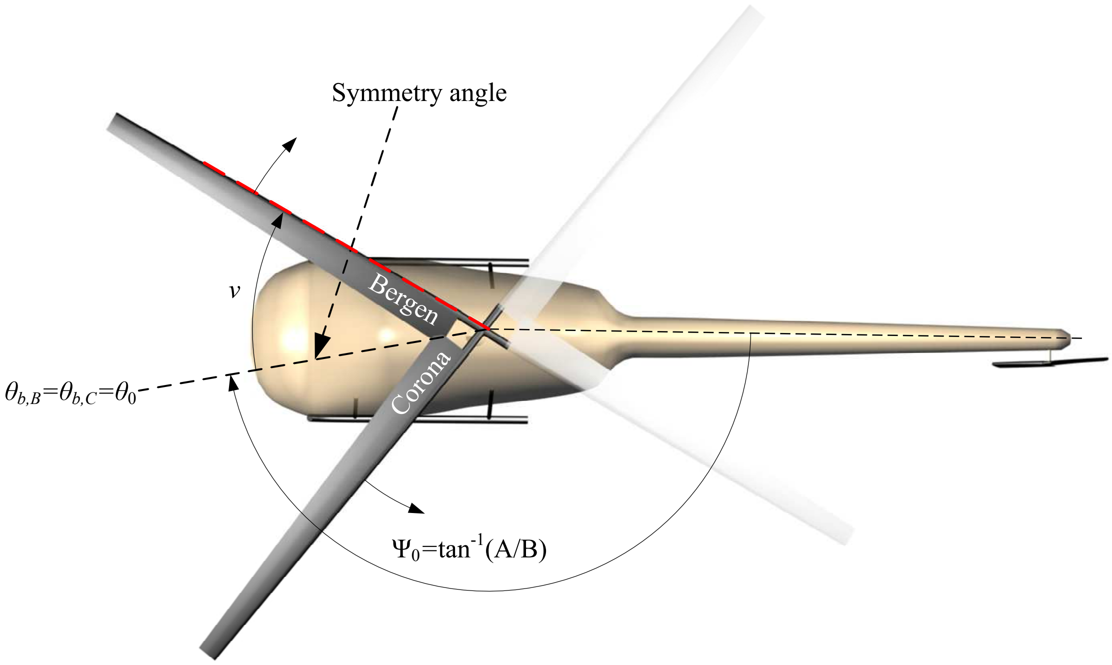


Figure 5.18: *The symmetry angle Ψ_0 where $\theta_{b,B}(\Psi_0 + \nu) = \theta_{b,C}(\Psi_0 - \nu)$ is shown.*

The blade pitch will be the same where it is equal to the collective pitch, i.e. where the cyclic pitch is zero. That means if

$$
\begin{aligned}
A_{1,mr}\cos(\Psi_0) &= -B_{1,mr}\sin(\Psi_0) \quad , && (5.62)\\
\Leftrightarrow \frac{A_{1,mr}}{B_{1,mr}} &= -\frac{\sin(\Psi_0)}{\cos(\Psi_0)} \quad , && (5.63)\\
\Leftrightarrow \Psi_0 &= \arctan\left(-\tfrac{A_{1,mr}}{B_{1,mr}}\right) \quad , && (5.64)
\end{aligned}
$$

then $\theta_{b,B}(\Psi_0 + \nu) = \theta_{b,C}(\Psi_0 - \nu)$, where ν is an arbitrary angle.
But this is not the only symmetry. For a difference of 180° the blade pitch of the two models are also equal:

$$\theta_{b,B}(\Psi{+}180) = \theta_0{-}A_{1,mr}\cos(\Psi{+}180){-}B_{1,mr}\sin(\Psi{+}180) = \theta_0{+}A_{1,mr}\cos(\Psi){+}B_{1,mr}\sin(\Psi) = \theta_{b,C}\,, \quad (5.65)$$

This is shown in Figure 5.19 on the next page.

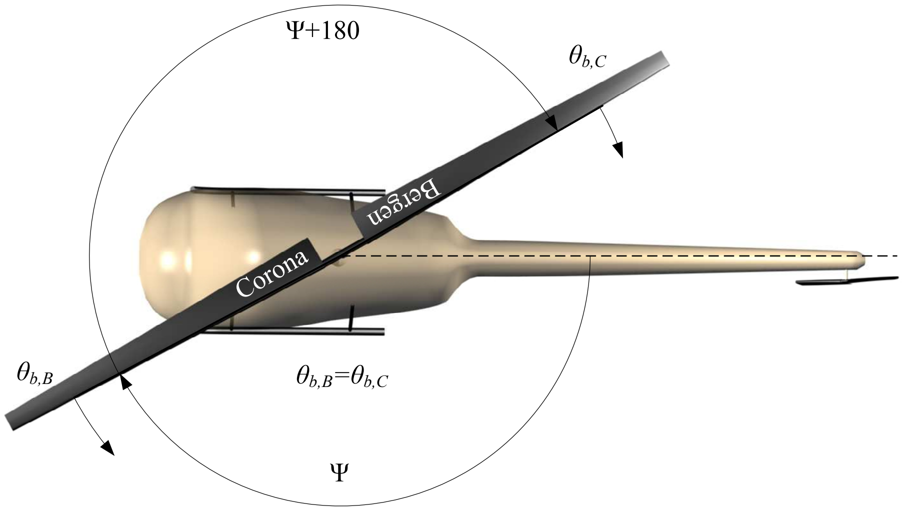


Figure 5.19: *The same blade pitch angle will be obtained at an* 180° *offset.*

From the above it seems likely that changing the signs of A and B in the input to the model might yield the right model. This would also give the wrong result because this would mean that the two helicopters behaves opposite to each other if the same input is given (eg. if a lateral input is given, then one helicopter would move to the left and the other would move to the right), which of course is not true.
It is also not correct that because the helicopters make the same maneuvers corresponding to the same input, the models must be the same. There are differences, and why that is the fact will be evident in the following sections.
If the model structure is considered (see Figure 5.3 on page 42), then the blade pitch affects the blade element lift, which again affects flapping dynamics, forces and torques on the helicopter. Each of them will be described in the following sections.

Flapping Dynamics

As mentioned in the previous section, the blade pitch is symmetric about the angle Ψ_0, and according to [Prouty, 1990, p. 151] the blade will exhibit a 90° lag between input and response, which means a 90° lag between the maximum aerodynamic input and the maximum flapping angle. This means that the maximum flapping angle occurs at the angle Ψ_0 and hence the flapping is symmetric about Ψ_0 as well. The conclusion to this symmetry is that the flapping remains the same for both rotation directions, and can therefore be leaved unchanged in the model. This is the case for both the main rotor and the stabilizer bar flapping.

Forces

Considering the forces, it is difficult to explain and visualize how the reverse rotor rotation affects the dynamics of the helicopter, instead of looking at the mathematical equations, the differences are elucidated. Only the effect on ${}^{H}F_{z,mr}$ is described, but the effects on ${}^{H}F_{x,mr}$ and ${}^{H}F_{y,mr}$ are similar though a bit more extensive to derive because of the direction of the lift. By using small angle approximation for the direction of the lift in (5.15), it is seen that $F_{z,mr} = -L$. To get the mean force acting in the hub, this is integrated around a revolution and multiplied by the number of blades (the lift is given in Equation (5.10)):

$$^{H}F_{z,mr} = -\frac{b}{2\pi}\frac{\rho}{2}\,C_{ls}\int_{0}^{2\pi}\int_{e}^{R} U_t^2\left(\theta_b + \frac{U_p}{U_t}\right)c\,dr\,d\Psi \quad . \tag{5.66}$$

To be able to evaluate the changes caused by reverse rotor direction, it is easiest to split this equation into two integrations:

$$^{H}F_{z,mr} = -\frac{b}{2\pi}\frac{\rho}{2}\,C_{ls}\left[\int_{0}^{2\pi}\int_{e}^{R} U_t^2\,\theta_b\,c\,dr\,d\Psi + \int_{0}^{2\pi}\int_{e}^{R} U_t\,U_p\,c\,dr\,d\Psi\right] \quad , \tag{5.67}$$

where U_t, the tangential velocity of the air illustrated in Figure 5.6 on page 45, is approximated by:

$$U_t \approx \Omega R\left(\frac{e+r}{R} + \mu_x\sin(\Psi) - \mu_y\cos(\Psi)\right) \quad . \tag{5.68}$$

Because of the opposite direction of the leading edge of the blade for the Corona helicopter, the effect from the wind has opposite sign:

$$U_{t,C} \approx \Omega R\left(\frac{e+r}{R} - \mu_x\sin(\Psi) + \mu_y\cos(\Psi)\right) \quad . \tag{5.69}$$

This equation exhibits the same properties as the equation for the blade pitch, namely that there is symmetry both around the angle from where the wind comes and 180°. When looking at the first integral in Equation (5.67) it is seen that the multiplication $U_t^2\,\theta_b$ will equal $U_{t,C}^2\,\theta_{b,C}$ for an angle offset of 180°. Since the integration is performed around a whole revolution, the result will be exactly the same for both models.

The same reasoning can be done for the second integral. However, here the integral has to be split up further. Then both the properties of symmetry around the wind direction and 180° offset must be applied to get to the same result, namely that the forces remain the same for both models. The same result can be derived for $^{H}F_{x,mr}$ and $^{H}F_{y,mr}$. Because of this, the model does not have to be changed when calculating the forces.

The inflow is calculated by using $F_{z,mr}$. Since this is the same for both models, the inflow is also the same and will therefore not have to be changed in the model.

Torques

The torque is calculated by using Equation (5.41). The torque consists of three parts as described in Section 5.1.6 on page 51; $\boldsymbol{\tau}_{hub}$, $\boldsymbol{\tau}_{res}$ and $\boldsymbol{\tau}_{aero}$. $\boldsymbol{\tau}_{hub}$ dependents on the forces, and since they do not change, $\boldsymbol{\tau}_{hub}$ does not change either. The torque originating from the restraint force from the blades does not change either, since this torque is only dependent on the flapping of the blade, which is not affected by the reverse rotor direction.

The only part of the model which is affected by the reverse rotor direction is the aerodynamic torque τ_{aero}, which is given by Equations (5.35) - (5.37). Comparing these equations with the force calculation (Equations (5.30) - (5.32)), it shows that a multiplication by e and $\sin(\Psi)$ or $\cos(\Psi)$ is performed in the torque calcutation. This means that the symmetry considerations is not applicable any more, and hence the two different models does not provide identical results. The result is that the equations must be re-derived and inserted into the model. The changes are as follows:

Blade pitch: The change of the blade pitch ($\theta_{b,C}$) is given in Equation (5.59) without the Bell-Hiller gains, blade twist and cross-coupling factor. These are obtained from Equation (5.12).

$$\theta_b = \theta_0 + A_{1,mr}\cos(\Psi) + B_{1,mr}\sin(\Psi) + \theta_{tw}\frac{e+r}{R} - K_1\beta \quad . \tag{5.70}$$

Tangential velocity:

$$U_{t,B} \approx \Omega R\left(\frac{e+r}{R} + \mu_x\sin(\Psi) - \mu_y\cos(\Psi)\right) \quad , \tag{5.71}$$

$$\Downarrow$$

$$U_{t,C} \approx \Omega R\left(\frac{e+r}{R} - \mu_x\sin(\Psi) + \mu_y\cos(\Psi)\right) \quad . \tag{5.72}$$

Perpendicular velocity: The perpendicular velocity is the same for both models.

Flapping derivative

$$\beta_B = \beta_C = a_0 - a_1\cos(\Psi) + b_1\sin(\Psi) \quad , \tag{5.73}$$

$$\Downarrow$$

$$\dot{\beta}_B = -(\dot{a}_1 - b_1\Omega_{mr})\cos(\Psi) + (\dot{b}_1 + a_1\Omega_{mr})\sin(\Psi) \quad , \tag{5.74}$$

$$\Downarrow$$

$$\dot{\beta}_C = -(\dot{a}_1 + b_1\Omega_{mr})\cos(\Psi) + (\dot{b}_1 - a_1\Omega_{mr})\sin(\Psi) \quad . \tag{5.75}$$

$$\tag{5.76}$$

5.2.2 Tail Rotor Thrust

The tail rotor on the Corona helicopter is on the other side of the tail boom and the thrust is therefore in the opposite direction. The opposite thrust is implemented by changing the thrust force provided by the tail rotor (given in Equation (5.42)) from $F_{y,tr,B} = -T_{tr}$ to $F_{y,tr,C} = T_{tr}$. Further the airflow providing drag on the tail fin is in the opposite direction as well. Therefore in Equation (5.47) $v_{i,tr,B}$ is changed to $-v_{i,tr,C}$.

5.2.3 No Collective Pitch

The main rotor thrust is not governed by collective pitch, but by the angular velocity of the main rotor. This is already implemented in the model since both collective pitch and angular velocity are input to the model, apart from that no dynamics are implemented on the angular velocity. In the model implementation, the dynamic equations used to calculate the collective pitch dynamics are changed to be used to calculate the main rotor rotation dynamics. This also means that the state θ_0 in the state vector (5.1) has to be switched with a state for the main rotor velocity, Ω_{mr}. The same second order model as for the servo motors is used for this motor.

5.2.4 Wire Attached

The wire is considered to drag in the direction ${}^{E}\boldsymbol{z}$ by a force equal to the weight of the wire lifted from the ground, i.e. proportional to the height of the helicopter. The force and torque given in {B} can therefore be written:

$$
\begin{aligned}
{}^{B}\boldsymbol{F}_{w} &= {}^{B}_{E}\mathcal{R}\,{}^{E}\boldsymbol{F}_{w} = {}^{B}_{E}\mathcal{R}\,(M_{w} \cdot g \cdot (z_{floor} -{}^{E} z) && (5.77)\\
{}^{B}\boldsymbol{\tau}_{w} &= {}^{B}\boldsymbol{\kappa}_{w} \times {}^{B}\boldsymbol{F}_{w} && (5.78)
\end{aligned}
$$

- ${}^{B}\boldsymbol{F}_{w}$ is the force provided by the wires, given in {B}.
- ${}^{B}_{E}\mathcal{R}$ is the rotation matrix mapping {E} to {B}.
- M_{w} is the mass of the wires given in [kg/m].
- $-{}^{E}z$ is the current altitude of the helicopter, given in $\{E\}$.
- z_{floor} is the distance from the origin of $\{E\}$ to the ground level.
- ${}^{B}\boldsymbol{\tau}_{w}$ is the torque provided by the wires, given in {B}.
- ${}^{B}\boldsymbol{\kappa}_{w}$ is the distance from the CM to where the wires are attached.

This force and torque can now be added in the force and torque summation given in Equations (5.51) and (5.52).

5.2.5 Stabilizer Bar Collective Pitch

The stabilizer bar has a constant collective pitch. The flapping of the stabilizer bar is calculated the same way as for the main rotor, except there is no restraint torque (see Section 5.1.5 on page 50). Therefore collective pitch can be included the same way, as when calculating the flapping dynamics of the main rotor blades, by adding a constant pitch ($\theta_{0,sb}$) to $\theta_{b,sb}$, as done for the main rotor in Equation (5.12).

5.2.6 Gyro

A gyro helps the pilot control the fast dynamics of the yaw rate of the helicopter. The gyro is implemented as a P-controller, with a gain K_{gyro}, for the yaw rate. This means that instead of controlling the pitch of the tail rotor, the pilot actually controls the yaw rate of the helicopter. This is illustrated in Figure 5.20 in comparison with the model given in Section 5.1.4 on page 47.

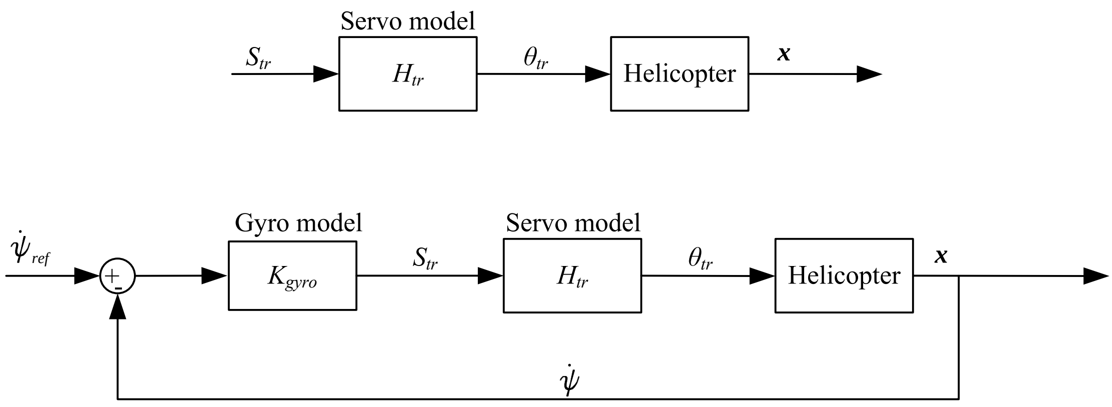


Figure 5.20: *Implementation of a gyro makes the pilot control the yaw rate of the helicopter instead of the pitch of the tail rotor.*

5.3 Implementation of the Model

The model is implemented, as described in [Hald et al., 2006, cha. 10], in MATLAB Simulink using ANSI C and C++ in S-functions. The equations which are changed, are re-derived in MAPLE and inserted into the model. The changes are implemented in a generic way, such that it is possible to choose between the Bergen model or the Corona model, just by changing a flag in the code. For integration of the model derivative, a fourth order Runge-Kutta method is used:

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4 \tag{5.79}$$

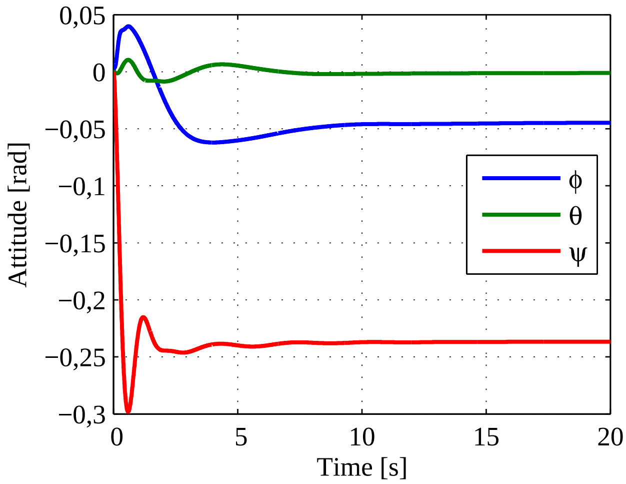
- k_x are intermediate derivatives multiplied by the step size: $k_x = hf(t,\boldsymbol{x})$.
- $\boldsymbol{x}_{n+1}$ is the next step iteration of the model.

The new formulas derived in Section 5.2.1 on page 63 are inserted into the following MAPLE scripts and the torques are re-derived and inserted into the model.
[CD-ROM, 2007, `./source/maple/Taux_reverse_rotor.mw`]
and
[CD-ROM, 2007, `./source/maple/Tauy_reverse_rotor.mw`]

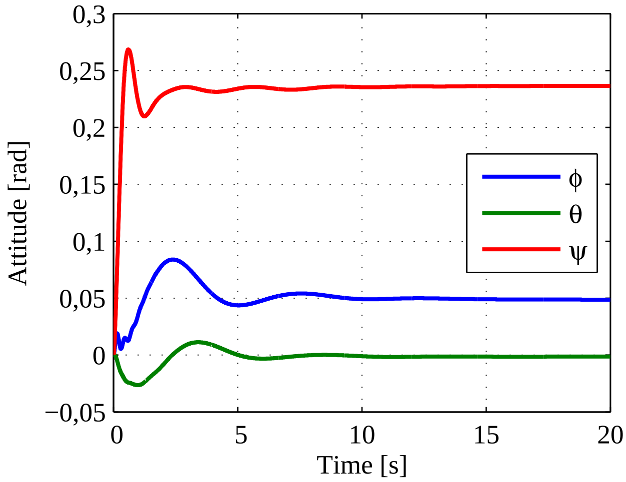
For easier accessibility for future projects, an overview of the implementation of the changes and the model itself is given in Appendix C on page 179.

5.4 Model Verification

The original model is verified in [Hald et al., 2006, cha. 10] using qualitative considerations on which input should cause which effects. This method is based on [Prouty, 1990, p. 444]. A simple example is that an increased lateral input should cause a decreased swash plate angle A_1, an increased flapping angle b_1 and a positive roll $\dot{\phi}$ motion (${}^{H}F_{y,mr}$ and ${}^{H}\tau_{x,mr}$ increases) -see Table 3.3. The same method is used for verifying the adapted model, which for some input yields opposite results due to the counter clockwise rotation. The verification has shown that the model behaves as expected.
Due to the tail rotor thrust forcing the helicopter to the side, the helicopter will have a small roll angle when in steady state hover. When the main rotor is turning the other way around the tail rotor thrust will be in the opposite direction as well. This means that at steady state hover the helicopter roll angle will have equal magnitude, but opposite sign. When using the same controller, this will also be the case for the yaw angle. The pitch angle should remain the same. A test of this has been conducted using the parameters for the Bergen helicopter and a simple P-controller to stabilize the helicopter. The results is shown in Figure 5.21, where it can be seen that the helicopter attitude is opposite for ϕ and ψ, but remains the same for θ as expected. Some oscillation is seen in the beginning. This is due to the simple non-tuned P-controller which is used to stabilize the helicopter in hover.



(a) **Θ** for steady state counter clockwise rotation of the main rotor like the Corona helicopter.



(b) $\dot{\mathbf{\Theta}}$ for steady state clockwise rotation of the main rotor like the Bergen helicopter.

Figure 5.21: *Simulation of helicopter attitude at steady state hover for both clockwise and counter clockwise rotation of the main rotor. It can be seen that the ϕ and ψ angles have opposite sign as expected.*

5.5 Parameter Determination

The Corona helicopter model contains 78. To be able to simulate the helicopter model, it is necessary to identify these parameters. Some parameters can be directly measured, and some has to be estimated through various experiments. This section describes the procedures of measuring and identifying these parameters. In each subsection the relevant parameters are listed in a table. For a full list of all the parameters in alphabetical order, see the nomenclature table (Table 3 on page XXI).

5.5.1 General Parameters

The model contains some general parameters listed below.

ρ	Density of air	1,29 $^{kg}/_{m^3}$
b	Number of main rotor blades	2 [·]
b_{tr}	Number of tail rotor blades	2 [·]
g	Gravitational acceleration	9,82 $^{m}/_{s^2}$
Ts	Sampling time	0,005 s

5.5.2 Location of Center of Mass

Before measuring distance parameters on the helicopter, it is necessary to locate the CM, because the CM serves as the reference point and the body frame origin. This has been done by hanging the helicopter freely in a single wire, and then taking a picture of the helicopter in its rest position (see Figure 5.22). The CM will then be located somewhere on the straight line coinciding with the wire. By repeating this procedure with the wire tied to another mount point on the helicopter, it will in theory be possible to find the CM where the line extensions coincide. However, when drawing a straight line on a two dimensional image, as done in Figure 5.22, it is not possible to determine the position of the line in the three dimensional space. In fact, the line can be regarded as a two dimensional plane, with only the edge visible in the three dimensional space. In Figure 5.23 the helicopter has been hung in another mount point, and a yellow line is now representing the new plane. The red line is the plane from before, rotated with the helicopter. The CM is now placed somewhere on the straight line (perpendicular to this page), where the yellow and the red plane is intersecting. This line is coloured cyan in Figure 5.24, where the helicopter has been rotated again, and a new, orange plane is superimposed. The position of the CM can be determined to be where the cyan line and the orange plane intersects. A mark is placed in the CM of the helicopter, and this serves now as the origin of $\{B\}$.

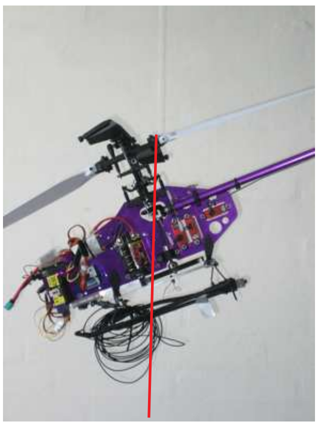


__Figure 5.22:__ The helicopter hanging in a single wire, which has been extended with a red line. The red line can be regarded as a plane seen from the edge. The CM will be located somewhere in this plane.

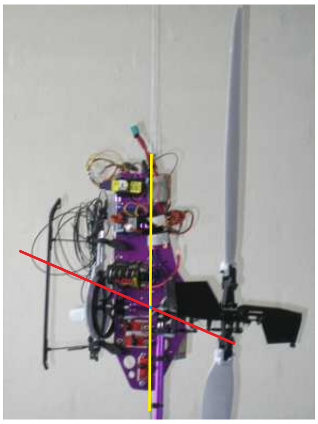


Figure 5.23: *Second image with the red plane from Figure 5.22 superimposed. The CM will be located somewhere on the straight line where the red and yellow plane intersects. This line is coloured cyan in Figure 5.24*

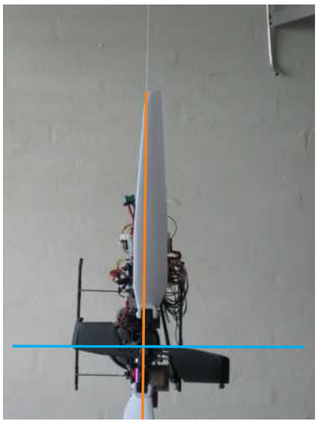


Figure 5.24: *Third image with the cyan line derived from the second image superimposed. The CM is located in the intersection of the orange plane and the cyan line.*

5.5.3 Distances

The distance parameters have been measured with a sliding gauge or a ruler, using centimeters as measuring resolution.

${}^{B}\boldsymbol{\kappa}_{fp}$	Position of CM of front plane[3] in $\{B\}$	$\begin{bmatrix} 0{,}04 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{hcam}$	Position of helicam in $\{B\}$	$\begin{bmatrix} -0{,}06 \\ -0{,}01 \\ 0{,}08 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{hub}$	Position of hubframe origin in $\{B\}$	$\begin{bmatrix} -0{,}01 \\ -0{,}01 \\ -0{,}11 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{IMU}$	Position of IMU in $\{B\}$	$\begin{bmatrix} 0{,}04 \\ 0{,}01 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{tf}$	Position of tail fin CM in $\{B\}$	$\begin{bmatrix} -0{,}47 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{tp}$	Position of tail plane CM in $\{B\}$	$\begin{bmatrix} -0{,}30 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{tr}$	Position of tail rotor frame in $\{B\}$	$\begin{bmatrix} -0{,}49 \\ -0{,}04 \\ -0{,}02 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{w}$	Position of to where the wires are attached to the helicopter given the in $\{B\}$	$\begin{bmatrix} 0{,}23 \\ 0{,}00 \\ 0{,}00 \end{bmatrix}$ m
c	Main rotor blade chord	0,05 m
c_{sb}	Stabilizer bar chord	0,05 m
c_{tr}	Tail rotor blade chord	0,03 m
e	Main rotor hinge offset	0,05 m
R	Main rotor radius	0,38 m
R_i	Inner radius of stabilizer bar	0,04 m
R_o	Outer radius of stabilizer bar	0,12 m
R_{tr}	Radius of tail rotor	0,09 m

5.5.4 Areas

All areas have been calculated from distance measurements or estimated. They have a resolution of square centimeters.

[3] Note that the word "front plane" can be misleading. It refers to the plane of the of the helicopter as seen from above excluding the tail boom (hence front plane). The plane is parallel to the ${}^{B}xy$-plane (see Figure 5.14 on page 55).

A_{fp}	Area of front plane (the area of the body in the ${}^{B}z$-direction exclusive the tail boom)	$0{,}0791\,\mathrm{m}^2$
A_{tf}	Area of tail fin	$0{,}0042\,\mathrm{m}^2$
A_{tp}	Area of tail plane (the area of the tail boom in the ${}^{B}z$-direction)	$0{,}0042\,\mathrm{m}^2$
A_x	Area of body in ${}^{B}x$ direction	$0{,}0072\,\mathrm{m}^2$
A_y	Area of body in ${}^{B}y$ direction	$0{,}0302\,\mathrm{m}^2$
A_z	Area of body in ${}^{B}z$ direction ($A_z = A_{fp} + A_{tp}$)	$0{,}0833\,\mathrm{m}^2$

5.5.5 Tail Rotor to Tail Fin Area Ratio

This parameter is used to compensate for the fact that the tail rotor disc does not completely cover the tail fin, when seen from the side:

$$K_{tf} = \frac{A_{tf,covered}}{A_{tf}}$$

- A_{tf} is the tail fin area
- $A_{tf,covered} = 0{,}0041\,\mathrm{m}^2$ is the tail fin area covered by the tail rotor disc.

K_{tf}	Factor compensating for that only a part of the tail fin is covering the tail thrust area	0,97 [·]

5.5.6 Masses

The masses has been determined with a scale using a measuring resolution of 10 g. The mass of the helicopter and the attached wire are the only mass parameters in the model, but it is necessary to determine masses of different parts of the helicopter in order to calculate their moments of inertia, see next section.

M	Helicopter mass	1,02 kg
M_w	Mass of attached wire pr. meter	0,10 kg/m

5.5.7 Moments of Inertia and Mass

The moment of inertias of the helicopter and the main rotor blade have been found using a method described in [Miller and Soulé, 1933]. The object is tied to a wire and used as a pendulum, see Figure 5.25. The moment of inertia around the pivot is then given by:

$$I_p = \frac{T^2 \cdot m \cdot g \cdot L}{4 \cdot \pi^2} \tag{5.80}$$

- T is the period of oscillation.
- m is the mass of the object.
- g is the gravitational acceleration.
- L is the distance from pivot to CM.

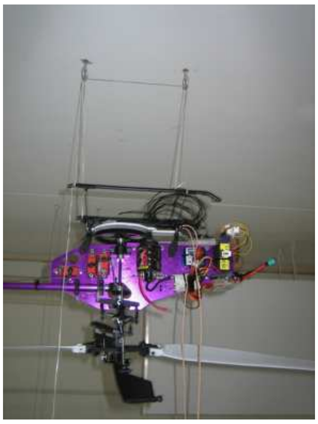


Figure 5.25: The helicopter hanging in a wire and swinging like a pendulum.

The moment of inertia with respect to the CM of the object can then be computed using the parallel-axis theorem:

$$I_{CM} = I_p - m \cdot L^2 \tag{5.81}$$

For the helicopter, this procedure is repeated two times. First, with respect to the ${}^{B}x$-axis and then with respect to the ${}^{B}y$-axis to obtain I_x and I_y. Finding the moment of inertia around the ${}^{B}z$-axis, the pendulum from Figure 5.25 is rotated such that the two wires twist around each other, and the object is rotating around the ${}^{B}z$-axis. Then the moment of inertia is given by

$$I_z = \frac{T^2 \cdot m \cdot g \cdot A^2}{16 \cdot \pi^2 \cdot l} \tag{5.82}$$

- T is the period of oscillation.
- m is the mass of the object.
- g is the gravitational acceleration.
- A is the distance between the mount point of the two wires.
- l is the length of the wires.

By assuming that the principal axes of the helicopter are coinciding with the body frame axes, the inertia matrix are given by

$$\mathcal{I} = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \tag{5.83}$$

The moment of inertia of the main rotor blade is only needed around the flapping pivot point, so Equation (5.80) is the one to use in this case.
The moment of inertia of one blade of the stabilizer bar has been calculated as a point mass. This is assumed to be sufficiently precise because the majority of the weight is in a metal screw in the outer most part of the bar.:

$$I_{sb} = r_{sb}^2 \cdot m_{sb} \tag{5.84}$$

- $r_{sb} = 0{,}12\,\mathrm{m}$ is the distance from the center of rotation to the point mass plus a small offset to account for the weight of the bar without the screw.
- $m_{sb} = 5{,}8\,\mathrm{g}$ is the mass of the screw.

The first mass moment of the main rotor blade is defined to be

$$m_b = \int r \, dm \tag{5.85}$$

From the definition of the CM, it is possible to find this quantity.

$$\begin{aligned} CM &= \tfrac{1}{M_{mr}} \int r \, dm \\ &\Updownarrow \\ \int r \, dm &= CM \cdot M_{mr} \\ &\Updownarrow \\ m_b &= CM \cdot M_{mr} \end{aligned} \tag{5.86}$$

- m_b is the first mass moment.
- $CM = 0{,}14\,\mathrm{m}$ is the distance from the hub to the CM of the main rotor blade.
- $M_{mr} = 0{,}04\,\mathrm{kg}$ is the mass of the main rotor blade.

$\mathcal{I}$	Inertia matrix of body	$\begin{bmatrix} 0{,}004415 & 0 & 0 \\ 0 & 0{,}024916 & 0 \\ 0 & 0 & 0{,}026528 \end{bmatrix} \mathrm{kg \cdot m^2}$
I_b	Inertia of main rotor blade	$0{,}001009\,\mathrm{kg \cdot m^2}$
$I_{b,sb}$	Inertia of stabilizer bar	$0{,}000084\,\mathrm{kg \cdot m^2}$
m_b	First mass moment of main rotor blade	$0{,}0053\,\mathrm{kg \cdot m}$

5.5.8 Gear Ratios

The gear ratios is found by counting the teeth on the gear wheels. The ratios are then given by:

$$G_1 = \frac{T_1}{T_2} \tag{5.87}$$

$$G_2 = \frac{T_3}{T_4} \tag{5.88}$$

- $T_1 = 102$ is the number of teeth on the gear wheel mounted at the end of the main rotor shaft.
- $T_2 = 9$ is the number of teeth on the gear wheel mounted on the DC motor.
- $T_3 = 21$ is the number of teeth on the bevel gear wheel mounted on the main rotor shaft.
- $T_4 = 10$ is the number of teeth on the bevel gear wheel mounted on the tail rotor shaft.

G_1	Gear ratio between DC motor and main rotor	$11{,}3\,[\cdot]$
G_2	Gear ratio between main rotor and tail rotor	$2{,}1\,[\cdot]$

5.5.9 Bell-Hiller Factors

The Bell-Hiller factors depend on the mechanical construction of the hub from the swash plate to the rotor blade/stabilizer bar.

$$K_b = \frac{a}{a+b} \cdot \frac{c}{b+c} \tag{5.89}$$

$$K_h = \frac{b}{b+c} \cdot \frac{a+b+c}{a+b} \tag{5.90}$$

- a = 1,3 cm is the horizontal distance from the hub to the connecting point from swash plate (see Figure 5.26)
- b = 0,8 cm is the horizontal distance from the swash plate connecting point to the main rotor connecting point
- c = 1,5 cm is the horizontal distance from the main rotor connecting point to the stabilizer bar connecting point

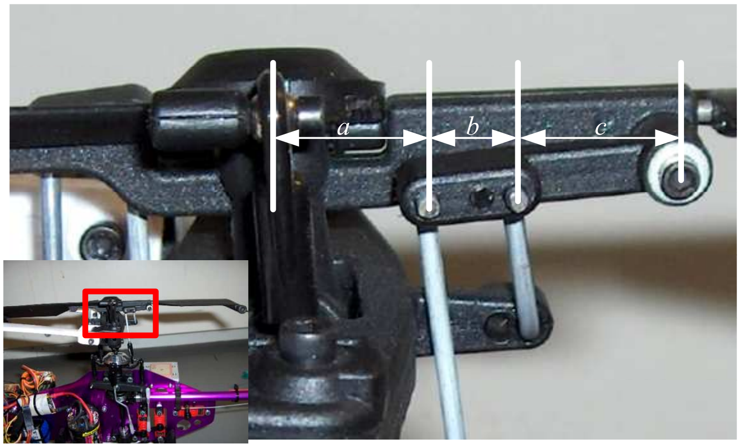


Figure 5.26: *Image showing a, b and c used to calculate the bell-hiller gains.*

When adding the bell and hiller gain, the result should yield 1:

$$K_b + K_h = \frac{a}{a+b} \cdot \frac{c}{b+c} + \frac{b}{b+c} \cdot \frac{a+b+c}{a+b} = 1 \tag{5.91}$$

K_b	Bell factor	0,41 [·]
K_h	Hiller factor	0,59 [·]

5.5.10 Gyro Gain

The gyro controls the tail pitch in order to stabilize the yaw velocity of the helicopter. It is necessary to know the gain from yaw velocity to the gyro contribution of the output pulse. To measure that the helicopter is mounted on a swivelling plate. While manually rotating the helicopter three different signals are logged: the input pulse to the gyro, the output pulse from the gyro, and the yaw velocity provided by the IMU. This experiment is done for four different input pulses, each at four different yaw velocities - i.e. sixteen experiments in all. The data is shown in Figure 5.27, and it can be seen that the gyro gain is not dependent on the input pulse width, only on yaw velocity. The relationship is approximately linear, and the gyro gain is the slope of the line.

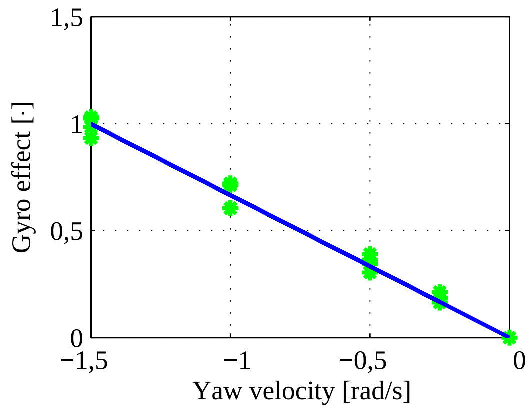


Figure 5.27: *Measurements of the gyro effect as a function of yaw velocity and different input signals.*

K_{gyro}	Gain from yaw velocity to gyro output	$-0{,}67$ s/rad

5.5.11 Servo Motor Parameters

Three servo motors of the type *MPX Tiny-S* are used as actuators on the helicopter. It is assumed that they can be modelled as a rate-limited second order system, as described in Section 5.1.4. The transfer functions are given as

$$H = \frac{\theta}{S} = \frac{K{\omega_n}^2}{s^2 + 2\zeta\omega_n s + {\omega_n}^2} \qquad . \tag{5.92}$$

The parameters has been found by applying a step to one of the servo motors, and measuring the time it takes to move from one extremity to the other. This has been done by filming a stopwatch with a resolution in milliseconds at the same time as the servo. To reduce errors of a single measurement, an average of four step times has been measured to 195 ms. A simulation of a rate-limited second order system is then fitted until it behaves approximately as the real system. Figure 5.28 shows the simulated step test output with the actual parameters implemented.

The transfer function describes the relation between the input to the servo motor, which is a unitless number in the range $[-1; 1]$, to the output, which is the swash plate lateral and longitudinal angles or tail rotor pitch angle in radians. Finding the gains of the lateral and longitudinal pitch servos involves geometrical analysis of the mechanical construction from the actuator position to the swash plate angle. Figure 5.29 shows the result of this analysis. In both cases the relation can be approximated with a linear function, where the slope of the line represents the DC gain of the servo.

Regarding the tail pitch servo, the relationship between actuator input and tail pitch has been measured by means of a laser pointer attached to the tail rotor blade. The result is shown in Figure 5.30.

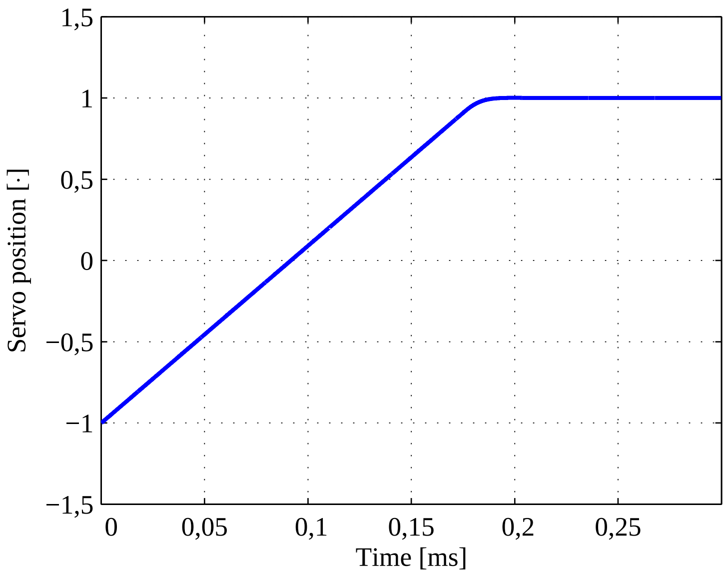
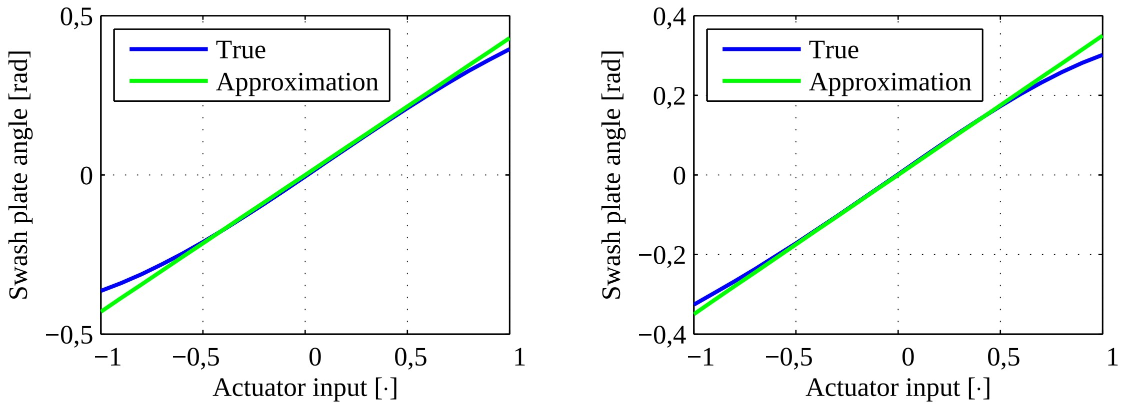


Figure 5.28: *A simulated step test. The input to the system is a step from* -1 *to* 1*, i.e. from one extremity to the other.*



(a) Lateral pitch servo.

(b) Longitudinal pitch servo.

Figure 5.29: *The relationship between actuator input and swash plate angle. The dashed line is the linear approximation of the true function.*

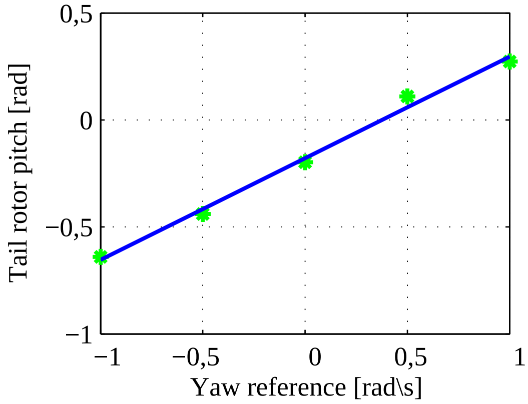


Figure 5.30: *The relationship between actuator input and tail rotor pitch angle. The dashed line is the linear approximation of the true function.*

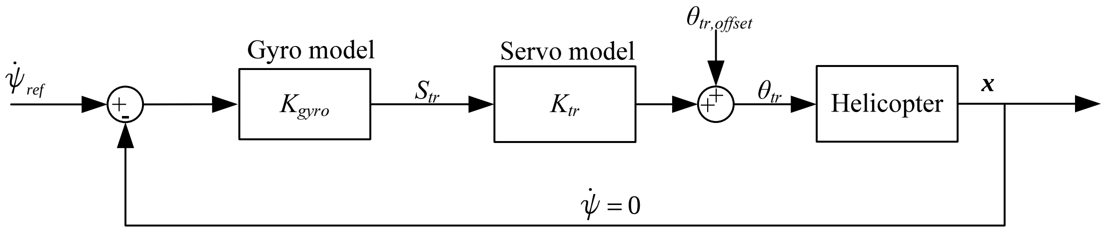


Figure 5.31: *Measurement of the tail servo gain. The input ψ_{ref} and the measured output angle $\theta_{tr} - \theta_{tr,offset}$ is measured in steady state. Therefore the transfer function H_{tr} is equal to the gain K_{tr}, and the actual measurement is $K_{gyro} \cdot K_{tr}$.*

The slope of the line in Figure 5.30 is the gain from the yaw reference ($\dot{\psi}_{ref}$) input to the gyro to the tail pitch angle, θ_{tr}. In Figure 5.31 a block diagram of the tail rotor control system is shown. It illustrates how the gyro affects the measurement. What has been measured is the yaw reference input to the gyro, $\dot{\psi}_{ref}$, and the tail pitch angle, θ_{tr}. The slope of the line in Figure 5.30 is the gain of the gyro and the gain of the tail pitch servo multiplied. Thus, in order to find the gain of the tail pitch servo, we must divide with K_{gyro}.

$$K_{tr} = \frac{K}{K_{gyro}} \tag{5.93}$$

- K_{tr} is the gain from the gyro output and to the tail pitch angle, i.e. the servo motor DC gain.
- $K = 0{,}48$ is the slope of the curve in Figure 5.30.
- K_{gyro} is the DC gain of the gyro found in the previous section.

Figure 5.31 also shows the offset in the tail pitch angle, $\theta_{tr,offset}$. This parameter is an offset in tail pitch angle, due to the fact that the tail pitch must generate a thrust to compensate for the drag from the main rotor blades, in order to keep the yaw angle of the helicopter constant. This parameter will be described later.

Notice, that due to the gyro, the controllable input to the tail pitch servo is an yaw rate reference in rad/s. The cyclic pitch servos has a unitless input in the range $[-1; 1]$, i.e. the saturation values for S_{lat} and S_{lon} are simply -1 and 1. S_{tr} must have the same saturation values, so the yaw reference input has the saturation values

$$\dot{\psi}_{ref,max} = \frac{1}{K_{gyro}} \tag{5.94}$$

and

$$\dot{\psi}_{ref,min} = -\frac{1}{K_{gyro}} \tag{5.95}$$

$\dot{\theta}_{max,tr}$	Rate limit of tail pitch servo	10,9 rad/s
$\dot{\theta}_{max,A}$	Rate limit of lateral pitch servo	10,9 rad/s
$\dot{\theta}_{max,B}$	Rate limit of longitudinal pitch servo	10,9 rad/s
$\omega_{n,tr}$	Undamped natural frequency of tail pitch servo	200 rad/s
$\omega_{n,A}$	Undamped natural frequency of lateral pitch servo	200 rad/s
$\omega_{n,B}$	Undamped natural frequency of longitudinal pitch servo	200 rad/s
ζ_{tr}	Damping ratio of tail pitch servo	0,85 [·]
ζ_{A}	Damping ratio of lateral pitch servo	0,85 [·]
ζ_{B}	Damping ratio of longitudinal pitch servo	0,85 [·]
K_{tr}	DC gain of tail pitch servo	0,71 [·]
K_{A}	DC gain of lateral pitch servo	0,43 [·]
K_{B}	DC gain of longitudinal pitch servo	0,35 [·]
$S_{lat,max}$	Input saturation of lateral pitch servo	1 rad
$S_{lat,min}$	Input saturation of lateral pitch servo	−1 rad
$S_{lon,max}$	Input saturation of longitudinal pitch servo	1 rad
$S_{lon,min}$	Input saturation of longitudinal pitch servo	−1 rad
$\dot{\psi}_{ref,max}$	Input saturation of tail pitch servo	1,5 rad/s
$\dot{\psi}_{ref,min}$	Input saturation of tail pitch servo	−1,5 rad/s

5.5.12 Main Motor Parameters

The main motor is modelled as a second order system, as well as the servos, though omitting the rate limit. The parameters for the main DC motor has basically been found in the same way as for the servo motors. The only difference is that it is not feasible to apply a step going from one extremity to the other, i.e. from no throttle to full throttle with the helicopter clamped to the ground. This would cause an extremely high torque from the DC motor, which could break the helicopter or parts of it. Instead, a smaller step size is chosen; through tests it has been found safe go from 60 rad/s to 85 rad/s. The time the rotor blades take from increasing the angular velocity correspondingly is found to approximatily 1,5 s. As there is no measurable overshoot, the damping ratio is 1 or larger. Increasing the damping ratio increases the settling time. By simulation it is estimated that a damping ratio of 1 is suitable. The DC gain is set to 1, because the software controlling the motor has been designed so that the input to the DC motor is angular velocity in rad/s, just as the output is. This leaves only the natural frequency to adjust, such that the model fits with the real motor. The result is shown in Figure 5.32.

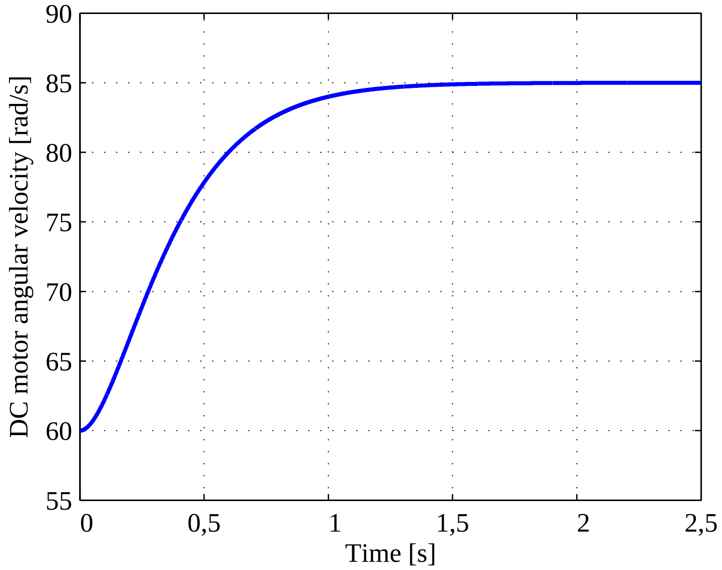


Figure 5.32: *A simulated step test for the DC motor. The input to the system is a step from* 60 rad/s *to* 85 rad/s.

The saturation parameters are only implemented as a safety precaution, in order to avoid dangerous situations with high angular velocities on the rotors.

$\omega_{n,mr}$	Undamped natural frequency of main rotor DC motor	5 rad/s
ζ_{mr}	Damping ratio of main rotor DC-motor	1 [·]
$\Omega_{mr,max}$	Input saturation of main DC motor	190 rad/s
$\Omega_{mr,min}$	Input saturation of main DC motor	0 rad/s
K_{mr}	DC gain of main DC motor	1 [·]

5.5.13 Drag Coefficients

To determine the drag coefficient of an object requires special test equipment, so this is not feasible in this project. Instead knowledge of different geometric shapes and their drag coefficients is used in combination with some 'common sense' considerations. Furthermore, the drag coefficients for the main and tail rotor blades has been found using experimental data as a basis for comparison. These two parameters are closely connected to the lift curve slopes for the rotor blades, and the procedure of finding them is presented in the next section.

C_d	Drag coefficient for main rotor blades	0,008 [·]
$C_{d,tr}$	Drag coefficient for tail rotor blades	0,008 [·]
d_{fp}	Drag coefficient for front plane	1,0 [·]
d_{tf}	Drag coefficient for tail fin	1,2 [·]
d_{tp}	Drag coefficient for tail plane	0,5 [·]
d_x	Longitudinal drag coefficient for body	1,0 [·]
d_y	Lateral drag coefficient for body	1,2 [·]
d_z	Vertical drag coefficient for body	1,0 [·]

5.5.14 Lift Curve Slopes and Drag Coefficients

The lift curve slope for the rotor blades is a scalar expression of the lift thrust as a function of the angle of attack (see Section 5.1.3 on page 45). Just as for the drag coefficients, it is not feasible (or necessary) to find these parameters with high precision. The procedure for determing these parameters are as follows:

1. Mount the helicopter on a weighing machine, such that the registered weight can be used to calculate the lift thrust.

2. Place the helicopter and weighing machine on a swiveling plate, and attach a force meter to the tail boom, such that the torque around the ${}^{B}z$-axis can be calculated. The setup is shown in Figure 5.33 on the facing page.

3. Conduct a series of experiments, where the lift thrust and yaw torque are logged as a function of main rotor angular velocity and tail rotor pitch.

4. Extract the same data (resulting lift thrust and resulting torque around ${}^{B}z$) from the implemented model, and make it take main rotor angular velocity and tail rotor pitch as input.

5. Find the lift curve slope for the main rotor blades, by fitting this parameter such that the model data is equal to the experimental data. It is assumed that the drag from the tail rotor can be omitted.

6. Find the lift curve slope for the tail rotor blades by fitting the change in yaw torque as a function of tail rotor pitch. The reason for using the difference instead of the absolute value of yaw torque, is that the absolute value will depend on the main rotor blade drag coefficient, which is not known yet.

7. Find the drag coefficient for the main rotor blades by fitting the absolute yaw torque values as function of main rotor angular velocity.

8. The drag coefficient for the tail rotor is assumed to be equal to the drag coefficient for the main rotor blades, as they have approximately the same shape.

9. The lift curve slope for the stabilizer bar is assumed to be somewhat smaller than the lift curve slope for the main rotor blades.

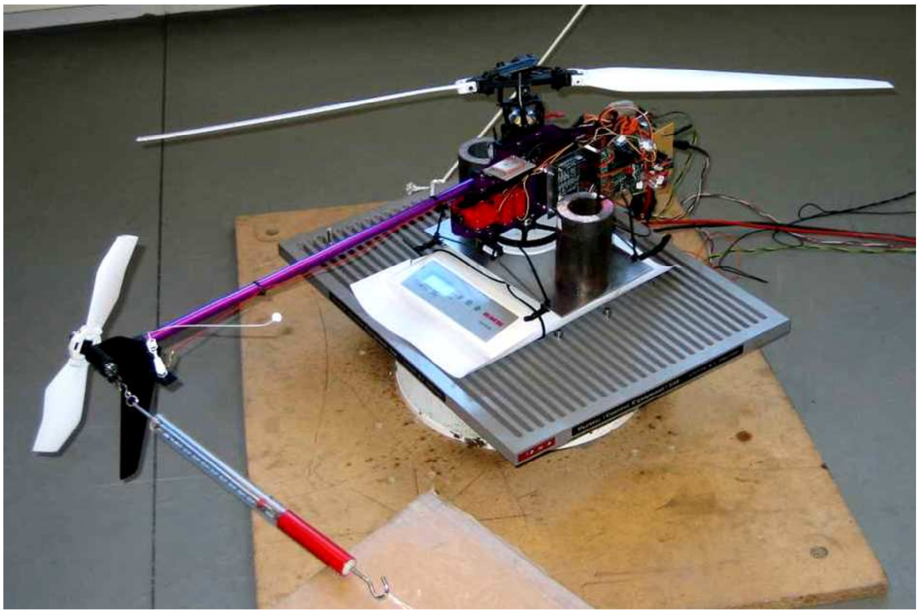


Figure 5.33: The setup to measure lift curve slopes and drag coefficients. The lift provided by the main rotor is measured by the weighing machine, and the thrust provided by the tail rotor is measured by the force meter. Coherent measurements of lift, thrust and rotation velocity are then taken to be able to derive the relevant parameters.

C_{ls}	Lift curve slope for main rotor blade	7,6 rad^{-1}
$C_{ls,sb}$	Lift curve slope for stabilizer bar	3 rad^{-1}
$C_{ls,tr}$	Lift curve slope for tail rotor blade	2,5 rad^{-1}

5.5.15 Collective Pitch and Twist

The helicopter does not have collective pitch for the main rotor as a variable control input, but the main rotor blades does have a fixed collective pitch in order to create lift thrust. This pitch angle depends on the distance from the hub, such that the pitch angle is largest close to the hub and decreases linearly towards the tip of the blade. Two parameters are used to express this phenomenon: The maximum collective pitch and the blade twist. Both parameters are measured by attaching two laser pointers on the blades parallel with the blade chord and pointing towards a wall 1 m away. By measuring the distance between the laser dots on the wall, it is possible to calculate the angle between the laser beams, and thereby between the blade chords at the position of the laser pointers.

The stabilizer bar has, in contrast to the one on the Bergen helicopter, collective pitch, which is fixed just as the main rotor blades, but has zero twist. This means that on the Corona helicopter, the stabilizer bar also provides thrust, but this is not accounted for in the model. This problem is handled by adding a small amount of collective pitch to the main rotor, so that in the model this rotor provides the additional lift which in reality is provided by the stabilizer bar.

As an example, the main rotor blade twist is given by

$$\theta_t = -\frac{\arctan(d)}{1 - e/R} \tag{5.96}$$

- θ_t is the main rotor blade twist.
- $d = 0{,}26\,\mathrm{m}$ is the distance between the laser dots.
- $1 - e/R$ is a factor accounting for that the measurement is not at the hub, but at a distance e from the hub.

Note that this equation requires one of the laser beams to point perpendicular towards the wall.
Regarding the tail rotor, it has collective pitch as a variable control input. The tail pitch angle is an affine function of the control input, described by

$$\theta_{tr} = K_{tr} S_{tr} + \theta_{tr,offset} \tag{5.97}$$

- θ_{tr} is the tail rotor pitch.
- K_{tr} is the DC gain of the tail pitch servo (see Section 5.5.11).
- S_{tr} is the input to the tail pitch servo.
- $\theta_{tr,offset}$ is the tail rotor pitch for $S_{tr} = 0$.

$\theta_{tr,offset}$ is measured the same way as θ_0 $\theta_{0,sb}$ and θ_t using laser pointers. See also Section 5.5.11 for relationship between input and output for the tail rotor.

θ_0	Main rotor collective pitch	0,35 rad
$\theta_{0,sb}$	Collective pitch of stabilizer bar	0,2 rad
θ_t	Main rotor blade twist	−0,29 rad
$\theta_{tr,offset}$	Collective pitch offset of tail rotor	0,28 rad

5.5.16 Spring Force Constant

The flapping spring force for the main rotor blade is measured by attaching a Newton-meter to the end of the blade, and pulling downwards. The data of this experiment is shown as dots in Figure 5.34, and it shows that the relationship between torque and flapping angle is approximately linear. The slope of the line is the spring force constant.

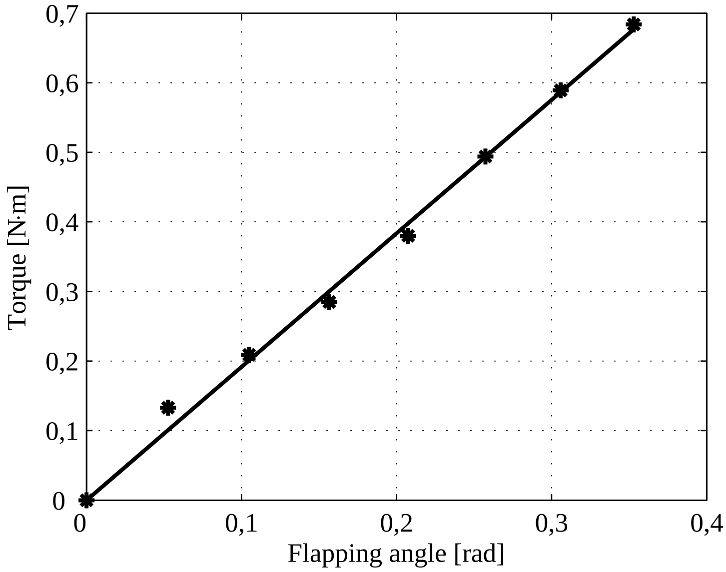


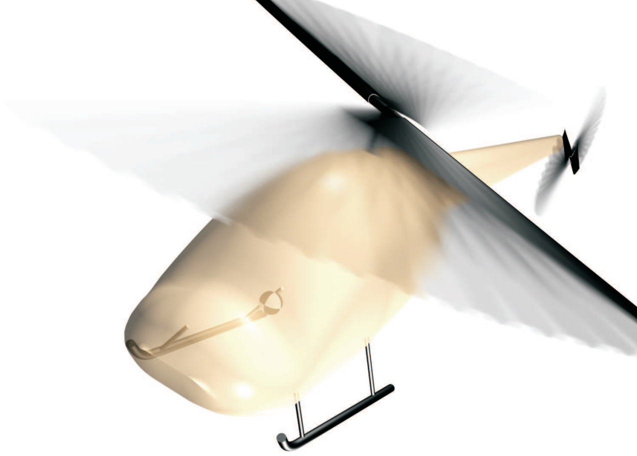
Figure 5.34: *Measurements of the main rotor blade's spring force.*

K_s	Spring force constant for main rotor blade	1,9 $^{\text{N·m}}/_{\text{rad}}$

All the parameters are calculated in the m-file
[CD-ROM, 2007, `./source/parameters/misc_parameters.m`].

5.6 Partial Conclusion

In this chapter, an overview over the existing first principles helicopter model has been given, and the modifications needed to adapt the model to the Corona 120 helicopter, has been developed. The adapted model has been verified similar to the procedure used in Hald et al. [2006], and the model works as expected when using the parameters for the Bergen helicopter. Last all the specific parameters for the Corona helicopter has been measured and found experimentally. It is assessed that the developed model with the estimated parameters, behaves as the real helicopter, and can therefore be used for controller development. It is possible to tune the parameters to make an even more accurate model. This can be done by using system identification methods on recorded flight data. The data can then be used together with the model, to update the parameters. For example, in La Civita [2003] frequency responce (to the pilot input) data is used to automatically correct the physical parameters in a first principles model, and a high bandwidth model is hereby obtained.



6

State Estimation

Chapter Contents

This chapter describes how an estimate of the current state is made from the sensor measurements. First, the approach for obtaining a state estimate using the camera setup and the IMU is presented. Then, the image processing software, which is measuring the absolute position and attitude of the helicopter is developed. This part is divided in two blocks, one block taking care of the marker localization, and another block for deriving the position and attitude. Last, an extended Kalman filter incoorporating sensor fusion and signal filtering is designed.

6.1 Overall Approach to Obtain a State Estimate

As the overall goal for this project is to stabilize the helicopter in hover, it is necessary to know at least the position and attitude of the helicopter in all three dimensions. Having this information, it will be possible to implement a position controller. However, it will be of great advantage in the control design to get information about the velocities (translatory and angular) as well. Because of the IMU mounted on the helicopter, this is certainly possible. As described in Chapter 4, we have two different sensors implemented: The IMU and the cameras.

The IMU

The IMU has six output signals: three translatory accelerations and three angular velocities. The three accelerations are denoted as

$$\ddot{\mathbf{\Xi}}_{IMU} = \begin{bmatrix} \ddot{x}_{IMU} \\ \ddot{y}_{IMU} \\ \ddot{z}_{IMU} \end{bmatrix} \quad , \tag{6.1}$$

and the three angular velocities as

$$\dot{\mathbf{\Theta}}_{IMU} = \begin{bmatrix} \dot{\phi}_{IMU} \\ \dot{\theta}_{IMU} \\ \dot{\psi}_{IMU} \end{bmatrix} \quad . \tag{6.2}$$

Because the helicopter is situated in the gravitational field of the earth, the IMU measures this as an upward acceleration of 9,82 $^{m}/_{s^2}$. This will have to be considered in the state estimation filter.

The Cameras

Each camera output is an RGB-format image of 640×480 pixels for the frontcam and sidecam and 320×240 for the helicam. That means that each pixel is described by three integers in the range $(0 - 255)$. Thus, one image is given by three 640×480 matrices, one containing the red color values, one containing the green colour values, and one containing the blue colour values. The frame rate of the cameras are 30 fps.

The strategy for obtaining a state estimate is illustrated in Figure 6.1.
The purpose of the cameras is to provide information about the position and attitude of the helicopter. To be able to identify this from the imagery, it is necessary to use some markers. Therefore the first block localizes these markers in the pictures. Hereafter the determined position of each marker from each camera can be used to calculate an individual position and attitude estimate of the helicopter. The position derivation block returns a position vector to the state estimation block denoted as

$$\mathbf{\Xi}_{cam} = \begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} \quad , \tag{6.3}$$

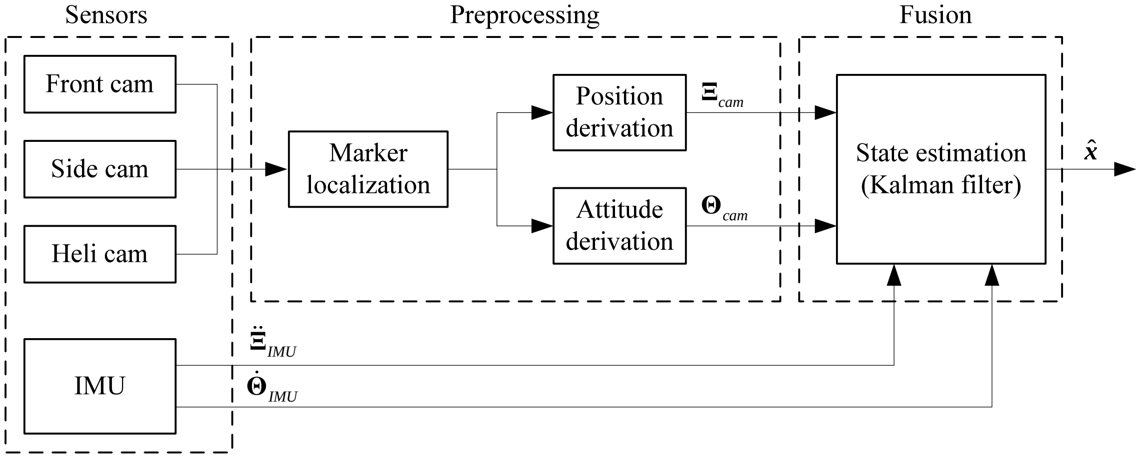


Figure 6.1: The overall structure of the state estimation of the helicopter.

and the output from the attitude derivation returns a vector denoted as

$$\boldsymbol{\Theta}_{cam} = \begin{bmatrix} \phi_{cam} \\ \theta_{cam} \\ \psi_{cam} \end{bmatrix} \quad . \tag{6.4}$$

The full measurement vector containing all the sensor output, is given by

$$\boldsymbol{z} = \begin{bmatrix} \ddot{\boldsymbol{\Xi}}_{IMU} \\ \dot{\boldsymbol{\Theta}}_{IMU} \\ \boldsymbol{\Xi}_{cam} \\ \boldsymbol{\Theta}_{cam} \end{bmatrix} = \begin{bmatrix} \ddot{x}_{IMU} \\ \ddot{y}_{IMU} \\ \ddot{z}_{IMU} \\ \dot{\phi}_{IMU} \\ \dot{\theta}_{IMU} \\ \dot{\psi}_{IMU} \\ x_{cam} \\ y_{cam} \\ z_{cam} \\ \phi_{cam} \\ \theta_{cam} \\ \psi_{cam} \end{bmatrix} \quad . \tag{6.5}$$

The idea is to put as much processing inside the state estimation block as possible. Here, a Kalman filter will take care of both sensor fusion and filtering of the sensor signals, and return the necessary position and velocity state estimates. The sensor signals contains redundancy, as the acceleration measurement of the IMU can be integrated two times to obtain a position estimate, which is measured by the cameras. Furthermore, by making one Kalman filter for state estimation, it is easy to add more sensors to the system, or changing the existing sensors. As mentioned in Chapter 2, the camera setup is in fact a substistute for a more generic position sensor, which works outside the lab. Eventually it will be necessary to change it to a GPS and maybe magnetometer sensor setup, and when this is done, one only have to change the measurement model in the Kalman filter to implement the new measurements.

6.2 Image Processing

Image processing concerns the derivation of the position and attitude of the helicopter from camera data. Since it is not the main focus of this project to analyse image data, it has been chosen to keep this block as simple as possible. This is done by using the two cameras placed on the wall to obtain a position estimate, and from this position estimate the helicam can be used to obtain an attitude estimate. By doing this, it is only necessary to know the orientation of the marker in the helicam image.

This section describes in detail how the image processing block is designed and implemented. First the design of markers on the helicopter is considered, and then how to localize them in an image. When this is done it is described how a position and attitude estimate of the helicopter is made from the marker positions in the image. Finally implementation and connection between the IPC and CC is considered.

6.2.1 Marker Design and Mounting

It has been chosen to place two markers on the helicopter. One marker is placed in the front of the helicopter in a direction towards the front camera, and one is placed on the side facing towards the side camera. The front marker is used to estimate the $y-z$ position of the helicopter, and the side marker is used to estimate the $x-z$ position. The redundancy of the z-axis estimate can be used to obtain a higher accuracy of the measurements.

It has been chosen to use a disc as marker, because it is rotation invariant in the plane of the disc. An elliptic deformation occurs when the disc is rotated in any other plane, but this deformation will be small due to the fact that the helicopter is stabilized in hover.

To be able to identify a marker clearly, the contrast between the marker and the surrounding background must be significant. To test which colours are the best, a paper sheet with white markers on different background colours (see Figure 6.2) has been filmed, both with and without a floodlight projector to illuminate the paper. The test, which is outlined in appendix D on page 187, showed that the white colour was very distinct, but none of the background colours were very clear because of the reflection of the paper. Therefore, it has been chosen to place a white paper marker on a non-glossy red background made of a cloth. The cloth and markers have been mounted on the helicopter as seen in Figure 6.3.

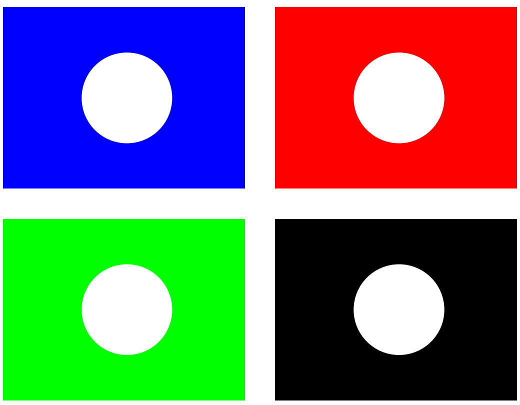


Figure 6.2: *The marker background colours used for testing the contrast.*

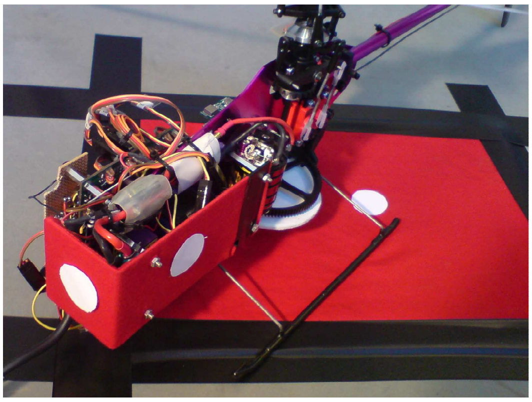


Figure 6.3: *The two markers mounted on the helicopter, and the marker placed on the floor.*

The size and the position of the markers are given in Table 6.2.1.

${}^{E}\boldsymbol{\kappa}_{mrk,hcam}$	Position of the marker on the floor given in $\{E\}$	$\begin{bmatrix} 0 \\ 0 \\ 1{,}02 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{mrk,fcam}$	Position of the front marker given in $\{B\}$	$\begin{bmatrix} 0{,}19 \\ -0{,}01 \\ 0{,}02 \end{bmatrix}$ m
${}^{B}\boldsymbol{\kappa}_{mrk,scam}$	Position of the side marker given in $\{B\}$	$\begin{bmatrix} 0{,}13 \\ -0{,}04 \\ 0{,}02 \end{bmatrix}$ m
r_{mrk}	The radius of a marker	0,015 m

6.2.2 Locating a Marker in an Image

To locate the marker in an image a Circular Hough Transform (CHT) is used.

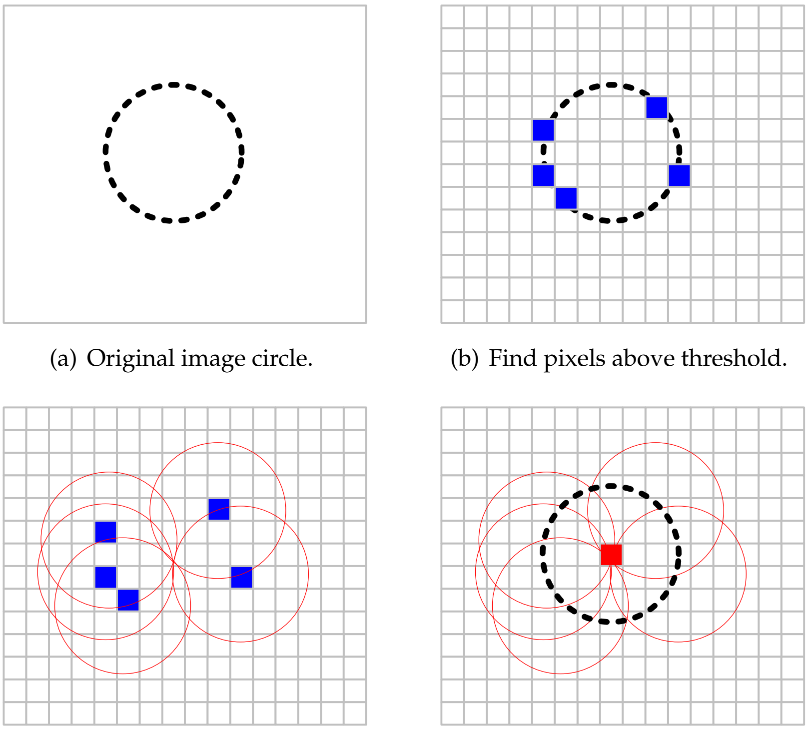
Hough Transform Theory

Originally the Hough transform was developed by Duda and Hart [1972] for finding parametrized lines in gray scale images, and further developed for edge finding in O'Gorman and Clowes [1973]. The basic idea of line detection can be outlined as follows. Given a pixel set (x_p, y_p) and a function $f(x_p, y_p)$ that describes the probability of the pixel being on the line,

1. describe each possible line in the image by a set of parameters e.g. slope and intersection ($y_p = ax_p + b$), or normal parametrization ($p = x_p \cos(\theta) + y_p \sin(\theta)$ [Duda and Hart, 1972]). The latter is the most used because it is easier to implement due to potential unboundedness of a and b.
2. Create an accumulator array for each parameter containing all the possible values of the parameter.
3. Investigate each pixel to see if it is a candidate to be on a line ($f(x_p, y_p) > T_p$, where T_p is some pixel threshold), and

4. if so, increment the accumulator arrays for all the values corresponding to lines going through the given pixel.

This procedure causes the accumulator representing a real line in the image to be incremented a lot of times, while other accumulators are only incremented a few times. Therefore it is only necessary to search for the maximum value of the accumulator array, to find the parameters for the most likely line in the image.
The Hough transform was later extended to more advanced shapes such as circles by Kimme et al. [1975] (CHT), and generalized to arbitrary shapes by Ballard [1981]. The parameters of circles are center and radius and the CHT can be used to find circles with arbitrary centers and radii. However, the computational complexity of the Hough transform increases polynomially with the number of parameters, so computation time can reduced by searching for circles with a specific radius. In Figure 6.4 the application of the CHT with a fixed radius is illustrated.



(a) Original image circle.

(b) Find pixels above threshold.

(c) Increment accumulator.

(d) Find center.

Figure 6.4: *Illustration of the CHT with fixed radius applied to an image. (a) The original image of a circle. (b) The discretization in 16 by 14 pixels with the pixels above the threshold filled. To reduce complexity of the image only five pixels which is above the threshold are emphasized, and the original circle is superimposed. (c) The center will be at a distance equal to the original circle radius from each pixel above the threshold. The accumulator array is incremented at all these points. (d) The highest value of the accumulator array will be where the circles intersect, which is equal to the correct center of the circle.*

In Kimme et al. [1975] it is further suggested to use the direction of the gradient of the image to only update the accumulator where it is likely for the circle center to be. This can increase the computational speed of the algorithm.

Application of the Circular Hough Transform

Inner Circle Search As the markers on the helicopter are discs, the CHT can be used on these if searching for a circle in the image with radius corresponding to the size of the marker. Assuming that the algorithm works as expected, a circle with the same radius as the marker will only be present exactly on top of the marker. If a smaller radius is used in the CHT there will be several possibilities for the estimated center. However, the center can only be as far away as the difference between the radius of the marker and the used radius in the algorithm. If on the other hand a larger radius is used there will be no circles found, and therefore the radius must be chosen carefully to not exceed the radius of the marker. Further some blurry effect will arise in the transition between the marker and background since the image quality of the used web cameras are not perfect. These facts leads to the conclusion that to find the marker in the image, it is necessary to search for a circle slightly smaller than the marker.

Outer Circle Search To ensure that it is the actual marker which is found, the background is analyzed as well. For the same reasons as for the inner circle search it will this time be preferable to search for a circle slightly larger than the marker radius.

Pixel threshold and accumulator incrementation To determine if a pixel is a candidate to be on the circle, a "distance" threshold T_p is used. The pixel "distance" D_p is calculated as the deviation from the expected colour:

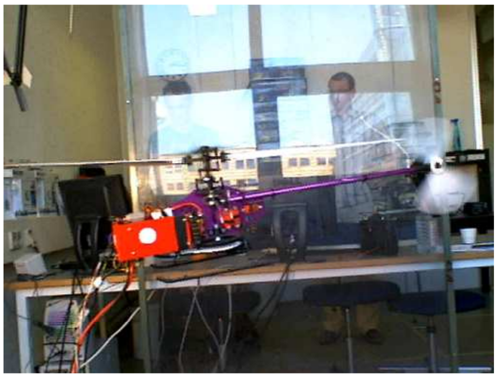
$$D_p = |R_p - R_e| + |G_p - G_e| + |B_p - B_e| \tag{6.6}$$

- D_p is the pixel colour distance.
- R_p, G_p and B_p are the red, green and blue values of a pixel, respectively.
- R_e, G_e and B_e are the expected red, green and blue values of a pixel, respectively.

If the distance is less than the preset threshold T_p the pixel is a candidate for being on the circle, and the accumulator array is incremented as shown in Figure 6.4. Instead of incrementing the accumulator array with 1, it is incremented by a value of $255 - D_p$. This makes pixels which are very close to the expected colour (D_p is very low) more significant.

Verification of the Algorithm The CHT has been implemented and tested on two images recorded by the frontcam and sidecam when flying manually in the lab. The images obtained are shown in Figure 6.5 on the following page.

(a) Front camera.



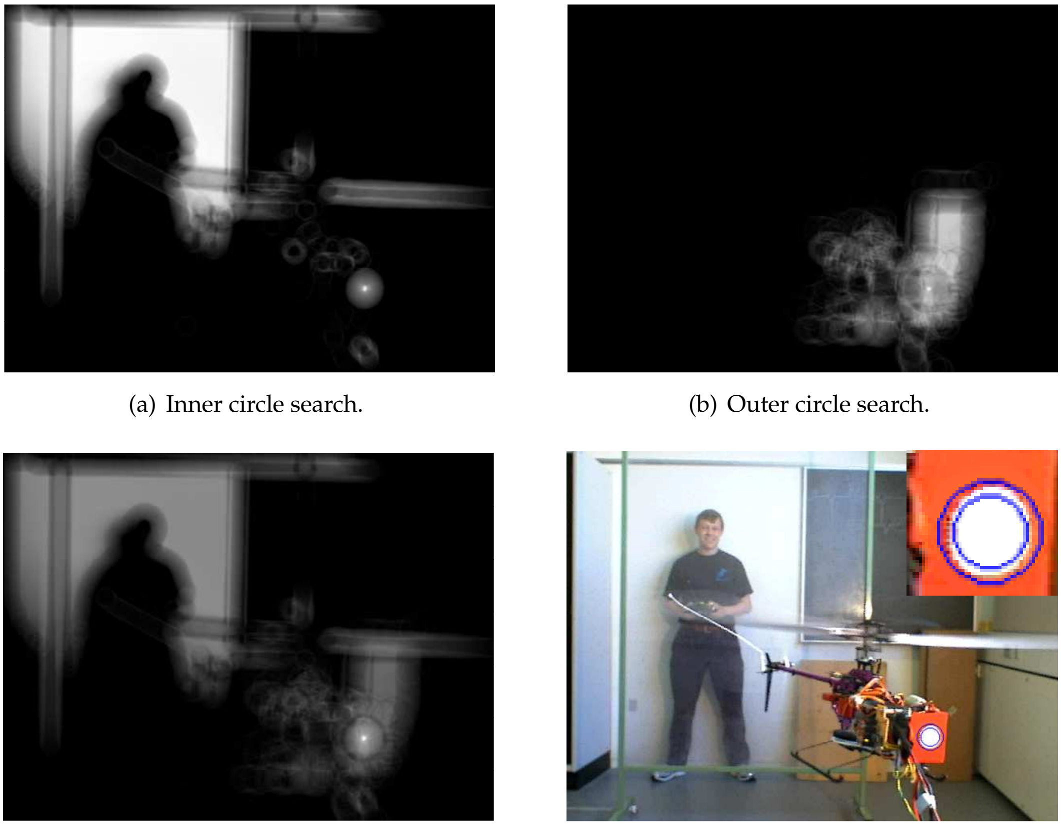
(b) Side camera.

Figure 6.5: *The helicopter in a manual test flight in the lab.*

The threshold and RGB-values has been chosen empirically from the results when processing different images. The threshold is chosen rather large in order to account for changing light conditions in the lab. Regarding the circle radius it is in this case set by testing different radii to see which fits the best. As described, it can be difficult to choose the right parameters, but the ones used for this test is tested on different images with different lighting conditions and is believed to be appropriate for the final system. The parameters used for the front camera CHT are as follows:

- Marker colour: (245, 245, 245)
- Inner circle threshold: 150
- Inner circle radius: 12 pixels
- Background colour: (210, 50, 50)
- Outer circle threshold: 150
- Outer circle radius: 17 pixels

When the image has been processed the accumulator array is normalized to 255, i.e. all values are scaled by $\frac{255}{max(accumulator)}$. This means that the array now represents a gray scale image with the most likely pixel to be the center (highest value of the accumulator array) white, and the lower value in the array, the more dark the pixels will be. The results of the processing of the image from the frontcam are illustrated in Figure 6.6 on the next page.



(a) Inner circle search.

(b) Outer circle search.

(c) Combined inner and outer circle search.

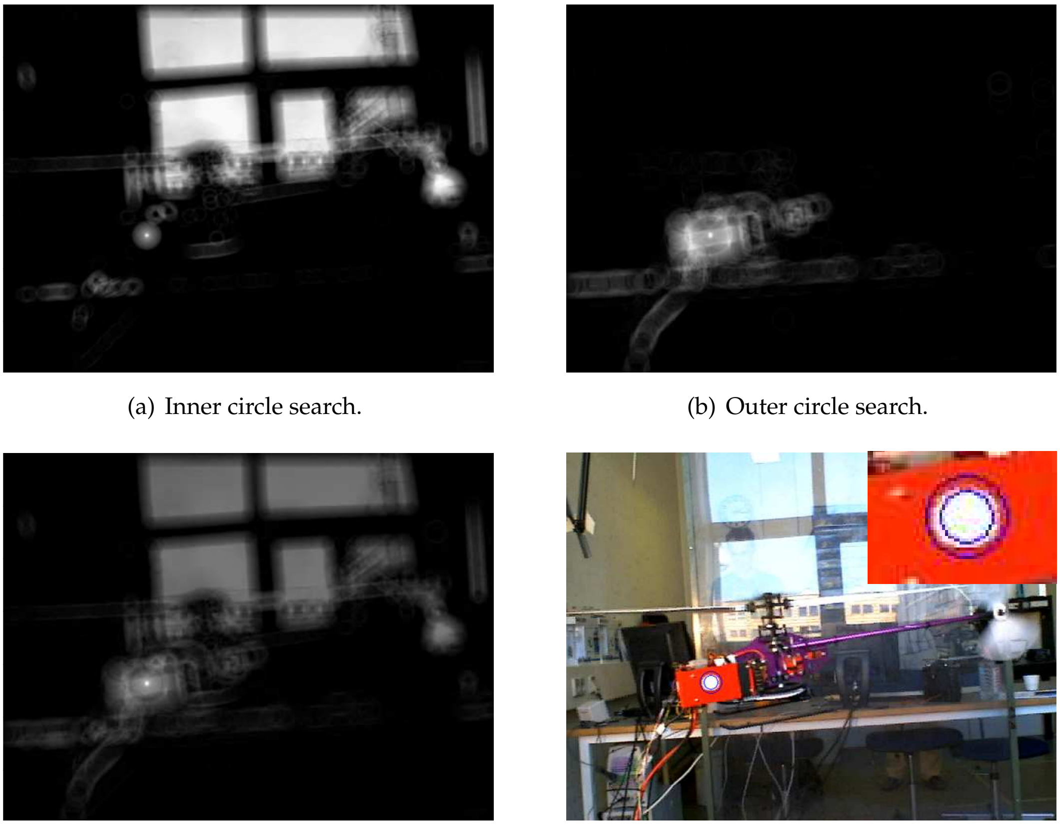
(d) Marker localized.

Figure 6.6: *The output of the accumulator array of the CHT shown for inner circle search, outer circle search and combined search of the image in Figure 6.5(a) shown in (a), (b) and (c). The resulting estimated marker position is drawn on the image in (d).*

In Figure 6.6(a) the accumulator array for the inside circle search is seen. It can be seen that the marker can be identified, but as some of the background is white, the output might be erroneous when only doing search for the inner circle. In Figure 6.6(b) the result for the outside circle search is shown. Here the marker is identified as well, but at some region around the marker there is also high accumulator values. In Figure 6.6(c) the inner and outer circle search is combined by adding the accumulator values for both searches. It is seen that the marker center emerges more clearly. In Figure 6.6(d) the inner and outer circles has been drawn where the center was found, and it is seen that the center is located correctly.
The algorithm is also verified for the side camera with the following parameters:

- Marker colour: (245, 245, 245)
- Inner circle threshold: 150
- Inner circle radius: 9 pixels
- Background colour: (210, 50, 50)
- Outer circle threshold: 150
- Outer circle radius: 14 pixels

In this case the marker is also localized and the results can be seen in Figure 6.7.



(a) Inner circle search.

(b) Outer circle search.

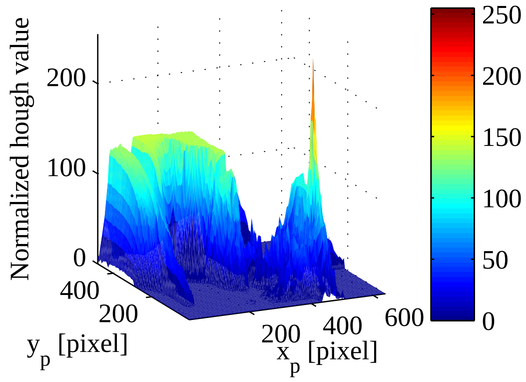
(c) Combined inner and outer circle search.

(d) Marker localized.

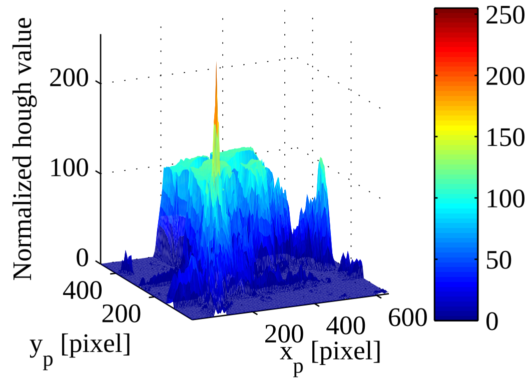
Figure 6.7: *The output of the accumulator array of the CHT shown for inner circle search, outer circle search and combined search of the image in Figure 6.5(b) shown in (a), (b) and (c). The resulting estimated marker position is drawn on the image in (d).*

On black and white output images from hough accumulator it can be difficult to see how certain it is that the right marker position is found. To emphasize that the algorithm is robust, a 3D plot of each of the accumulator arrays in Figures 6.6(c) and 6.7(c), is plotted in Figure 6.8. A very distinct peak is seen on both frontcam image and the sidecam image. Further, the mean value is only 29 and 22 for the front and side accumulator respectively, which gives a peak-to-mean ratio of at least 8,5.

Optimization Some optimization to make the algorithm even more robust to different changes in for example light conditions has been considered. When both inner and outer circle search has been applied, it has been tried to rule out all centers where not both the inner and outer search gives a high accumulator value. This is done by checking all accumulator values and if either the array for inner or the array for outer search is below some threshold, then the combined accumulator is set to 0. It has been seen that this improves the uniqueness of the circle center in the combined accumulator array. However, in the existing implementation of the algorithm this procedure requires a lot of



(a) 3D plot of the output of the hough accumulator array from Figure 6.6(c).



(b) 3D plot of the output of the hough accumulator array from Figure 6.7(c).

Figure 6.8: *On the two figures a very distinct peak is seen, and the conclusion is therefore that the algorithm for finding the marker is very reliable.*

extra processing time, and it is assessed that gain of the method is lower than this cost.
Because the lighting conditions might change from one flight to another, then the background colour might change as well. To compensate for this, a function, that measures the background colour just beside where the marker is located and uses this as background colour next time, has been implemented. The disadvantage of this method is that if one image yields the wrong center, the background colour will be measured wrong as well, and therefore the possibility of finding the right center in the next image will be decreased. The possiblility of this scenario is decreased by using a default background colour if the colour measured is too far away from the expected background colour (e.g. if a blue colour is measured instead of the expected red background colour).
It is seen in Figures 6.6(a) and 6.7(a) that a great deal of the background is white as well, and this might cause the marker to be estimated at the wrong position. If using the gradient in the picture as described in O'Gorman and Clowes [1973], only the edge between the marker and the background will appear clearly as a marker. This has been tried but does not seem to work very well. Another disadvantage by using the gradient is that it is necessary to know exactly the size of the circle to search for.

Increasing computational speed

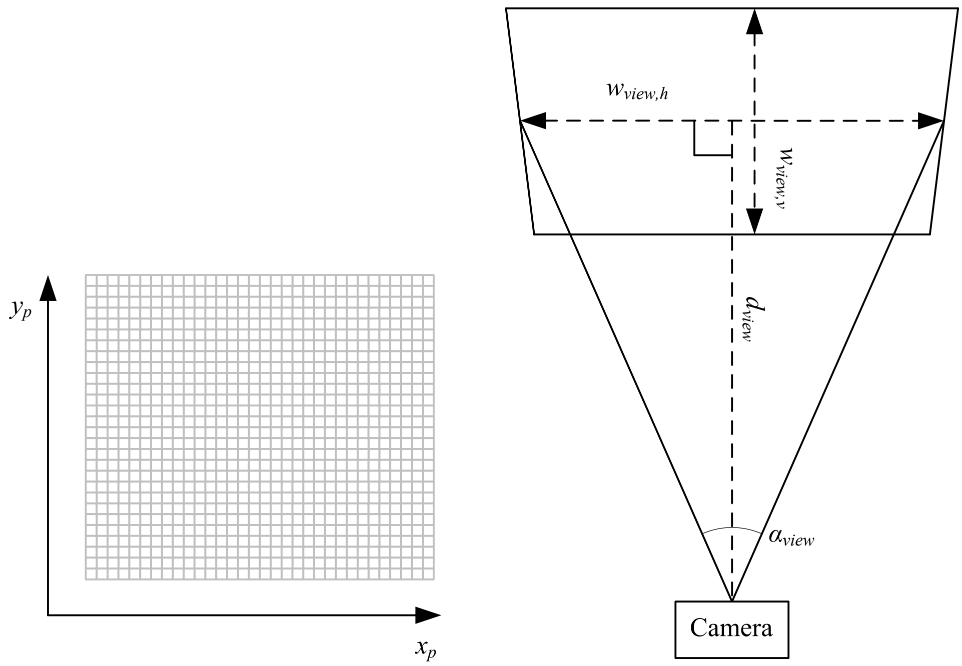
In this application the image size is 480×640, which means that 307840 pixels are examined for each image. But since the helicopter has a limited velocity compared to the frame rate of the cameras, it is not moving very far away from the measured position during one frame. Thus, by examining a "region of interest" (ROI) in the image where the marker was last time, the number of pixels can be reduced considerably. If examining an area of for example 100×100 pixels, then only 10 000 pixels are examined which is less than 1/30 of the original number.

Reprocessing and Error Handling

A problem with using a ROI is that, if the ROI is chosen too small or the helicopter velocity is too large, the marker might be out of the ROI before the next frame, and a wrong estimate of the marker center might be obtained. However, if this case can be discovered, it is possible to do an reprocessing of the whole picture to find the right position of the marker. This can be done in several ways. If the maximum value of the accumulator array is lower than usual then the marker position estimate might be wrong. Anther possibility is to count the number of pixels found below the threshold, and if this number is not large enough, then the estimate might be erroneous. Tests of both methods have been done with both yielding good results. But both methods have a threshold which have to be tuned specifically to other parameters used in the CHT. For example, the threshold of low value of the accumulator array is dependent on the size of the marker, which changes dynamically and the number of pixels found below the threshold is dependent on the size of the ROI. Therefore a more generic method is used. The maximum number of times, the accumulator array can be incremented, is equal to the number of pixels at a distance equal to the radius of the circle searched for. The maximum value of the accumulator array is therefore normalized according to the total number of pixels in the inner and outer circle. Since, at each incrementation, it is only possible to add 255 to the accumulator array, the maximum value is 255. It has been tested empirically that if the marker is located correct, the value ranges around 180 to 230, dependent on the operating conditions and which camera is used. Based on these values, it has been chosen to use a threshold of 170 to determine if a reprocessing of the whole is image is necessary.
Another possibility, is that the marker is totally outside the image. In this case it is not possible to locate where the marker is at all. But, if the algorithm uncritically chooses the place in the image, which looks most like a marker, it might cause a sudden jump and a total erroneous position estimate, which again might lead to large wrong control input to the helicopter. This is a scenario which might lead the helicopter to suddenly crash. To avoid this, it has been chosen to reuse the marker position estimate from the previous image, if the threshold is still below 170, when an reprocessing of the whole image has been conducted.
This error handling discussion concludes the processing of each image individually. Now the use of the marker position estimates are considered.

6.2.3 Camera Properties

To be able to use the pixel position of the image, it is needed to know the properties of the cameras. The pixel co-ordinates are defined as shown in Figure 6.9(a) on the facing page, with the lower left pixel denoted (1, 1).
It is assumed that each pixel correspond to a specific angle of view ($\alpha_{pix,wall}$ and $\alpha_{pix,hcam}$ respectively for the two types of cameras) no matter if it is in the edge of an image or at the center. Further it is assumed that horizontal and vertical angle spanned by each pixel is the same. This angle can be found by measuring the whole opening angle α_{view} as shown in Figure 6.9(b) and dividing by the number of pixels (res_v and res_h for vertical and horizontal resolution respectively).



(a) Definition of the pixel co-ordinates in an image.

(b) The opening angle of the camera and the angle corresponding to each pixel can be determined by measuring the parameters given in the figure. w_{view} is measured by holding a ruler in front of the camera and inspecting how much of it it is possible to see.

Figure 6.9: *Determination of camera properties*

The camera and measured parameters are:

- $d_{view} = 1\,\text{m}$
- $w_{view,fcam,h} = w_{view,scam,h} = 0{,}82\,\text{m}$
- $w_{view,fcam,v} = w_{view,scam,v} = 0{,}60\,\text{m}$
- $res_{fcam,h} = res_{scam,h} = 640$
- $res_{fcam,v} = res_{scam,v} = 480$
- $w_{view,hcam,h} = 0{,}76\,\text{m}$
- $w_{view,hcam,v} = 0{,}57\,\text{m}$
- $res_{hcam,h} = 320$
- $res_{hcam,v} = 240$

The opening angle α_{view} is calculated as the double of the right-angled triangle spanned by d_{view} and $w_{view}/2$:

$$\alpha_{view} = 2 \cdot \arctan\left(\frac{w_{view}}{2 \cdot d_{view}}\right) \quad . \tag{6.7}$$

The angle spanned by a pixel is calculated by dividing by the resolution. The horizontal and vertical angle are slightly different due to measurement errors, and therefore a mean between the horizontal and vertical angle is used:

$$\alpha_{pix} = \left(\frac{\alpha_{view,h}}{res_h} + \frac{\alpha_{view,v}}{res_v}\right)/2 \quad . \tag{6.8}$$

This yields the results:

- $\alpha_{pix,fcam} = \alpha_{pix,scam} = 1{,}22 \cdot 10^{-3}$ rad
- $\alpha_{pix,hcam} = 2{,}31 \cdot 10^{-3}$ rad

6.2.4 Determination of the Position of CM

When the markers in each image from the wallcams are found, it is possible to find the direction of the vector from the camera towards the marker. By adding the position of the camera in $\{E\}$ and subtracting the position of the marker relative to the CM of the helicopter, a line passing through the CM is obtained ($\boldsymbol{l}_{fcam}$ and $\boldsymbol{l}_{scam}$), and this CM will be located exactly where the lines intersect. This is illustrated in Figure 6.10.

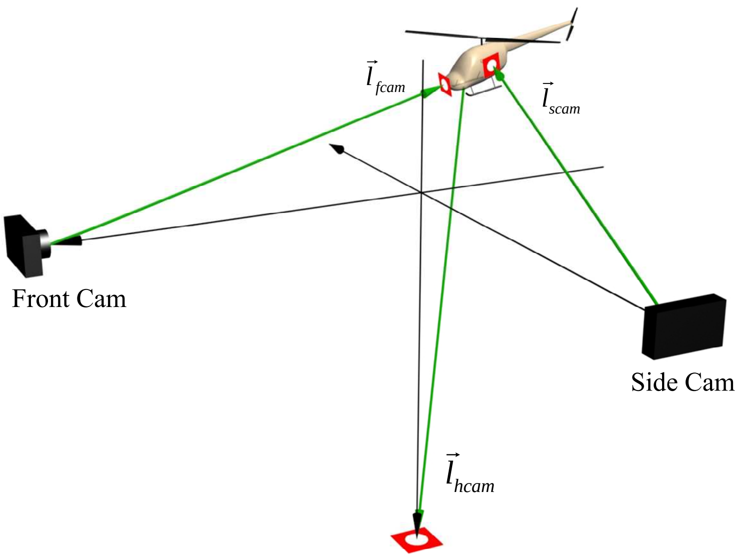


Figure 6.10: *A line from each camera towards the marker is created. The position of the helicopter can be calculated finding the intersection between the lines if the marker offset from the CM is taken into account.*

Finding the Unit Vector Towards the Marker

To find the unit vector towards the center of the marker ${}^E\boldsymbol{v}_{mrk}$, two angles are used. θ_{mrk} is the angle between an offset position and the center of the marker (x_p,y_p). The offset position (x_{off},y_{off}) is the location of the marker center when the helicopter is situated in zero position. The offset position does not need to be exactly in the center of the image, if the initial orientation of a camera is not exactly aligned with $\{E\}$. The other angle ϕ_{mrk} is the angle between horizontal and the center of the marker seen from the offset position. The parameters are shown in Figure 6.11.

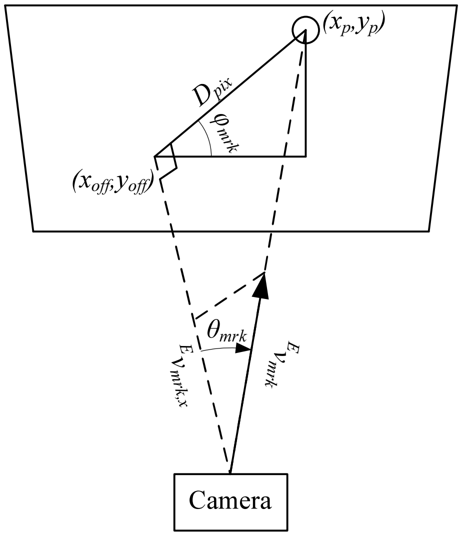


Figure 6.11: *Illustration of the parameters used to determine the unit vector from the camera towards the marker center (x_p,y_p).*

θ_{mrk} is found by calculating the distance between the marker and the offset in pixels D_{pix} and multiplying by the angle spanned by one pixel. ϕ_{mrk} is found by using the tangent rule:

$$\theta_{mrk} = \alpha_{pix} \sqrt{(x_p - x_{off})^2 + (y_p - y_{off})^2} \quad , \tag{6.9}$$

$$\phi_{mrk} = \arctan\left(\frac{y_p - y_{off}}{x_p - x_{off}}\right) \quad . \tag{6.10}$$

Note that it is necessary to use a four quadrant tangent function (such as `atan2()` in MATLAB), which is able to distinguish between the signs of x and y:
`if` $(x \geq 0) \Rightarrow \arctan\left(\frac{y}{x}\right)$
`else if` $(x < 0) \Rightarrow sign(y) \cdot \pi + \arctan\left(\frac{y}{x}\right)$.

Now it is possible to construct the vector pointing towards the marker. This must be done individually for the two cameras, since the image x- and y-axis does not correspond to the earth frame orientation.
Assuming ($x_{off,fcam}$,$y_{off,fcam}$) is in the direction of the origin of the earth frame and the camera is in level with the earth frame (${}^E\kappa_{fcam,z} = 0$), then it can be seen from Figure 6.11 that the length of ${}^E\boldsymbol{v}_{mrk,fcam,x}$ is $\cos(\theta_{mrk,fcam})$. Notice that the distance to the marker does not enter the equation since ${}^E\boldsymbol{v}_{mrk}$ is scaled to unit size. Further it can be seen from Figure 6.10 that the direction is opposite of the x axis of $\{E\}$. The length of ${}^E\boldsymbol{v}_{mrk,fcam,y}$ is equal to the length of $\sin(\theta_{mrk,fcam})\cos(\phi_{mrk,fcam})$, and finally the length of ${}^E\boldsymbol{v}_{mrk,fcam,z}$ is $\sin(\theta_{mrk,fcam})\sin(\phi_{mrk,fcam})$. Both the z and y direction has opposite direction as well as the x direction.

Similar considerations can be done for the vector pointing towards the side marker from the side camera. The following result is then obtained:

$$
{}^{E}\boldsymbol{v}_{mrk,fcam,tmp} = \begin{bmatrix} -\cos(\theta_{mrk,fcam}) \\ -\sin(\theta_{mrk,fcam})\cos(\phi_{mrk,fcam}) \\ -\sin(\theta_{mrk,fcam})\sin(\phi_{mrk,fcam}) \end{bmatrix} \quad , \tag{6.11}
$$

$$
{}^{E}\boldsymbol{v}_{mrk,scam,tmp} = \begin{bmatrix} -\sin(\theta_{mrk,scam})\cos(\phi_{mrk,scam}) \\ \cos(\theta_{mrk,scam}) \\ -\sin(\theta_{mrk,scam})\sin(\phi_{mrk,scam}) \end{bmatrix} \quad . \tag{6.12}
$$

If the camera is not in level with the earth frame the vectors must be rotated to take the angle towards the origin of $\{E\}$ into account. This is done by rotating ${}^{E}\boldsymbol{v}_{mrk,fcam,tmp}$ around the y-axis and ${}^{E}\boldsymbol{v}_{mrk,scam,tmp}$ around the x-axis according to

$$
{}^{E}\boldsymbol{v}_{mrk,fcam} = \begin{bmatrix} \cos(\xi_{fcam}) & 0 & \sin(\xi_{fcam}) \\ 0 & 1 & 0 \\ -\sin(\xi_{fcam}) & 0 & \cos(\xi_{fcam}) \end{bmatrix} \cdot {}^{E}\boldsymbol{v}_{mrk,fcam,tmp} \quad , \tag{6.13}
$$

$$
{}^{E}\boldsymbol{v}_{mrk,scam} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\xi_{scam}) & -\sin(\xi_{scam}) \\ 0 & \sin(\xi_{scam}) & \cos(\xi_{scam}) \end{bmatrix} \cdot {}^{E}\boldsymbol{v}_{mrk,scam,tmp} \quad . \tag{6.14}
$$

- ξ_{fcam} is the angle between horizontal and the origin of $\{E\}$ seen from the front camera: $\xi_{fcam} = \arctan\left(\frac{-{}^{E}\kappa_{fcam,z}}{{}^{E}\kappa_{fcam,x}}\right)$.
- ξ_{scam} is the angle between horizontal and the origin of $\{E\}$ seen from the side camera: $\xi_{scam} = \arctan\left(\frac{-{}^{E}\kappa_{scam,z}}{{}^{E}\kappa_{scam,y}}\right)$.

To create a parametric vector equation for the line going through the camera instead of going through the origin of $\{E\}$, the position of the camera is added:

$$
\boldsymbol{l}_{fcam}(p_{fcam}) = {}^{E}\boldsymbol{v}_{mrk,fcam} \cdot p_{fcam} + {}^{E}\boldsymbol{\kappa}_{fcam} \quad , \tag{6.15}
$$

$$
\boldsymbol{l}_{scam}(p_{scam}) = {}^{E}\boldsymbol{v}_{mrk,scam} \cdot p_{scam} + {}^{E}\boldsymbol{\kappa}_{scam} \quad . \tag{6.16}
$$

- $\boldsymbol{l}_{fcam/scam}$ line passing through the CM of the helicopter. Obtained by the front camera and side camera respectively.
- $p_{fcam/scam} \in \mathcal{R}$ describes each point of the line.
- ${}^{E}\boldsymbol{\kappa}_{fcam}$ is the position of the front camera in $\{E\}$.
- ${}^{E}\boldsymbol{\kappa}_{scam}$ is the position of the side camera in $\{E\}$.

These lines passes through the markers, but it is wanted to describe lines going through the CM of the helicopter. Therefore an offset corresponding to the distance from the CM to the marker is subtracted:

$$
\boldsymbol{l}_{fcam}(p_{fcam}) = {}^{E}\boldsymbol{v}_{mrk,fcam} \cdot p_{fcam} + {}^{E}\boldsymbol{\kappa}_{fcam} - {}^{E}\boldsymbol{p}_{mrk,fcam} \tag{6.17}
$$

$$
\boldsymbol{l}_{scam}(p_{scam}) = {}^{E}\boldsymbol{v}_{mrk,scam} \cdot p_{scam} + {}^{E}\boldsymbol{\kappa}_{scam} - {}^{E}\boldsymbol{p}_{mrk,scam} \tag{6.18}
$$

- ${}^{E}\boldsymbol{p}_{mrk,fcam/scam}$ is the vector from the CM of the helicopter to the marker. Notice that ${}^{E}\boldsymbol{p}_{mrk,fcam/scam} = {}^{B}\boldsymbol{\kappa}_{mrk,fcam/scam}$ if the attitude of the helicopter is $\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$

Now it is wanted to find the intersection between the lines, i.e. where $\boldsymbol{l}_{fcam} = \boldsymbol{l}_{scam}$. In the real application this is hardly ever fulfilled, but the lines should be very close to each other. To find where the lines are closest to each other a least squares estimate of the distance between the two lines are used, i.e. $min(||\boldsymbol{l}_{fcam} - \boldsymbol{l}_{scam}||_2)$. The least squares estimate can be found by first rewriting the equation:

$$
\begin{aligned}
& \boldsymbol{l}_{fcam} - \boldsymbol{l}_{scam} = 0 \\
\Updownarrow \\
& {}^{E}\boldsymbol{v}_{mrk,fcam} \cdot p_{fcam} + {}^{E}\boldsymbol{\kappa}_{fcam} - {}^{E}\boldsymbol{p}_{mrk,fcam} - \left({}^{E}\boldsymbol{v}_{mrk,scam} \cdot p_{scam} + {}^{E}\boldsymbol{\kappa}_{scam} - {}^{E}\boldsymbol{p}_{mrk,scam}\right) = 0 \\
\Updownarrow \\
& \begin{bmatrix} {}^{E}\boldsymbol{v}_{mrk,fcam} & -{}^{E}\boldsymbol{v}_{mrk,scam} \end{bmatrix} \begin{bmatrix} p_{fcam} \\ p_{scam} \end{bmatrix} = \begin{bmatrix} {}^{E}\boldsymbol{\kappa}_{scam} - {}^{E}\boldsymbol{\kappa}_{fcam} + {}^{E}\boldsymbol{p}_{mrk,fcam} - {}^{E}\boldsymbol{p}_{mrk,scam} \end{bmatrix} \\
\Updownarrow \\
& \mathcal{A}\boldsymbol{x} = \boldsymbol{b}
\end{aligned}
\tag{6.19}
$$

By using the pseudo inverse of $\mathcal{A}$ ($\mathcal{A}^{+}$), the solution to the least squares problem is given by

$$
\boldsymbol{x} = \begin{bmatrix} p_{fcam} \\ p_{scam} \end{bmatrix} = \mathcal{A}^{+} \cdot \boldsymbol{b} = (\mathcal{A}^T\mathcal{A})^{-1}\mathcal{A}^T \cdot \boldsymbol{b} \tag{6.20}
$$

By insertion into (6.15) and (6.16) the CM can be found by calculating the middle point between the lines:

$$
\Xi_{cam} = \frac{\boldsymbol{l}_{fcam}(p_{fcam}) + \boldsymbol{l}_{scam}(p_{scam})}{2} \tag{6.21}
$$

The distance between the lines ($d_{l,fcam,scam} = ||\boldsymbol{l}_{fcam}(p_{fcam}) - \boldsymbol{l}_{fcam}(p_{fcam})||_2$) serves as an estimate of how accurate the calculation of the CM position is. If the distance is large it is an indication that there might be something wrong, and it is possible to disregard the sample. This estimate is also used to feed back to the marker algorithm and force an update of the whole image, instead of only the ROI, if the error is too large.

Marker Size When the position of the helicopter has been found, it is possible to calculate the distance to the markers. This can be used to derive the radii of the markers in the images and use these in the next iteration when finding the markers using the CHT. The distance to the marker is

$$
d_{mrk} = ||\Xi_{cam} - {}^{E}\boldsymbol{\kappa}_{cam} + {}^{E}\boldsymbol{p}_{mrk}||_2 \tag{6.22}
$$

The angle extend of the marker is calculated by the tangent rule. Finally the angle is converted to pixels by dividing by the angle per pixel α_{pix} and rounding of to an integer.

$$
r_{pix,mrk} = round\left(\frac{\arctan\left(\frac{r_{mrk}}{d_{mrk}}\right)}{\alpha_{pix}}\right) \tag{6.23}
$$

As written in Section 6.2.2 on page 91 it is necessary to search for a marker which is a bit smaller than the actual size of the marker. Experiments has shown that the following values are usable

- Inner circle radius = $0{,}9 \cdot r_{pix,mrk}$
- Outer circle radius = $1{,}5 \cdot r_{pix,mrk}$

Region Of Operation The Region of operation (ROO) is defined as the set of all possible locations of the helicopter, where the respective markers are within the field of view for both the front camera and side camera. To determine the ROO, advanced geometric analysis of the field of view for each camera i necessary. This would result in a mathematical formulation of the region, but not create a better visual overview of the region. A visual overview would be good for the pilot who has to maintain the helicopter within the region until the control is switched to autonomous. Instead it has been chosen to make an algorithm, to find out if the helicopter is in the ROO, for a given position of the CM. From this a 3D figure which shows the ROO is created.
The ROO is three dimensional convex (all inner angles are below 180°) polytope with eight vertex points (like a deformed cube). A polytope can be described by its faces (or more general, facets if the polytope is not in three dimensions), which bounds the polytope. For example a cube is bounded by its six faces. The interior set of points in a polytope can be described by n linear inequalities [Weisstein, 2007],

$$\mathcal{A}_{n\times d}\,\boldsymbol{x} \;\geq\; \boldsymbol{b} \tag{6.24}$$

- $\mathcal{A}$ is a real matrix
- n is the number of faces
- d is the dimension of the polytope
- $\boldsymbol{x}$ is a point in the space
- $\boldsymbol{b}$ is a vector.

Exemplified by a dice located in $\{E\}$, and showing the number 6. z must be less than the plane in which the face with six dots lie, and larger than the plane in which the face with one dot lie, e.g. $z \leq 1$ and $z \geq 0$.
The ROO is characterized by the four bounding planes of each camera (which can be seen in Figure 4.3 on page 26). A plane can be described by a normal vector to the plane $\boldsymbol{n}$, and a point on the plane, where all points in the plane satisfies the equation

$$\boldsymbol{n} \bullet (\boldsymbol{r} - \boldsymbol{p}) = 0, \tag{6.25}$$

- $\boldsymbol{n}$ is the normal vector to the plane
- $\boldsymbol{r}$ is a point in the space
- $\boldsymbol{p}$ is a point on the plane
- $\bullet$ is the vector dot product

If $\boldsymbol{n} \bullet (\boldsymbol{r} - \boldsymbol{p}) \geq 0$, then the angle between the normal vector and $\boldsymbol{r}$ is less than $\pi/2$ rad. This can be used to create the inequalities in Equation (6.24). If a normal vector pointing into the view of the camera is found, then the angle between this and a point should be less than $\pi/2$ rad. A normal vector to a plane can be found by the cross product of two vectors in the plane. Since it is known that the direction vector towards each of the corner pixels lies in two planes, e.g. the pixel (0, 0) lies in both the left and bottom plane,it is possible to find the normal vectors of each plane. A direction vector ${}^{E}\boldsymbol{v}_{x_p,y_p,cam}$ towards a pixel is easily found using Equations (6.11)-(6.14). The cross product is taken in a clockwise direction, to make the vectors point inwards as wanted:

$$\begin{array}{llll} \boldsymbol{n}_{left,fcam} = {}^{E}\boldsymbol{v}_{0,0,fcam} \times {}^{E}\boldsymbol{v}_{0,480,fcam} & \boldsymbol{n}_{left,scam} = {}^{E}\boldsymbol{v}_{0,0,scam} \times {}^{E}\boldsymbol{v}_{0,480,scam} \\ \boldsymbol{n}_{top,fcam} = {}^{E}\boldsymbol{v}_{0,480,fcam} \times {}^{E}\boldsymbol{v}_{360,480,fcam} & \boldsymbol{n}_{top,scam} = {}^{E}\boldsymbol{v}_{0,480,scam} \times {}^{E}\boldsymbol{v}_{360,480,scam} \\ \boldsymbol{n}_{right,fcam} = {}^{E}\boldsymbol{v}_{360,480,fcam} \times {}^{E}\boldsymbol{v}_{360,0,fcam} & \boldsymbol{n}_{right,scam} = {}^{E}\boldsymbol{v}_{360,480,scam} \times {}^{E}\boldsymbol{v}_{360,0,scam} \\ \boldsymbol{n}_{bottom,fcam} = {}^{E}\boldsymbol{v}_{360,0,fcam} \times {}^{E}\boldsymbol{v}_{0,0,fcam} & \boldsymbol{n}_{bottom,scam} = {}^{E}\boldsymbol{v}_{360,0,scam} \times {}^{E}\boldsymbol{v}_{0,0,scam} \end{array} \tag{6.26}$$

From Equations (6.17)-(6.18), it is known that $\boldsymbol{p}_{front} = {}^{E}\boldsymbol{\kappa}_{fcam} - {}^{E}\boldsymbol{p}_{mrk,fcam}$ lies on the front camera planes, and $\boldsymbol{p}_{side} = {}^{E}\boldsymbol{\kappa}_{scam} - {}^{E}\boldsymbol{p}_{mrk,scam}$ lies on the side camera planes. By rearranging (6.25), the linear inequalities can be obtained:

$$\boldsymbol{n} \bullet (\boldsymbol{r} - \boldsymbol{p}) \geq 0 \tag{6.27}$$

$$\Updownarrow$$

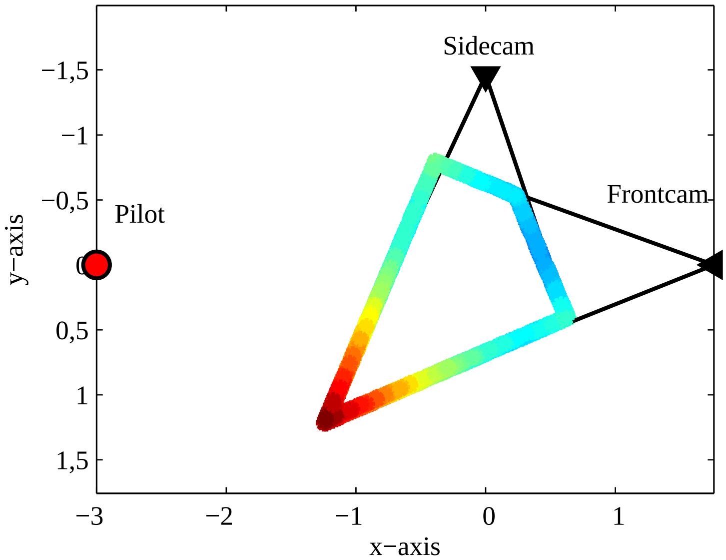
$$\boldsymbol{n} \bullet \Xi - \boldsymbol{n} \bullet \boldsymbol{p} \geq 0 \tag{6.28}$$

$$\Downarrow$$

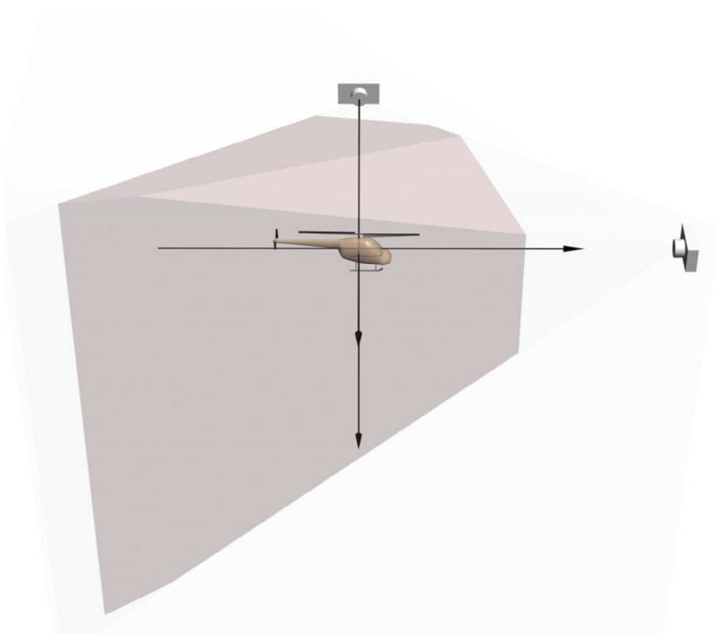
$$\mathcal{A} \cdot \Xi - \begin{bmatrix} \mathcal{A}_1 \cdot \boldsymbol{p}_{front} \\ \mathcal{A}_2 \cdot \boldsymbol{p}_{side} \end{bmatrix} \geq 0 \tag{6.29}$$

- $\mathcal{A} = \begin{bmatrix} \boldsymbol{n}^T_{left,fcam} \\ \vdots \\ \boldsymbol{n}^T_{bottom,scam} \end{bmatrix}$
  is an 8×3 matrix consisting of the eight normal vectors transposed.
- $\mathcal{A}_1$ is the first four rows of $\mathcal{A}$.
- $\mathcal{A}_2$ is the last four rows of $\mathcal{A}$.

These inequalities are implemented in the MATLAB function
[CD-ROM, 2007, `/source/image_processing/camera_is_in_view.m`].
To obtain a visualization of the ROO, a Monte Carlo simulation has been performed. 1 000 000 randomly chosen points in an area larger than the region of view are evaluated. All, which satisfies the linear inequalities, are plotted in a 3D plot, which is shown from different angles in Figure 6.12(a). In Figure 6.12(b), the obtained points from the simulation, is used to create a 3D figure.



(a) The ROO seen from above. All points, which are not close to a side camera bound, are removed.



(b) Region seen from the operating table in lab.

Figure 6.12: *3D plot of all points within the view of both cameras from a Monte Carlo simulation of* 1 000 000 *randomly chosen points. In Figure 6.12(b), the simulation points is used to create a 3D figure. The view is from the same perspective as the image in Figure 4.4 on page 26.*

6.2.5 Determination of the Helicopter Attitude

As written in Section 6.2.1 on page 88 it has been chosen to calculate the attitude of the helicopter, with basis in the known position of it, and the position of the marker in the helicam view. In section 6.2.4 the unit vector from the cameras towards the markers are found. It has been found that when the helicopter is close to hover as desired, then this method is most accurate for the roll and pitch angle, but not as accurate for the yaw angle. The method is therefore used for calculating ϕ and θ, and an alternative method is given for calculation of ψ.

Derivation of Algorithm for Calculating ϕ and θ

The unit vector from the helicam, towards the floor marker, is given in $\{B\}$ because the helicam is mounted on the helicopter. It is also possible to find the same unit vector given in $\{E\}$. This means that the same vector is obtained given both in $\{B\}$ and $\{E\}$.

$$
\begin{aligned}
{}^{B}\boldsymbol{v}_{mrk,hcam} &= \begin{bmatrix} -\sin(\theta_{mrk,hcam})\sin(\phi_{mrk,hcam}) \\ -\sin(\theta_{mrk,hcam})\cos(\phi_{mrk,hcam}) \\ \cos(\theta_{mrk,hcam}) \end{bmatrix} \quad , & (6.30) \\
{}^{E}\boldsymbol{v}_{mrk,hcam} &= \frac{{}^{E}\boldsymbol{\kappa}_{mrk,hcam} - \boldsymbol{\Xi}}{\|\,{}^{E}\boldsymbol{\kappa}_{mrk,hcam} - \boldsymbol{\Xi}\|_2} \quad . & (6.31)
\end{aligned}
$$

It is known that

$$
{}^{E}\boldsymbol{v}_{mrk,hcam} = {}^{E}_{B}\mathcal{R}\,{}^{B}\boldsymbol{v}_{mrk,hcam} \quad , \tag{6.32}
$$

and therefore it is possible to determine ${}^{E}_{B}\mathcal{R}$. The solution is however not unique because if the helicopter revolves around ${}^{B}\boldsymbol{v}_{mrk,hcam}$, the two vectors in (6.32) will remain the same even though ${}^{E}_{B}\mathcal{R}$ changes. It is however possible to find the "smallest" attitude satisfying (6.32), i.e. the attitude closest to zero. For example, if the helicopter position is zero $\left(\boldsymbol{\Xi} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T\right)$ and the position of the marker is in the center of the image, then $\phi = \theta = 0$ but ψ can be anything. The algorithm will find the smallest angle satisfying Equation (6.32), i.e. $\phi = \theta = \psi = 0$. This method is assumed satisfactory since it is wanted to keep the helicopter in hover and only small deviations from hover is expected.
According to Euler's eigenaxis rotation theorem a rigid body can be brought from any arbitrary initial orientation to an arbitrary final orientation by a rotation around an eigenaxis [Bak, 2002, p. 18]. The eigenaxis for the helicopter rotation can be found by the cross product of the two unit vectors. Since $\|\boldsymbol{a} \times \boldsymbol{b}\|_2 = \|\boldsymbol{a}\|_2 \cdot \|\boldsymbol{b}\|_2 \sin(\theta)$ (where θ is the angle between the vectors), and both the vectors have unit length it is easy to find the rotation angle.

$$
\begin{aligned}
\boldsymbol{e} &= {}^{E}\boldsymbol{v}_{mrk,hcam} \times {}^{B}\boldsymbol{v}_{mrk,hcam} & (6.33) \\
\theta_{rot} &= \arcsin(\|\boldsymbol{e}\|_2) & (6.34)
\end{aligned}
$$

- $\boldsymbol{e}$ is the eigenaxis around which ${}^{E}_{B}\mathcal{R}$ rotates the helicopter.
- θ_{rot} is the rotation angle around the eigenaxis.

From the eigenaxis and the rotation angle, a rotation matrix can be created [Bak, 2002]:

$$
{}^{E}_{B}\mathcal{R} = \begin{bmatrix} c(\theta_{rot}) + e_1^2(1-c(\theta_{rot})) & e_1e_2(1-c(\theta_{rot})) + e_3s(\theta_{rot}) & e_1e_3(1-c(\theta_{rot})) - e_2s(\theta_{rot}) \\ e_2e_1(1-c(\theta_{rot})) - e_3s(\theta_{rot}) & c(\theta_{rot}) + e_2^2(1-c(\theta_{rot})) & e_2e_3(1-c(\theta_{rot})) + e_1s(\theta_{rot}) \\ e_3e_1(1-c(\theta_{rot})) + e_2s(\theta_{rot}) & e_3e_2(1-c(\theta_{rot})) - e_1s(\theta_{rot}) & c(\theta_{rot}) + e_3^2(1-c(\theta_{rot})) \end{bmatrix} \tag{6.35}
$$

- $c(\theta_{rot}) = \cos(\theta_{rot})$ and $s(\theta_{rot}) = \sin(\theta_{rot})$

By inspection of

$$
{}^{E}_{B}\mathcal{R} = {}^{B}_{E}\mathcal{R}^{T} = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{6.36}
$$

(see Equation (5.53)) it can be seen that the Euler angles can be found as:

$$
\phi = \arctan\left(\frac{{}^{E}_{B}\mathcal{R}_{32}}{{}^{E}_{B}\mathcal{R}_{33}}\right) \tag{6.37}
$$

$$
\theta = \arcsin(-{}^{E}_{B}\mathcal{R}_{31}) \tag{6.38}
$$

$$
\psi = \arctan\left(\frac{{}^{E}_{B}\mathcal{R}_{12}}{{}^{E}_{B}\mathcal{R}_{11}}\right) \tag{6.39}
$$

- ${}^{E}_{B}\mathcal{R}_{xy}$ is the number in the x^{th} row and y^{th} column of $\mathcal{R}$.

Yaw estimation

As written, the above described method is not very precise for ψ when the helicopter is close to hover. Imagine the helicopter is in hover with $\mathbf{\Xi} = \mathbf{0}$ and $\mathbf{\Theta} = \begin{bmatrix} 0 & 0 & \psi \end{bmatrix}^T$. Then the position of the marker in the helicam view would be at the center of the image, even if ψ changes, and the above algorithm would result in an estimated yaw angle of 0. To find the correct yaw angle, a white "tap" is put on the marker on the floor, as shown in Figure 6.13. This is used in the marker location algorithm to derive the yaw angle.

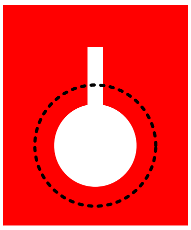


Figure 6.13: *The floor marker is expanded with a tap, to be able to find the yaw angle. The algorithm searches for most consecutive white pixels, at a distance from the marker center greater than the marker size.*

A search for white pixels, at a distance greater than the radius from the marker center, indicated by the black circle on Figure 6.13, is performed. The tap is then located where most consecutive white pixels occur. From this location it is possible to determine the angle.

This method has been implemented, but not tuned and tested in real flight. A proof of concept experiment has been conducted by manually holding the helicopter above the floor marker, and turning it. This has shown that it is possible to find the yaw angle of the helicopter.

6.2.6 Verification of the Position and Attitude Determination

The algorithms for calculating the position and attitude of the helicopter, has been tested qualitatively. This is done by giving several known pixel position input, and watching if the output behaves as expected. First all input pixel positions are set equal to the offset position ($(x_p,y_p) = (x_{off},y_{off})$ for all three cameras), and the helicopter markers are set to be located in the CM of the helicopter ($\boldsymbol{\kappa}_{mrk,fcam/scam} = \mathbf{0}$). This results in an output of $\boldsymbol{\Xi} = \mathbf{0}$, $\boldsymbol{\Theta} = \mathbf{0}$ and a position error estimate of 0, as expected. Next input as shown in Table 6.1 are given. The test showed that the output from the algorithm yielded the correct results.

Increase $x_{p,fcam}$ or decrease $x_{off,fcam}$	y decreases
Decrease $x_{p,fcam}$ or increase $x_{off,fcam}$	y increases
Increase $y_{p,fcam}$ or decrease $y_{off,fcam}$	z decreases
Decrease $y_{p,fcam}$ or increase $y_{off,fcam}$	z increases
Increase $x_{p,scam}$ or decrease $x_{off,scam}$	x decreases
Decrease $x_{p,scam}$ or increase $x_{off,scam}$	x increases
Increase $y_{p,scam}$ or decrease $y_{off,scam}$	z decreases
Decrease $y_{p,scam}$ or increase $y_{off,scam}$	z increases
Increase $x_{p,hcam}$ or decrease $x_{off,hcam}$	ϕ decreases
Decrease $x_{p,hcam}$ or increase $x_{off,hcam}$	ϕ increases
Increase $y_{p,hcam}$ or decrease $y_{off,hcam}$	θ increases
Decrease $y_{p,hcam}$ or increase $y_{off,hcam}$	θ decreases
Correct $\boldsymbol{\kappa}_{mrk,fcam}$	$\boldsymbol{\Xi} \approx \begin{bmatrix} 0 \\ -\kappa_{y,mrk,fcam} \\ -\kappa_{z,mrk,fcam}/2 \end{bmatrix}$ and $\boldsymbol{\Theta}$ changes according to $\boldsymbol{\kappa}_{mrk,fcam}$
Set $\boldsymbol{\kappa}_{mrk,fcam} = \mathbf{0}$ and correct $\boldsymbol{\kappa}_{mrk,scam}$	$\boldsymbol{\Xi} \approx \begin{bmatrix} -\kappa_{x,mrk,scam} \\ 0 \\ -\kappa_{z,mrk,scam}/2 \end{bmatrix}$ and $\boldsymbol{\Theta}$ changes according to $\boldsymbol{\kappa}_{mrk,scam}$

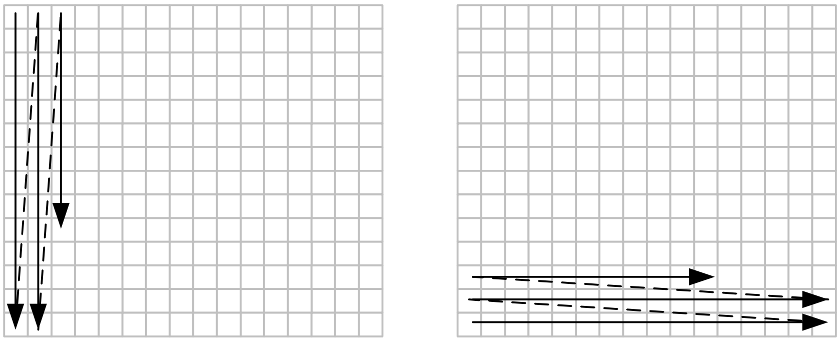
Table 6.1: *For different input to the algorithm for calculating the position and attitude of the helicopter, the corresponding output are shown. Note that the position error estimate increases with some input, because only changing the pixel position of one marker, is not possible in reality.*

6.2.7 Implementation of Marker Processing

The combined algorithms for finding the CM and attitude of the helicopter has been implemented in SIMULINK. The acquisition of the images from the webcams has been done using the image acquisition toolbox block ”video input”. Since this block only exists in the Windows version of MATLAB, Windows is chosen as OS for the IPC. The output from the “video input” blocks are three different matrices (*Red*, *Green* and *Blue*) the size of an image, each containing the values for one colour.

The CHT functions has been implemented in C++ by Anders la Cour-Harbo, and a wrapper file has been made to test the CHT on individual images. For implementation in

SIMULINK it has been chosen to implement the algorithm in an S-function, which makes it possible to use the existing implementation in C++. Each of the cameras are processed in a separate SIMULINK block, and the wallcams and helicam S-function are implemented in two different files (`wallcam_Sfun.cpp` and `helicam_Sfun.cpp`).
The indexing structure of a matrix in MATLAB is so that the first byte in the memory is the upper left part of the Matrix (i.e. pixel (1,res_v) in the image) and the next is the pixel below (1,$res_v - 1$). In a normal image the indexing starts from the lower left corner and advances to the right. This is illustrated in Figure 6.14.



(a) MATLAB matrix indexing.

(b) Bitmap data indexing.

Figure 6.14: Image (a) shows how MATLAB *indexes and saves an image matrix consecutively in the memory, and (b) shows how a bitmap image is saved in memory.*

Furthermore a bitmap image in the memory saves each RGB value of a pixel consecutively whereas the "video input" provides the three matrices individually. Therefore before proceeding with the CHT transform, the input matrices to the S-function must be merged and rearranged according to:

$$Mem[3 \cdot (k \cdot res_h + j)] = Red[res_h \cdot j - k + res_v - 1] \quad , \tag{6.40}$$
$$Mem[3 \cdot (k \cdot res_h + j) + 1] = Green[res_h \cdot j - k + res_v - 1] \quad , \tag{6.41}$$
$$Mem[3 \cdot (k \cdot res_h + j) + 2] = Blue[res_h \cdot j - k + res_v - 1] \quad , \tag{6.42}$$
$$j = 0,1,2\ldots,res_h - 1 \quad , \quad k = 0,1,2\ldots,res_v - 1 \quad . \tag{6.43}$$

- *Mem* is the memory block containing the image as shown in Figure 6.14b.
- *Red*,*Green*,*Blue* is the memory block provided by MATLAB

The calculation of the position of the helicopter CM as well as the attitude of the helicopter is implemented as a function in an m-file. This enables the calculation to be implemented in an m-file S-function in SIMULINK. The output from this block is sent to the CC.
A block diagram of the implementation in SIMULINK is shown in Figure 6.15.

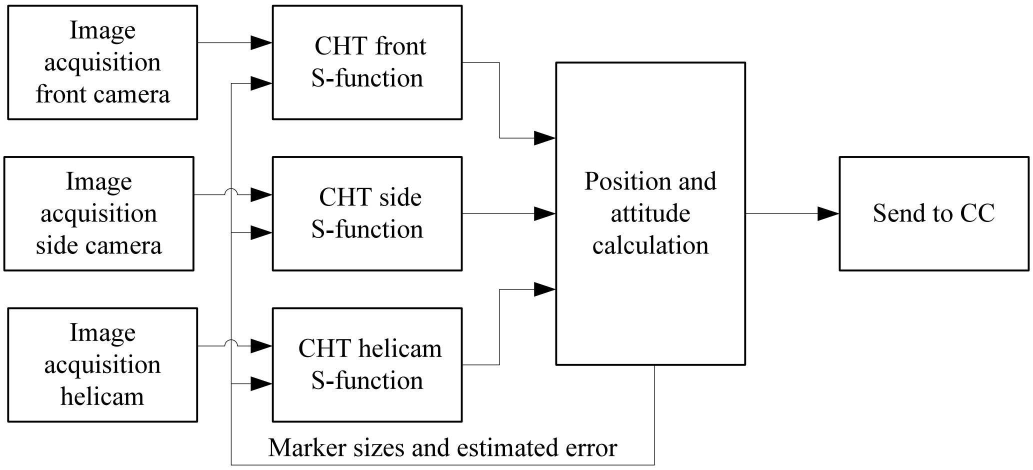


Figure 6.15: *A block diagram of the implementation of the image processing in* SIMULINK.

When starting the system it is necessary to calibrate the offsets ($x_{off,cam}$,$y_{off,cam}$). This is done by holding the helicopter in the inertial state and reading the output from the CHT. This output is then set as a parameter for the position and attitude calculation.

6.2.8 Connection between CC and IPC

For the CC to be able to use the derived position and attitude of the helicopter, it is necessary to transmit the data from the IPC to the CC. This is done by a simple serial connection using the "to instrument" block from the instrument control toolbox in SIMULINK. For easy implementation, it has been chosen to only send one byte for each state, which means that numbers between −128 and 127 can be send. Therefore the position is converted to cm and rounded before it is send. The attitude is also multiplied by 100, which gives an angle resolution of $\arcsin(1/100) = 0{,}01\,\text{rad} = 0{,}57\,°$. This is assumed sufficient in the current implementation. To separate each sample, a control character (chosen to be 125) is send at the end of each sample.

6.2.9 Test of the Image Processing Subsystem

The whole image processing subsystem, is quantitatively tested by placing the helicopter in different known positions, and watching the output. First, the marker algorithm is calibrated according to the description in Appendix G.1 on page 205. Then the helicopter is placed in zero position, and it is checked if the output is close to $\mathbf{\Xi} = \mathbf{0}, \mathbf{\Theta} = \mathbf{0}$. Then the helicopter is placed in two different positions, as far from the cameras as possible and as close to the cameras as possible, to see if the algorithm works in both near and far range. Further this tests shows if the algorithm can handle displacement in all three axes at the same time. The tested positions of the CM are measured to be:

$$\mathbf{\Xi}_1 = \begin{bmatrix} -0{,}75 \\ 0{,}80 \\ 0{,}50 \end{bmatrix} \quad , \quad \mathbf{\Xi}_1 = \begin{bmatrix} 0{,}22 \\ -0{,}43 \\ -0{,}31 \end{bmatrix} \text{m} \qquad . \tag{6.44}$$

These positions are obtained by watching the output display from the cameras, and moving the helicopter until the markers are close to the bounds of the cameras.

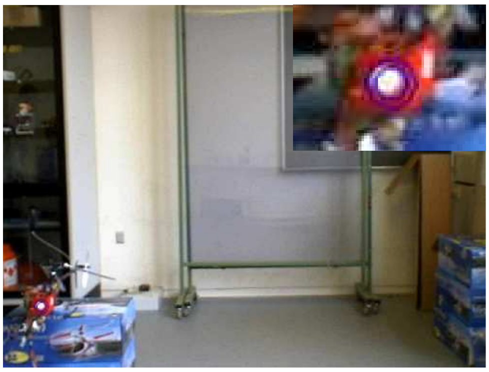
The positions are tested using the region of marker view algorithm, derived in 6.2.4 on page 102. The output inequalities yields the result:

$$\Xi_1 : \begin{bmatrix} 0{,}11 \\ 0{,}77 \\ 0{,}87 \\ 0{,}21 \\ 0{,}77 \\ 0{,}84 \\ 0{,}16 \\ 0{,}09 \end{bmatrix} \geq \mathbf{0} \quad , \quad \Xi_2 : \begin{bmatrix} 0{,}50 \\ 0{,}05 \\ 0{,}06 \\ 0{,}51 \\ 0{,}03 \\ 0{,}03 \\ 0{,}38 \\ 0{,}39 \end{bmatrix} \geq \mathbf{0} \quad , \tag{6.45}$$

which means, since all values are positive, that the markers should be within view. It is seen that inequality 1,4,7,8 for Ξ_1 yields low values. This indicates that the marker is close to the left and bottom bound for the front camera, and right and bottom bound for the side camera. This is correct according to the position of the helicopter. Similar observations can be done for Ξ_2. This verifies correct operation of the algorithm for determining if the markers are in view.
When the helicopter has been placed, the simulation is run for a few seconds to obtain the output from the function. While running, images including the detected marker is saved. The output is shown in Figure 6.16, where it can be seen that the markers are detected correct for all four images. The output pixelpositions of the marker center are:

$$\Xi_1 : \begin{bmatrix} (x_{p,fcam}, y_{p,fcam}) \\ (x_{p,scam}, y_{p,scam}) \end{bmatrix} = \begin{bmatrix} (49\,,83) \\ (523\,,40) \end{bmatrix} \quad , \quad \Xi_2 : \begin{bmatrix} (x_{p,fcam}, y_{p,fcam}) \\ (x_{p,scam}, y_{p,scam}) \end{bmatrix} = \begin{bmatrix} (600\,,434) \\ (74\,,445) \end{bmatrix} , \tag{6.46}$$

When plotting for the whole simulation period, it is seen that the pixel positions have a maximum deviation of 1 pixel, which indicates that the correct marker is found all the time.



(a) Front camera far away.

(b) Side camera far away.

(c) Front camera close up.

(d) Side camera close up.

Figure 6.16: *Images showing the marker position when testing the image processing subsystem. The markers are located correctly, and the algorithm determines the correct position.*

The output state etimate and estimated error, calculated on basis of the pixels are:

$$\hat{\Xi}_1 = \begin{bmatrix} -0{,}71 \\ 0{,}82 \\ 0{,}51 \end{bmatrix} \quad , \quad \hat{\Xi}_2 = \begin{bmatrix} 0{,}17 \\ -0{,}48 \\ -0{,}30 \end{bmatrix} \quad , \tag{6.47}$$

$$\hat{\Xi}_{err,1} = 0{,}04 \quad , \quad \hat{\Xi}_{err,2} = 0{,}02 \quad . \tag{6.48}$$

Comparing to the measured positions in (6.44), it is seen that the estimated position is close to the correct position. The small errors might be due to not completely horizontal alignment of the cameras. Further it is difficult to manually measure the exact position of the helicopter, så there might also be small errors in (6.44).
The size of the markers in the images have been measured by counting pixels in MSPaint. The blurred area in the edge of the markers has not been regarded as a part of the marker. The pixel radii of the images shown in Figures 6.16(a)-6.16(d) respectively are; 4, 4, 7 and 10. Calculated size of the markers (see Equation (6.23)) are 4, 5, 8 and 12. These are close to the correct size, though a little larger. This can be explained by not taking the blurred edge into account when measuring the radius.
From this test, it can be concluded that the image processing subsystem works as expected. Further tests of the image processing subsystem, and the connection to the CC, is done

together with the test of the state estimation algorithm (see Section 6.4 on page 121).

6.3 Sensor Fusion

Based on the output from the IMU and image preprocessing, it is possible fuse these measurements to an estimate of the helicopter states. However, only the 6 rigid body states and their derivates are estimated, as these are the ones which will be used in the controller. The state vector is thus reduced from dimension 30 (equation (5.1)) to dimension 12:

$$\boldsymbol{x} = \begin{array}{l} \left.\begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \psi \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}\right. \end{array} \begin{array}{l} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{E}\boldsymbol{\Xi} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{E}\boldsymbol{\Theta} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{B}\dot{\boldsymbol{\Xi}} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{B}\dot{\boldsymbol{\Theta}} \end{array} \quad . \tag{6.49}$$

In a conventional Kalman filter, the control input is propagated through the model to yield the predicted state output. Then this state prediction is updated using the sensor output. A schematic of this concept is illustrated in Figure 6.17.

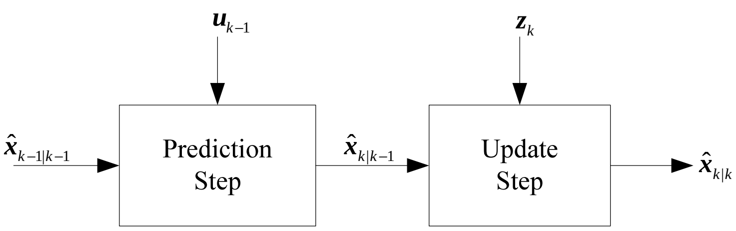


Figure 6.17: *The concept of a conventional Kalman filter.*

In our case, the control input is given by

$$\boldsymbol{u} = \begin{bmatrix} S_{mr} \\ S_{lat} \\ S_{lon} \\ \dot{\psi}_{ref} \end{bmatrix} \quad , \tag{6.50}$$

and the measurement vector is as mentioned in Section 6.1

$$z = \begin{bmatrix} \ddot{x}_{IMU} \\ \ddot{y}_{IMU} \\ \ddot{z}_{IMU} \\ \dot{\phi}_{IMU} \\ \dot{\theta}_{IMU} \\ \dot{\psi}_{IMU} \\ x_{cam} \\ y_{cam} \\ z_{cam} \\ \phi_{cam} \\ \theta_{cam} \\ \psi_{cam} \end{bmatrix} \quad . \tag{6.51}$$

6.3.1 Previous Work

Also in former helicopter projects, Kalman filtering has been used for estimation and sensor fusion. In Hald et al. [2006], two Kalman filters are implemented, one linear and one non-linear, which is linearized about a nominal trajectory (hover). The first is used to fuse the sensor measurements to get the velocity and attitude of the helicopter. These states are then regarded as new measurements for the next Kalman filter. This filter performs a full state estimate i.e. all the 30 states are estimated. This is an advantage since it is easier to make a linear model based controller when the whole state is known. A disadvantage by using full state estimation is that the information provided by the sensors are spread over more states and therefore more uncertainty is inevitably coupled to the estimate of each state.

In Mustafic et al. [2005] an Unscented Kalman Filter (UKF) is used to incorporate the non-linearities (only the non-linearities of the rigid body states are used and it is therefore not a full state estimate) of the model. However, this problem has only been treated theoretically.

In Bisgaard et al. [2007] two estimators are considered for a helicopter slung load system: a model free filter using the IMU measurements as control input, and a model based UKF. The conclusion of this article is that the UKF performes better than the IMU-driven filter, but at the cost of a larger computational burden.

6.3.2 IMU driven Extended Kalman Filter

On the basis of the previous work done in the department it is chosen to implement the Kalman filter as an IMU driven filter because this filter performs well with a small computational burden and the whole helicopter model is not needed in the filter. Compared to the full dynamic model described in Chapter 5, a much simpler rigid body process model is implemented based on the IMU output. The non-linear equations for the rigid body model are derived on which any type of Kalman filter can be applied. Comparing an UKF, and Extended Kalman Filter (EKF) and a Linearized Kalman Filter (LKF), the UKF is the most accurate but is more difficult to implement and has a larger computational burden as well. Since the LKF will have to be linearized in hover, it is assumed that some of the dynamics of the helicopter will be lost in this linearization, and further the LKF will loose accuracy the further away from hover the helicopter state is. On this basis it has in this project been chosen to implement the filter as an Extended Kalman Filter (EKF). Because an IMU driven filter is used, then instead of using (6.50) as the control input for the EKF, we will use

$$\boldsymbol{u} = \begin{bmatrix} \ddot{x}_{IMU} \\ \ddot{y}_{IMU} \\ \ddot{z}_{IMU} \\ \dot{\phi}_{IMU} \\ \dot{\theta}_{IMU} \\ \dot{\psi}_{IMU} \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = \ddot{\boldsymbol{\Xi}}_{IMU} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = \dot{\boldsymbol{\Theta}}_{IMU} \end{matrix} \quad . \tag{6.52}$$

The measurement vector is then limited to the last six elements of $\boldsymbol{z}$ in (6.51):

$$\boldsymbol{z} = \begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \\ \phi_{cam} \\ \theta_{cam} \\ \psi_{cam} \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{E}\boldsymbol{\Xi}_{cam} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} = {}^{E}\boldsymbol{\Theta}_{cam} \end{matrix} \quad . \tag{6.53}$$

The one step iteration of the non-linear dynamic model in an EKF is given by

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{w}_{k-1}) \quad . \tag{6.54}$$

- $\boldsymbol{f}$ is a 12-dimensional vector function.
- $\boldsymbol{x}_{k-1}$ is the state at time $k-1$.
- $\boldsymbol{u}_{k-1}$ is the control input at time $k-1$.
- $\boldsymbol{w}_{k-1}$ is the process noise at time $k-1$, assumed to be normal distributed with zero mean and covariance $\boldsymbol{Q}_k$ ($\boldsymbol{w}_k \sim N(0,\boldsymbol{Q}_k)$).

and the measurement model is given by

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{v}_k) \quad . \tag{6.55}$$

- $\boldsymbol{h}$ is a 6-dimensional vector function.
- $\boldsymbol{x}_k$ is the state at time k.
- $\boldsymbol{v}_k$ is the measurement noise at time k, assumed to be normal distributed with zero mean and covariance $\mathcal{R}_k$ ($\boldsymbol{v}_k \sim N(0,\mathcal{R}_k)$).

Figure 6.18 shows a block diagram of the Kalman filter with the noise matrices added. Also the estimation error (a posteriori) covariance matrix $\hat{\mathcal{P}}_{k|k}$ and the prediction error (a priori) covariance matrix $\hat{\mathcal{P}}_{k|k-1}$ are shown. Below, the equations for the prediction step and the update step, respectively, are given [Grewal and Andrews, 2001, p. 180].

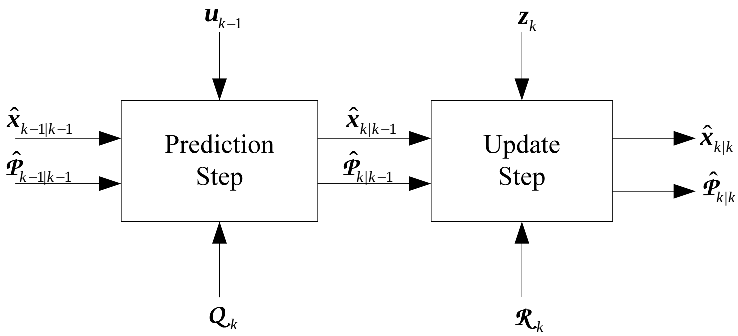


***Figure 6.18:** The concept of a Kalman filter with noise and covariance matrices added.*

Predict

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{0}) \quad , \tag{6.56}$$

$$\hat{\mathcal{P}}_{k|k-1} = \mathcal{F}_{k-1}\hat{\mathcal{P}}_{k-1|k-1}\mathcal{F}_{k-1}{}^T + \mathcal{Q}_{k-1} \quad . \tag{6.57}$$

- ► $\hat{\boldsymbol{x}}_{k-1|k-1}$ is the estimated state vector at time $k-1$.
- ► $\hat{\boldsymbol{x}}_{k|k-1}$ is the predicted state vector at time k.
- ► $\hat{\mathcal{P}}_{k|k-1}$ is the prediction error covariance matrix of the filter at time k given $\hat{\mathcal{P}}$ at time $k-1$.
- ► $\mathcal{F}_{k-1}$ is the linearized state propagation matrix at time $k-1$, i.e. the Jacobian of $\boldsymbol{f}$ with $\hat{\boldsymbol{x}}_{k-1|k-1}$ inserted:

$$\mathcal{F}_{k-1} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{k-1|k-1}} \quad .$$

Update

$$\mathcal{K}_k = \hat{\mathcal{P}}_{k|k-1}\mathcal{H}_k{}^T(\mathcal{H}_k\hat{\mathcal{P}}_{k|k-1}\mathcal{H}_k{}^T + \mathcal{R}_k)^{-1} \quad , \tag{6.58}$$

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \mathcal{K}_k(\boldsymbol{z}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_{k|k-1}, \boldsymbol{0})) \quad , \tag{6.59}$$

$$\hat{\mathcal{P}}_{k|k} = \hat{\mathcal{P}}_{k|k-1} - \mathcal{K}_k\mathcal{H}_k\hat{\mathcal{P}}_{k|k-1} \quad . \tag{6.60}$$

- ► $\mathcal{K}_k$ is the Kalman gain at time k.
- ► $\mathcal{H}_k$ is the linearized measurement model matrix given the predicted state, i.e. the Jacobian of $\boldsymbol{h}$ with $\hat{\boldsymbol{x}}_{k|k-1}$ inserted:

$$\mathcal{H}_k = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{k|k-1}} \quad .$$

The Process Model

The process model is derived for the four states ${}^E\boldsymbol{\Xi}$, ${}^E\boldsymbol{\Theta}$, ${}^B\dot{\boldsymbol{\Xi}}$, ${}^B\dot{\boldsymbol{\Theta}}$ independently. The first three are calculated as first order euler integrations in discrete time, i.e. the old value plus the derivative multiplied by the sampling time:

$$ {}^{E}\hat{\mathbf{\Xi}}_{k|k-1} = T_s \cdot \left({}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}\, {}^{B}\hat{\dot{\mathbf{\Xi}}}_{k-1|k-1} \right) + {}^{E}\hat{\mathbf{\Xi}}_{k-1|k-1} \qquad . \tag{6.61} $$

- ${}^{E}\hat{\mathbf{\Xi}}_{k|k-1}$ is the predicted position of the helicopter given in $\{E\}$ at time k.
- T_s is the sampling time.
- ${}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}$ is the estimated rotation matrix at time $k-1$ converting a vector given in $\{B\}$ to a vector given in $\{E\}$.
- ${}^{B}\hat{\dot{\mathbf{\Xi}}}_{k-1|k-1}$ is the estimated translatory velocity of the helicopter given in $\{B\}$ at time $k-1$.
- ${}^{E}\hat{\mathbf{\Xi}}_{k-1|k-1}$ is the estimated position of the helicopter given in $\{E\}$ at time $k-1$.

$$ {}^{E}\hat{\mathbf{\Theta}}_{k|k-1} = T_s \cdot \left({}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}\, {}^{B}\hat{\dot{\mathbf{\Theta}}}_{k-1|k-1} \right) + {}^{E}\hat{\mathbf{\Theta}}_{k-1|k-1} \qquad . \tag{6.62} $$

- ${}^{E}\hat{\mathbf{\Theta}}_{k|k-1}$ is the predicted attitude of the helicopter given in $\{E\}$ at time k.
- ${}^{B}\hat{\dot{\mathbf{\Theta}}}_{k-1|k-1}$ is the estimated angular velocity of the helicopter given $\{B\}$ at time $k-1$.
- ${}^{E}\hat{\mathbf{\Theta}}_{k-1|k-1}$ is the estimated attitude of the helicopter given in $\{E\}$ at time $k-1$.

$$ {}^{B}\hat{\dot{\mathbf{\Xi}}}_{k|k-1} = T_s \cdot \left({}^{B}\hat{\ddot{\mathbf{\Xi}}}_{k-1|k-1} \right) + {}^{B}\hat{\dot{\mathbf{\Xi}}}_{k-1|k-1} \qquad . \tag{6.63} $$

- ${}^{B}\hat{\dot{\mathbf{\Xi}}}_{k|k-1}$ is the translatory velocity of the helicopter given in $\{B\}$ at time k.
- T_s is the sampling time.
- ${}^{B}\hat{\ddot{\mathbf{\Xi}}}_{k-1|k-1}$ is the translatory acceleration of the helicopter given $\{B\}$ at time $k-1$.
- ${}^{B}\hat{\dot{\mathbf{\Xi}}}_{k-1|k-1}$ is the velocity of the helicopter given in $\{B\}$ at time $k-1$.

Since it has been chosen to use an IMU driven Kalman filter, the IMU measurements are used as input to derive $\hat{\ddot{\mathbf{\Xi}}}$. The IMU incoorporates the gravitational field of the Earth in its output, such that when the helicopter is standing on the ground, the IMU measures a negative (upward) acceleration at the ${}^{E}z$-axis, which is pointing downwards. It might be easier to understand if the opposite situation is considered: When the IMU measures zero acceleration on all axes, it is because it is falling towards the Earth with a positive acceleration of 9,82 m/s². So to stop this acceleration, it is necessary to apply a negative acceleration of equal size to the IMU. Thus,

$$ \ddot{\mathbf{\Xi}}_{IMU} = \left\{ {}^{B}\ddot{\mathbf{\Xi}} \right\}_E - {}^{B}_{E}\mathcal{R}\, {}^{E}\boldsymbol{g} \qquad . \tag{6.64} $$

- $\ddot{\mathbf{\Xi}}_{IMU}$ is the IMU measurement.
- $\left\{ {}^{B}\ddot{\mathbf{\Xi}} \right\}_E$ is the acceleration of the body frame as seen from the earth frame, and given in $\{B\}$.
- ${}^{B}_{E}\mathcal{R}$ is the rotation matrix converting a vector given in $\{E\}$ to a vector given in $\{B\}$.
- ${}^{E}\boldsymbol{g}$ is the gravitational acceleration vector given in $\{E\}$.

$\left\{ \ddot{\mathbf{\Xi}} \right\}_E$ is found using the transport theorem (see Section 5.1.8):

$$ \left\{ {}^{B}\ddot{\mathbf{\Xi}} \right\}_E = \left\{ {}^{B}\ddot{\mathbf{\Xi}} \right\}_B + {}^{B}\dot{\mathbf{\Theta}} \times {}^{B}\dot{\mathbf{\Xi}} \qquad . \tag{6.65} $$

- $\left\{{}^{B}\ddot{\Xi}\right\}_B$ is the acceleration of the body frame as seen from the body frame, and given in $\{B\}$.
- ${}^{B}\dot{\Theta}$ is the angular velocity of $\{B\}$ given in $\{B\}$.
- ${}^{B}\dot{\Xi}$ is the translatory velocity of $\{B\}$ given in $\{B\}$.

In Equation (6.63) the acceleration $\hat{\ddot{\Xi}}$ equals $\left\{\ddot{\Xi}\right\}_B$ from Equation (6.65). So now we can combine Equations (6.63), (6.64) and (6.65) to yield the final process model for $\dot{\Xi}$:

$$\begin{aligned}
{}^{B}\hat{\dot{\Xi}}_{k|k-1} &= T_s \cdot \left(\left\{{}^{B}\ddot{\Xi}\right\}_B\right) + {}^{B}\hat{\dot{\Xi}}_{k-1|k-1} \quad . \\
&\Updownarrow \\
{}^{B}\hat{\dot{\Xi}}_{k|k-1} &= T_s \cdot \left(\left\{{}^{B}\ddot{\Xi}\right\}_E - {}^{B}\hat{\dot{\Theta}}_{k-1|k-1} \times {}^{B}\hat{\dot{\Xi}}_{k-1|k-1}\right) + {}^{B}\hat{\dot{\Xi}}_{k-1|k-1} \quad . \\
&\Updownarrow \\
{}^{B}\hat{\dot{\Xi}}_{k|k-1} &= T_s \cdot \left(\ddot{\Xi}_{IMU,k-1} + {}^{B}_{E}\hat{\mathcal{R}}_{k-1|k-1}{}^{E}g - {}^{B}\hat{\dot{\Theta}}_{k-1|k-1} \times {}^{B}\hat{\dot{\Xi}}_{k-1|k-1}\right) + {}^{B}\hat{\dot{\Xi}}_{k-1|k-1} .
\end{aligned} \tag{6.66}$$

The process model for $\dot{\Theta}$ is using the IMU measurement $\dot{\Theta}_{IMU}$ directly, but to incoorporate the measurement in the filter, a linear weighting between the state and the measurement is introduced:

$$ {}^{B}\hat{\dot{\Theta}}_{k|k-1} = \begin{bmatrix} \dot{\Theta}_{IMU,k-1} & {}^{B}\hat{\dot{\Theta}}_{k-1|k-1} \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} \quad , \quad (q \in [0;1]) $$

- ${}^{B}\hat{\dot{\Theta}}_{k|k-1}$ is the predicted angular velocity of the helicopter at time k.
- $\dot{\Theta}_{IMU,k-1}$ is the angular measurements of the IMU.
- ${}^{B}\hat{\dot{\Theta}}_{k-1|k-1}$ is the estimated angular velocity of the helicopter at time $k-1$.
- q is the weighting factor, which is a scalar between 0 and 1.

By using the weighting factor, q, a first order autoregressive (AR(1)) process is defined, which functions as a low pass filter with the cutoff frequency determined by the value of q. The value of q will be found by handtuning when testing the filter.

To sum up, the four state transition models are as follows:

$$ {}^{E}\hat{\Xi}_{k|k-1} = T_s \cdot \left({}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}{}^{B}\hat{\dot{\Xi}}_{k-1|k-1}\right) + {}^{E}\hat{\Xi}_{k-1|k-1} \quad , \tag{6.67} $$

$$ {}^{E}\hat{\Theta}_{k|k-1} = T_s \cdot \left({}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}{}^{B}\hat{\dot{\Theta}}_{k-1|k-1}\right) + {}^{E}\hat{\Theta}_{k-1|k-1} \quad , \tag{6.68} $$

$$ {}^{B}\hat{\dot{\Xi}}_{k|k-1} = T_s \cdot \left(\ddot{\Xi}_{IMU,k-1} + {}^{B}_{E}\hat{\mathcal{R}}_{k-1|k-1}{}^{E}g - {}^{B}\hat{\dot{\Theta}}_{k-1|k-1} \times {}^{B}\hat{\dot{\Xi}}_{k-1|k-1}\right) + {}^{B}\hat{\dot{\Xi}}_{k-1|k-1} \tag{6.69} $$

$$ {}^{B}\hat{\dot{\Theta}}_{k|k-1} = \begin{bmatrix} \dot{\Theta}_{IMU,k-1} & {}^{B}\hat{\dot{\Theta}}_{k-1|k-1} \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} \quad . \tag{6.70} $$

These four equations constitutes the full process model f, as given in Equation (6.56). To verify that the model is correct, a series of tests with a known input is given to the model. From the corresponding output, it is in Appendix E on page 189 verified that the model is correct.

The Measurement Model

The measurement model $\boldsymbol{h}$, given in Equation (6.55), is linear since the 6 measurements coming from the image processing is the direct measurement of 6 of the states:

$$\boldsymbol{h}(\hat{\boldsymbol{x}}_{k|k-1},\boldsymbol{0}) = \begin{bmatrix} \hat{x}_{k|k-1} \\ \hat{y}_{k|k-1} \\ \hat{z}_{k|k-1} \\ \hat{\phi}_{k|k-1} \\ \hat{\theta}_{k|k-1} \\ \hat{\psi}_{k|k-1} \end{bmatrix} \quad . \tag{6.71}$$

6.3.3 Implementation

The IMU-driven EKF has been implemented in an m-file, which is used in Simulink as an embedded S-function, called `EKF_single_iteration_sfun`. This function takes the following input: T_s, $\boldsymbol{x}_{k-1}$, $\boldsymbol{z}$, $\boldsymbol{u}$, $\hat{\mathcal{P}}_{k-1}$ and q, and returns $\boldsymbol{x}_k$ and $\hat{\mathcal{P}}_k$. In Simulink the output are fed back to the corresponding input through a unit delay. The unit delay block is also initialising $\boldsymbol{x}$ and $\hat{\mathcal{P}}$ when the filter is started. $\boldsymbol{x}_1$ is set to $\boldsymbol{0}$, and $\hat{\mathcal{P}}_1$ is set to $\mathcal{I}_{12\times12}$.
Inside the `EKF_single_iteration_sfun` block, the noise matrices $\mathcal{Q}$ and $\mathcal{R}$ are defined. It is assumed that they are constant, and that there are no cross couplings between the axes (e.g. the covariance between x and y or between $\dot{y}$ and $\dot{z}$ is zero). The process noise $\mathcal{Q}$ is derived on basis of noise measurements from the IMU. The variance of the IMU is measured to:

$$\sigma^2_{acc,IMU} = 1\,\mathrm{m^2/s^4} \quad , \quad \sigma^2_{rot,IMU} = 0{,}25\,\mathrm{rad^2/s^2} \quad . \tag{6.72}$$

- $\sigma^2_{acc,IMU}$ is the noise variance on the three acclerometers. Notice that the value is not correct, since the accelerometers are saturated when flying and hence does not yield a valid output. It is though estimated that the value is close to the correct noise variance.
- $\sigma^2_{rot,IMU}$ is the noise variance on the three rotation measurements.

By integrating a variable, the variance is multiplied by the sampletime. This is exemplified by the covariance between x and $\dot{x}$:

$$E\left[(x-\bar{x})\cdot(\dot{x}-\bar{\dot{x}})\right] = E\left[T_s\cdot(\dot{x}-\bar{\dot{x}})\cdot(\dot{x}-\bar{\dot{x}})\right] = T_s\cdot E\left[(\dot{x}-\bar{\dot{x}})\cdot(\dot{x}-\bar{\dot{x}})\right] \quad . \tag{6.73}$$

- T_s is the sample time.
- $E[\cdot]$ is the expected value.

Using this property, the process noise becomes:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & T_s & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_s & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_s & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_s & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_s & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & T_s \\ T_s^{-1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & T_s^{-1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & T_s^{-1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & T_s^{-1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & T_s^{-1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & T_s^{-1} & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} T_s^4 \cdot \sigma^2_{acc,IMU} \\ T_s^4 \cdot \sigma^2_{acc,IMU} \\ T_s^4 \cdot \sigma^2_{acc,IMU} \\ T_s^2 \cdot \sigma^2_{rot,IMU} \\ T_s^2 \cdot \sigma^2_{rot,IMU} \\ T_s^2 \cdot \sigma^2_{rot,IMU} \\ T_s^2 \cdot \sigma^2_{acc,IMU} \\ T_s^2 \cdot \sigma^2_{acc,IMU} \\ T_s^2 \cdot \sigma^2_{acc,IMU} \\ \sigma^2_{rot,IMU} \\ \sigma^2_{rot,IMU} \\ \sigma^2_{rot,IMU} \end{bmatrix} . \tag{6.74}$$

Regarding the measurement noise matrix $\mathcal{R}$, the output from the IPC has been observed during operation. The standard deviation for the position is around 5 cm and for the attitude around 0,1 rad. This means that the variances and the measurement noise matrix are:

$$\sigma^2_{pos,IPC} = 0{,}0025\,\mathrm{m}^2 \quad , \quad \sigma^2_{\mathrm{att,IPC}} = 0{,}01\,\mathrm{rad}^2 \qquad . \tag{6.75}$$

- $\sigma^2_{pos,IPC}$ is the noise variance on the three position measurements from the IPC.
- $\sigma^2_{att,IPC}$ is the noise variance on the three attitude measurements from the IPC.

$$\mathcal{R} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \sigma^2_{pos,IPC} \\ \sigma^2_{pos,IPC} \\ \sigma^2_{pos,IPC} \\ \sigma^2_{att,IPC} \\ \sigma^2_{att,IPC} \\ \sigma^2_{att,IPC} \end{bmatrix} \qquad . \tag{6.76}$$

Even though the noise variances has been found experimentally, the values are not believed to be very accurate, and some tuning might be necessary when the Kalman filter is implemented.

Because of the rotation matrix in the process model, the expression for f is extensive. Therefore MAPLE has been used to derive the Jacobian of f. The result is a 12×12 matrix, which can be found in the Maple file on
[CD-ROM, 2007, `./source/maple/kalman_filter.mw`]. The Jacobian of h is more straightforward since it is linear already:

$$\mathcal{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad . \tag{6.77}$$

On [CD-ROM, 2007, `./source/state_estimation`] the following files can be found:

- `EKF.mdl`. The SIMULINK model.
- `EKF_single_iteration.m`. The source code for the Kalman filter.
- `EKF_single_iteration_sfun.m`. The S-function calling `EKF_single_iteration.m`.
- `rotation_matrix.m`. A subfunction used in `EKF_single_iteration.m` for rotation matrix calculations.

6.4 Test and Verification of State Estimation

In this section the combined state estimation system containing marker localization, position and attitude determination, connection between IPC and CC, and sensor fusion is tested. The tests has been performed by moving the helicopter around in the region of camera view, while the whole system is running. In this way the all the subsystems are verified, because if one subsystem is not working correctly, then the output of the state estimate will be erroneous. Three tests have been done: a translatory test, an attitude test and a step test.

6.4.1 Translatory Test

When holding the helicopter around zero position, both the CC and IPC is started. The helicopter is subsequently moved in the positive direction of each axis and back again, while it is tried to keep the attitude zero. The output from the position algorithm on the IPC is shown in the upper graph in Figure 6.19. The corresponding input received on the CC is shown in the lower graph.

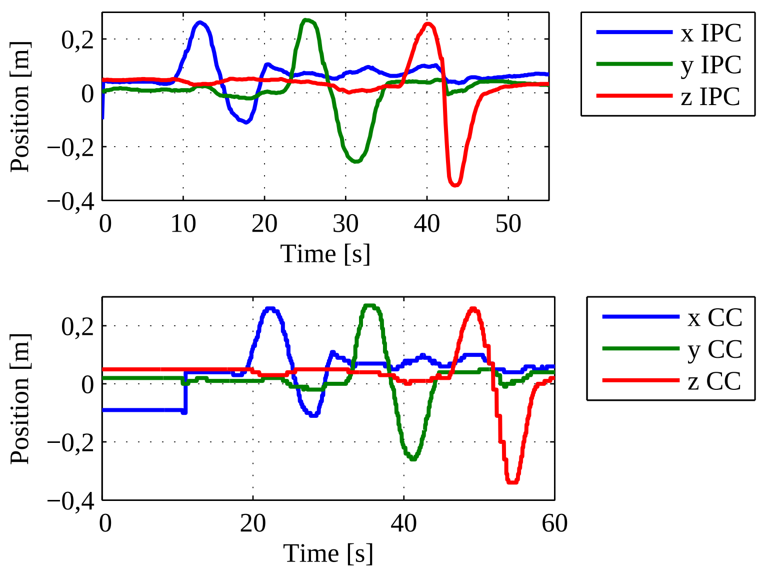


Figure 6.19: *Comparison of the output from the IPC and the input on the CC. It can be seen that there is some time offset coming from different start times of the simulation. Further the quantization on the input to the CC can be seen.*

The figure verifies that the connection between the two computers works as expected. On the input to the CC some staircase form can be seen. This might be due to the computers not running real time. The CC runs pseudo real time, which means that if a whole sample time has not elapsed when one iteration is calculated, then the computer sleeps until the whole sample time has elapsed. This should not yield any problems, since the computer is approximately $3-4$ times as fast as necessary, when using the current implementation. The image processing computer however, is heavy loaded with the marker localization algorithm. The computer works as fast as possible, and no real time considerations has been done. It has been observed that the simulation some times slows down a bit with no obvious reason. The reason for this is assumed to be because Windows is used as operating system and some times interrupts the processing. When running optimal, it is only possible to obtain a sample rate of around 20 Hz. But, for example if the marker is outside the region of interest, this sample rate is lowered. The above would result in the sample time fluctuations on the CC, i.e. the samples coming to the CC is not with equidistant time intervals. Furthermore, the serial connection on the CC is heavy loaded since it is used for both the servoboard connection, the IMU and the IPC. This might also lead to congestion in the serial connection, which also might lead to the above problem. Means to solve these issues has not been investigated further. Some of the staircase effect is also due to the quantization level used when sending the data over the serial connection. Figure 6.19 also shows that the output corresponds to the trajectory to of the helicopter, and therefore that the whole image processing system works. Looking at the graphs, it is seen that the time offset is different. This is because the computers were not started at the same time. Because the computers run individually, the time axis is also not completely correct for any of the computers. To be able to compare the signals on both computers, an identical time axis must be found. The IMU acceleration data has been studied to be able to see if position changes can be linked to acceleration changes. The IMU acceleration data is plotted together with the position data in Figure 6.20. It can be seen that no significant acceleration changes occur when the helicopter moves.

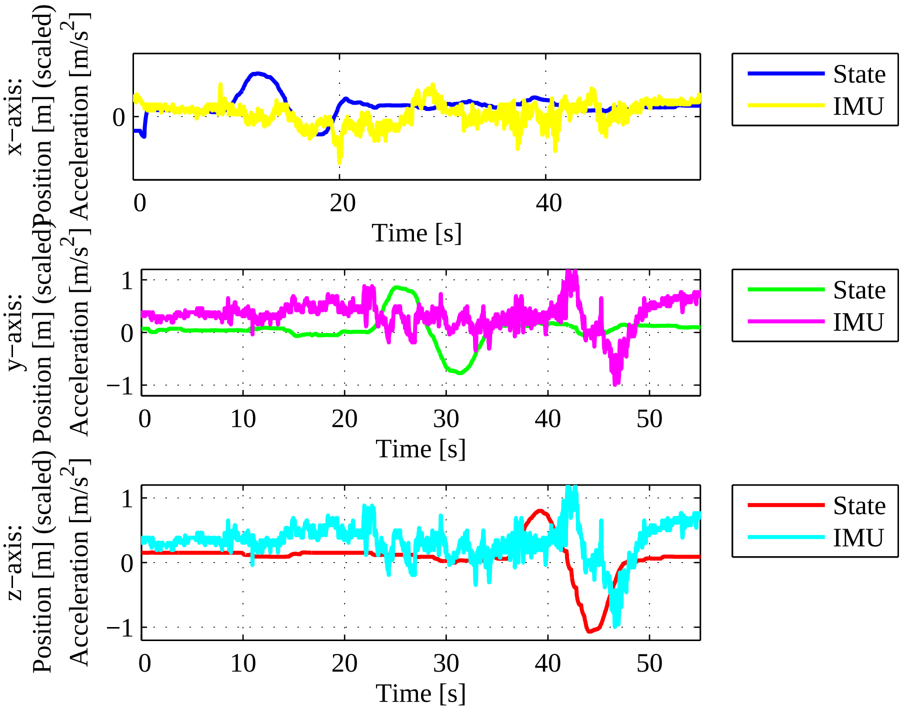
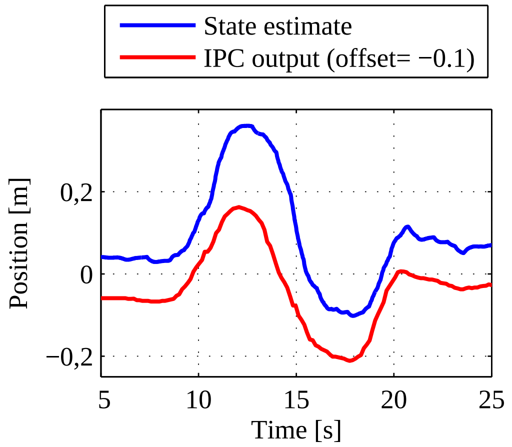
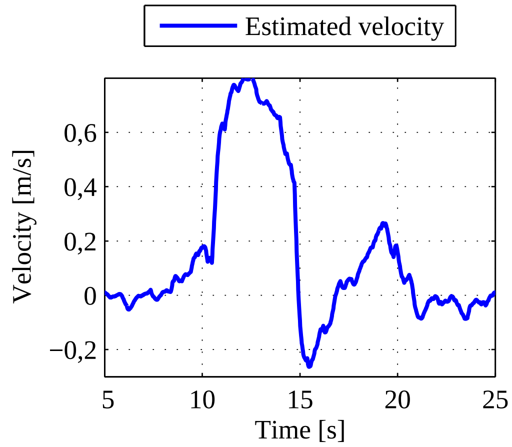


Figure 6.20: *Comparision of the IMU acceleration data, and the state estimate. Notice that $\ddot{z}$ is plotted with an offset of* 9,82 m/s² *to account for the gravity.*

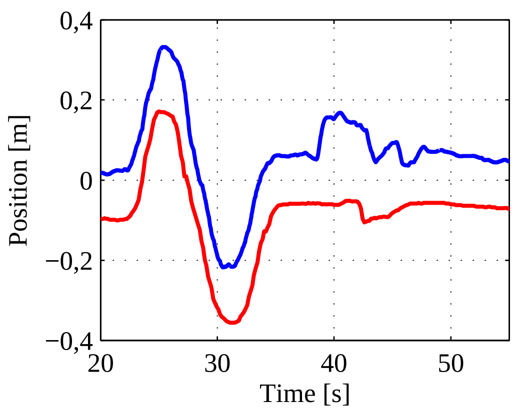
Instead of using the IMU data, an identical start and an end point for each of the two graphs in Figure 6.19 has been located. For example the point where x starts to rise can be used as a start point. From this information, the time axis on the IPC has been stretched to match the CC time axis. This has been done on all the following figures in this section. To verify that the Kalman filter used for sensor fusion works as expected, the output position is compared to the output from the IPC. In Figures 6.21(a), 6.21(c) and 6.21(e) these are plotted together for the x,y, and z axis respectively. An offset on the IPC output has been added to be able to distinguish the lines. It is seen that the input and output corresponds well to each other. However in Figure 6.21(e) it is seen that the z axis has been displaced. This is due to the sample time fluctuations described above. In Figures 6.21(b), 6.21(d) and 6.21(f) the velocity output from the Kalman filter is shown. This output also corresponds well to the estimated positions and the actual movement of the helicopter.



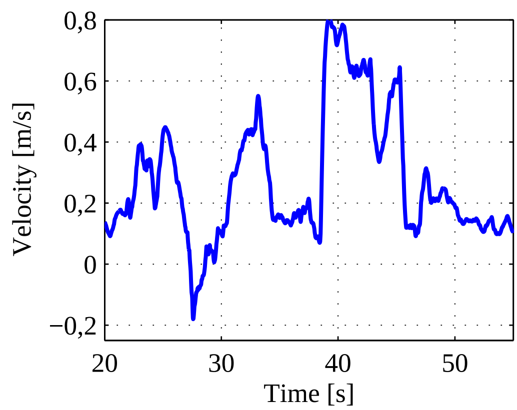
(a) x-axis input and output to/from the Kalman filter.



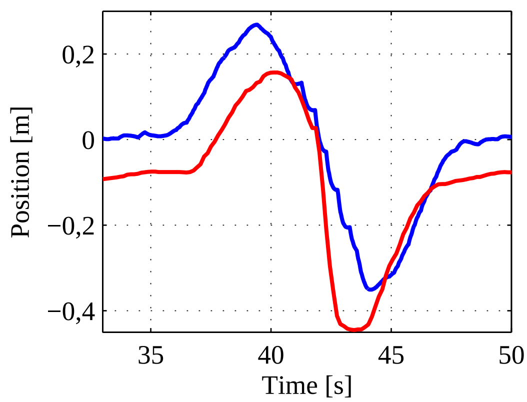
(b) $\dot{x}$ output from the Kalman filter.



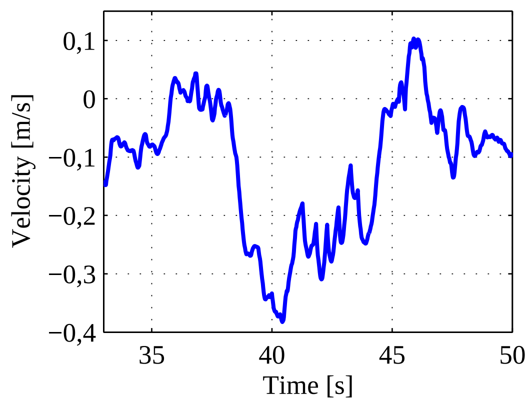
(c) y-axis input and output to/from the Kalman filter.



(d) $\dot{y}$ output from the Kalman filter.



(e) z-axis input and output to/from the Kalman filter.

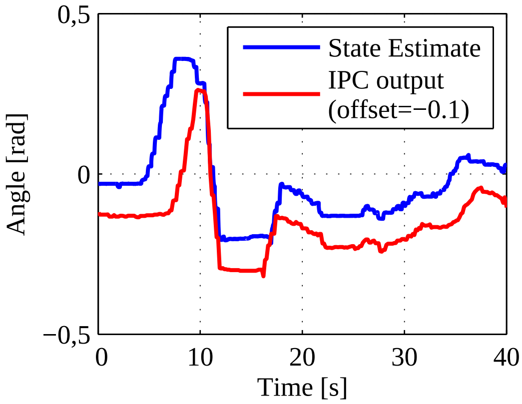


(f) $\dot{z}$ output from the Kalman filter.

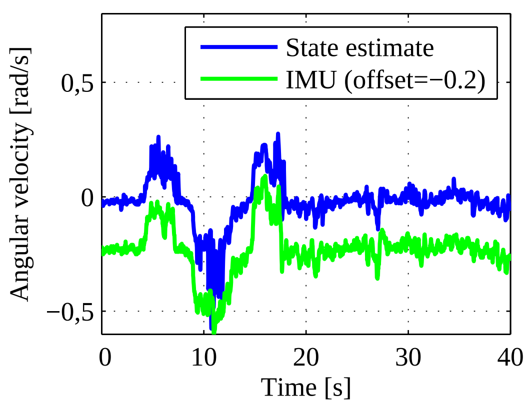
Figure 6.21: *The figure shows the output from the kalman filter for the translatory test. The left figures shows the output from the kalman filter together with the output on the IPC. An offset is added to avoid the curves being on top of each other. The right figures shows the velocity output from the Kalman filter. It can be seen that the sensor fusion works as expected for translatory movements. Notice that the time axis are different, and the figures are segments of the whole test shown in Figure 6.19.*

6.4.2 Attitude Test

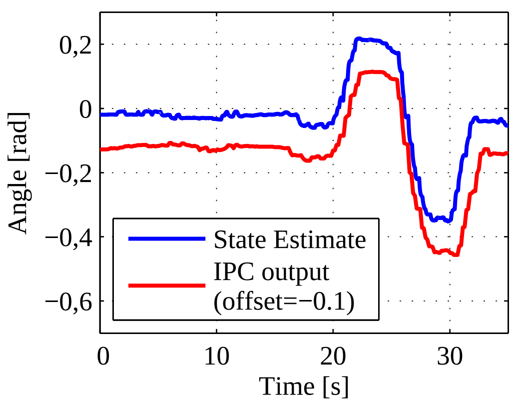
The second test is performed to check the attitude state estimation. As when testing the translatory movement, the helicopter is held around zero position. Then a consecutive roll, pitch and yaw motion is performed. In Figure 6.22(a) the attitude output is shown for the pitch and roll movement. As well as for the translatory movement, both the image processing and the sensor fusion yields the correct results. The estimated angular velocity is shown in Figure 6.22(b), and this also corresponds well to the input. In contrast to the IMU acceleration measurements, it is in this case seen that it is possible to see rotation on the IMU rotation measurements. When zooming in, it is seen that the Kalman filter smothes the angular velocity.



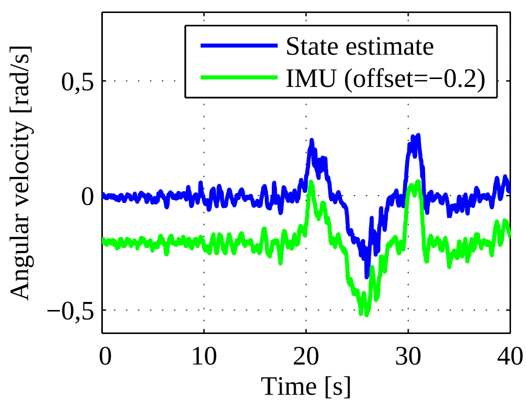
(a) State estimate and IPC output for ϕ.



(b) State estimate and IMU measurement for ϕ.



(c) State estimate and IPC output for θ.



(d) State estimate and IMU measurement for θ.

Figure 6.22: *Test results for the attitude estimation test. Figure 6.22(a) and Figure 6.22(c) shows the output attitude estimate for ϕ and θ from the IPC and CC respectively. Figure 6.22(b) and Figure 6.22(d) shows the estimated angular velocity ($\dot{\phi}$ and $\dot{\theta}$) of the helicopter.*

As written in Section 6.2.5 on page 106, the algorithm is not fully developed yet. It is however tested if it is possible to get a plausible result from the algorithm. In Figure 6.23 the output from the yaw estimation is shown. It is seen that the yaw estimation works, but some outliers occurs when positive yaw is performed. This should be investigated further before using the yaw estimation algorithm.

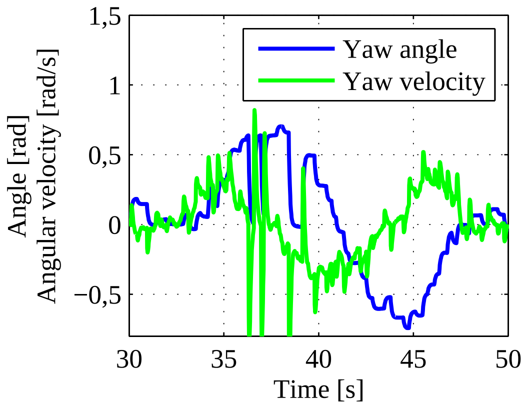
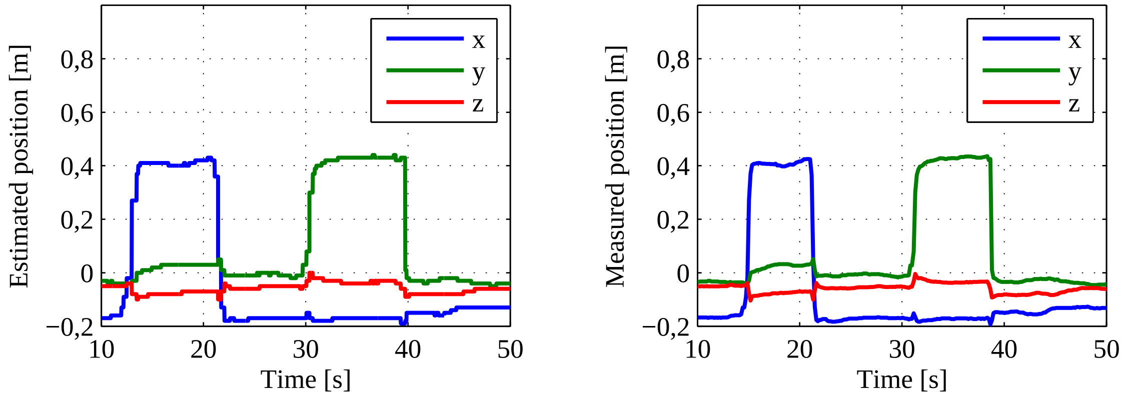


Figure 6.23: *Output from the yaw estimation algorithm. It is seen that it is possible to estimate the yaw angle, but some sudden jumps occurs when performing positive yaw.*

6.4.3 Step Test

To see if the marker is still found if the helicopter is moved fast, a step test is performed. As with the two preceding tests, the helicopter is held still. Then first a sudden movement in the x direction is performed, and then in the y direction. This will test the marker localization algorithm for both the frontcam and the sidecam. In this test the algorithm is set to save pictures if the marker is not found in the region of interest. This is done to be able to see what has happened afterwards. Because of this, an extra delay in the processing on the IPC occurs. In Figure 6.24(a) and Figure 6.24(b) the estimated position from the Kalman filter, and the calculated position from the IPC are shown. It is seen that the helicopter position is detected correctly. By inspection of the figures, it is seen that the time width of the steps are different. This is because when making the step, then the IPC runs slower, and does not produce as many samples.



(a) Estimated position state, $\hat{\Xi}$.

(b) Measured position on the IPC, Ξ_{cam}.

Figure 6.24: *The estimated position and the output from the translatory step test.*

It has been tried to estimate the maximum total delay in the state estimation system, when performing a step test. This has been done by plotting the acceleration measurement together with the estimated pixel position of the center of the marker, and the state estimate. Figure 6.25 shows this plot for the x-axis. The IMU and state estimate has been scaled to be equal in magnitude to the pixel position.

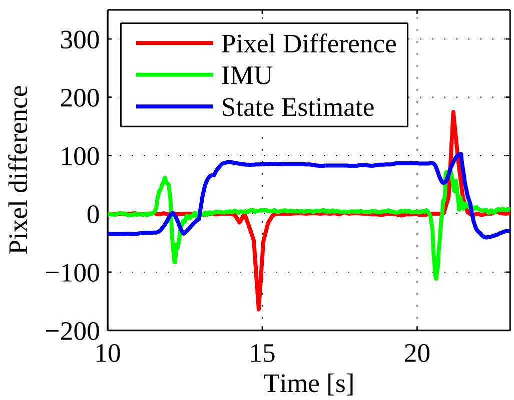


Figure 6.25: *The delay through the state estimation can be seen as the distance between IMU and state estimate. The figure shows the plots for the x-axis and x pixel for the front camera. The total delay before the state estimate reacts has been measured to around* 0,5 s. *It can be seen that the state estimate reacts faster than the pixel position. This is an effect of the reduced sample rate of the IPC.*

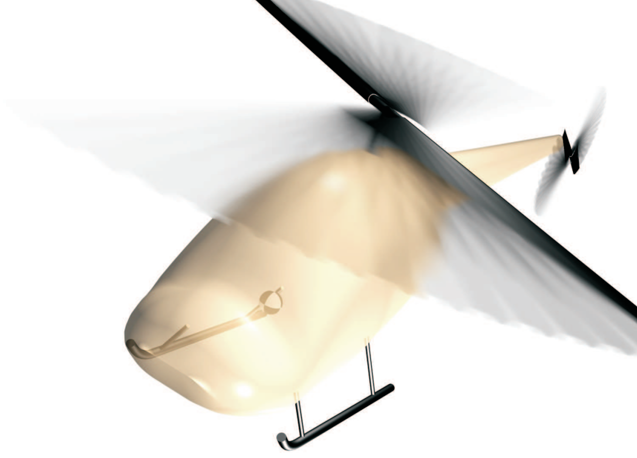
It can be seen that the state estimate reacts faster to the step than the pixel position. This is of cause not the case in reality, but is an effect of the slower sample rate of the IPC, when performing a step. The conclusion is that most of the time inaccuracy is caused by the time axis warp on the IPC. The maximum possible delay is measured between the reaction of the IMU and the state estimate. This has been measured to around 0,5 s, but is also an effect of the IPC saving images to the hard disk.

6.4.4 State Estimation Test Summary

Overall the tests has shown that the state estimation works as desired. However, some minor flaws has been detected. Different sample times on the CC and IPC might cause inaccuracies and noise on the output. But most of these inaccuracies occur when trying to compare the output from the IPC and CC, and not during operation. Further, if the IPC performs a full update, then a large delay in the samples might occur.

6.5 Partial Conclusion

In this chapter it has been described how to obtain a state estimate of the helicopter. It has been chosen only to obtain an estimate of the rigid body states ($\mathbf{\Xi}$, $\dot{\mathbf{\Xi}}$, $\mathbf{\Theta}$ and $\dot{\mathbf{\Theta}}$), since these are the only states needed for state feedback. Image processing software, for locating markers on the helicopter and on the floor, has been designed. An algorithm for obtaining the position and attitude of the helicopter from the marker positions, has been developed. This position and attitude estimate is fused with IMU data in a Kalman filter, which outputs the estimated state. A test of the combined sensor fusion and state estimation has shown, that it is possible to obtain a correct state estimate.



7

CONTROL

Chapter Contents

In this chapter the control design is documented. It is initiated with some general considerations regarding controllers for UAV's. This leads to the design of a PID controller, which is developed on the basis of a simplified linear model of the helicopter. The PID controller is tested on the non-linear model in a simulation, before it is applied to the real helicopter. The chapter is concluded with a brief analysis of more advanced controller types used for UAV control in other research projects. This is done, to give an overview over what type of control could be utilized later in the HAMOC project.

7.1 Basic Control Structure

The primary goal for controlling the helicopter in this project, is to stabilize the helicopter in hover, which is an unstable equilibrium point. This may be done in a number of ways, using different controller types and design methods.
Usually a controller for a UAV utilizes an inner loop for controlling the attitude, and an outer loop controlling the translation by giving attitude input [Johnson and Kannan, 2002]. But in more advanced controllers as for example non-linear or model based controllers, the inner and outer loops are not separated. Essentially a controller needs to control the position of the helicopter. If only using the position as direct feedback to the controller, it could potentially cause very large errors, if a new far away position reference is given, or if switching to autonomous when far away from the reference. Adding to this, an integral controller on the position could cause serious wind-up errors. The consequence may be an unstable system. To be able to determine how to get from one position to another, a trajectory generator is usually made. This generates an optimal trajectory of flight given the initial position and the end position (and perhaps some waypoints in between.) Preferably the trajectory is generated online by use of the information about the current position. This is called adaptive adaptive trajectory generation (see Johnson and Kannan [2005]). As a guide to make a trajectory generator, the trajectories human pilots tends to implement, can be followed. Roll and pitch angular rate trajectories made by human pilots can be well approximated by piecewise linear and constant segments [Gavrilets et al., 2002].
The structure of the above described overall control structure is illustrated in Figure 7.1.

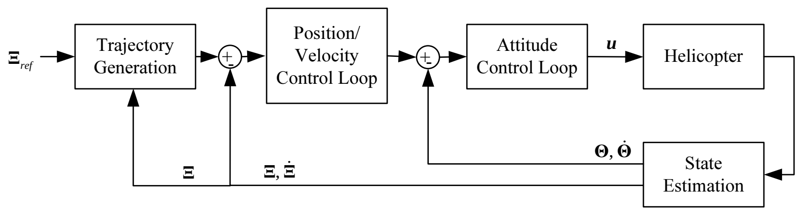


Figure 7.1: *The basic structure of an inner and an outer control loop often used to control UAV's. Further a trajectory generator is added to make sure the control input does not produce large errors.*

7.1.1 The *PID* Controller Structure

Basic *PID* controllers, controlling a helicopter in an inner and an outer loop as described in the previous section, has been mentioned in papers but no literature really provides a thorough description. The reason is believed to be that this type of control does not really perform very accurately or robust, and hence is not interesting for control research. However, since the controller is simple to implement (though difficult to tune), it can make a helicopter hover and will therefore be used to test if the system and sensor estimation functions properly, before more advanced controllers are considered. The following control considerations is therefore not based on other research literature.
The basic structure used for controlling the helicopter using *PID* controllers, is shown in Figure 7.2 on the facing page.

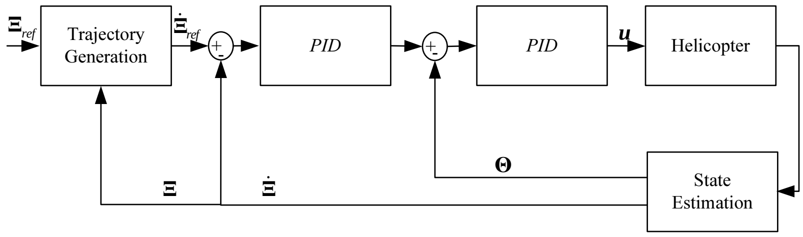


Figure 7.2: The overall structure of a double loop PID controller. The inner controls the attitude, and the outer controlling the position or velocity.

As seen, the structure resembles that of the simple double loop control in Figure 7.1. However, since the *PID* controller is a one dimensional controller (i.e. a SISO system from the error to the control signal), Figure 7.2 is not entirely accurate, since the system has four control input (S_{mr}, S_{lat}, S_{lon} and S_{tr}) and the state estimation has a 12 dimensional output vector (see Section 6.3). The system can though be simplified by considering each input as only affecting one state and neglecting all cross couplings, e.g. S_{lon} is considered only to affect the rotation around the x-axis, which again affects the motion along the y-axis. This means that the plant, i.e. the helicopter, can be seen as multiple SISO systems. The SISO systems are shown in Figure 7.3. This means that for each of the four states $\dot{z}$, ϕ, θ, ψ, that is directly controlled by one of the four input, a controller can be developed separately.

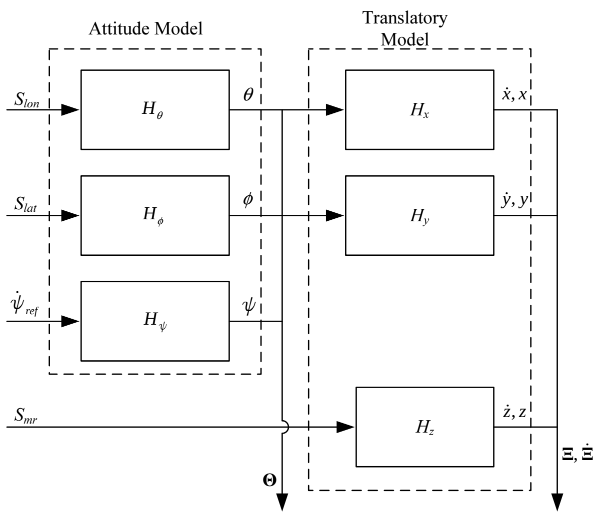


Figure 7.3: Neglecting cross coupling in the system, the transfer function from each control input can be treated seperately.

Thus, the inner loop consists of three *PID* controllers, controlling the attitude of the helicopter. Now, if it is wanted to control the velocity of the helicopter, it can be done by adding controllers for the three remaining degrees of freedom ($\dot{x}$, $\dot{y}$ and $\dot{z}$). These controllers take a velocity reference from the trajectory generator, and produce a desired attitude reference for the inner control loop. The final structure is shown in Figure 7.4, where it is chosen to keep a constant yaw angle, and only navigate horizontally by pitching and rolling the helicopter.

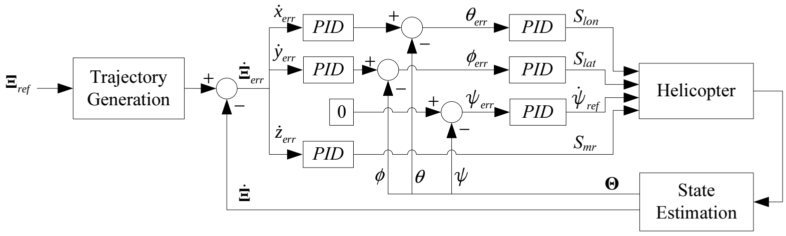


Figure 7.4: *The overall structure using PID controllers and a trajectory generator to control the helicopter.*

Note that the x-axis velocity controller controls rotation around the y-axis and vice versa. This control configuration has been tested briefly on the model of the Bergen helicopter from Hald et al. [2006] including sensor estimation. Though only using handtuned *P* controllers, this test has verified that it is possible to stabilize the helicopter model, given small velocity references (less than $10\,\mathrm{m/s}$). By designing a dedicated *PID* controller for the Corona helicopter, it should be possible to stabilize this as well, and then verify that the systems works, before more advanced controllers are considered.

7.2 *PID* Controller Development

The procedure for developing a *PID* controller for the helicopter is as follows. First, the non-linear model of the helicopter is linearized in hover. The linear model is now used as basis for the design of *PID*-controllers for all control input. When the controller performs satisfactory for the linear model in hover, it is applied to the non-linear model in a hover simulation. Now the controller is handtuned for optimizing the perfomance in the simulation. When this is done, it is applied to the real helicopter. For safety reasons, the controller is applied stepwise for each control input in the following manner:

- First, the altitude (z-axis) controller is applied, and the three remaining control signals are controlled manually by the pilot.
- Next, the lateral (y-axis) controller is applied, and the longitudinal (x-axis) and the yaw motion is controlled manually by the pilot.
- Next, the longitudinal controller is applied, and the yaw motion is controlled manually by the pilot.
- Finally, the yaw controller is applied.

When the controller is applied fully to the real helicopter, it might be necessary to do some handtuning of the controller again, in order to increase the performance.

7.2.1 Linearization of Model

The non-linear helicopter model is a complex set of differential equations with a lot of cross couplings. If this model is linearized, it becomes an advanced linear model, and it would require considerable amounts of time to make a controller for it. Therefore, a much simpler approach is taken, as a new linear model is developed from scratch using some rough assumptions about how a helicopter is behaving in a hover situation. All cross couplings will be omitted in this simplified model. This method will not yield the correct linearized model, but it is assumed that the simplified model will be good enough for designing a PID controller. The linearized model is derived in Appendix F on page 195, where it is also compared to the developed non-linear model presented in Chapter 5. It is concluded that the model is feasible for developing a controller for hover. Figure 7.5 through 7.8 presents the linearized model.

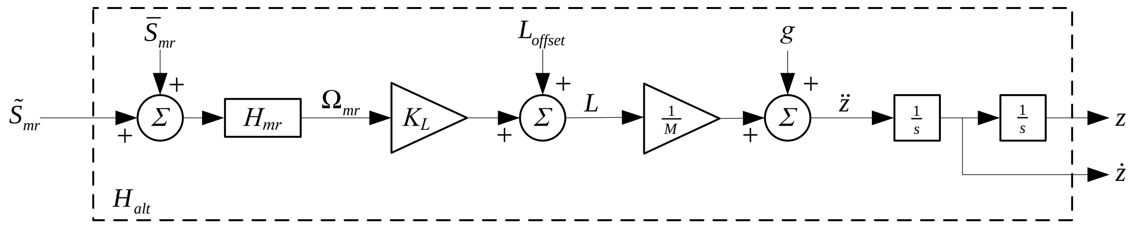


Figure 7.5: *Block diagram of the altitude model, H_{alt}, which shows how it has been implemented in* SIMULINK.

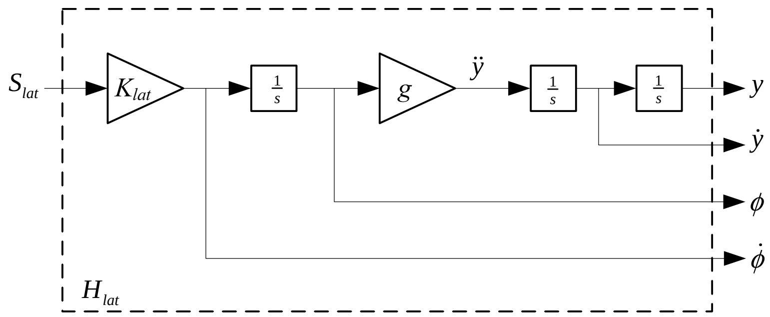


Figure 7.6: *Block diagram of the lateral model, H_{lat}, which shows how it has been implemented in* SIMULINK.

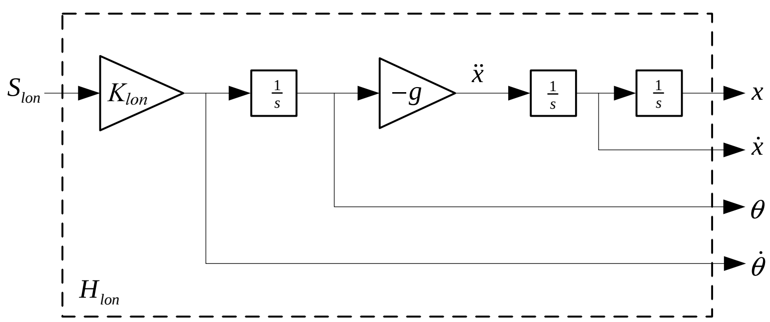


Figure 7.7: *Block diagram of the longitudinal model, H_{lon}, which shows how it has been implemented in* SIMULINK.

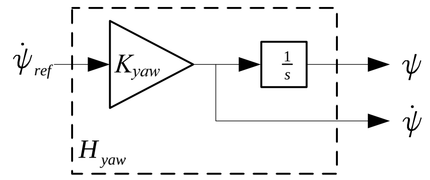


Figure 7.8: *Block diagram of the yaw model, H_{yaw}, which shows how it has been implemented in* Simulink.

7.2.2 Controller Design

In the following sections, *P*, *PI*, or *PID* controllers will be developed for each linear model. Figure 7.9 shows a block diagram of the *PID* controller, which in general can be expressed as

$$C(s) = P + \frac{I}{s} + Ds \tag{7.1}$$

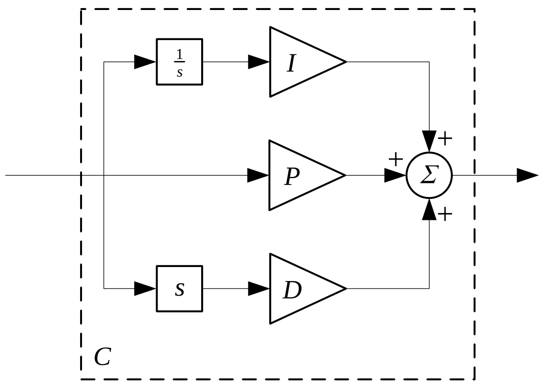


Figure 7.9: *A general block diagram of the PID controller.*

Ziegler-Nichols "ultimate sensitivity method" [Franklin et al., 2002, p. 221] will be used to find the *P*, *I* and *D* gains in each case. The procedure is to set *I* and *D* to 0, and then increase *P* until the output of the closed loop system is oscillating with constant amplitude. This happens when a pole of the closed loop system is placed on the imaginary axis of the complex plane. This gain is called the ultimate gain, K_u, and the period of the oscillations is called P_u. Having these values, the *PID* gains are tuned according to Table 7.1.

Control Type	*P*	*PI*	*PID*
P	$0{,}5K_u$	$0{,}45K_u$	$0{,}6K_u$
I	–	$0{,}54{}^{K_u}/_{P_u}$	$1{,}2{}^{K_u}/_{P_u}$
D	–	–	$0{,}075K_uP_u$

Table 7.1: *The Ziegler-Nichols tuning scheme for a PID controller.*

In practice, K_u and P_u are found by implementing the closed loop system in a Matlab m-file, which is plotting the step response of the system. K_u is tuned such that the step response oscillation amplitude decays very slowly. The value of K_u is estimated with three significant digits. P_u is simply found by inspection of the plotted step response.

Altitude Controller

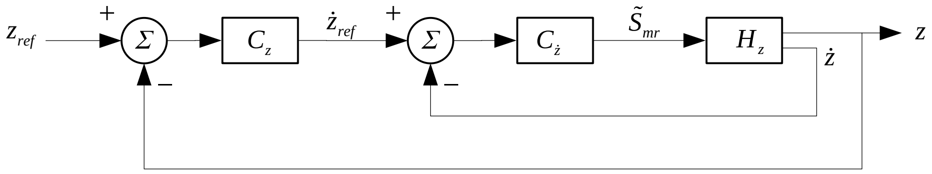


Figure 7.10: *A block diagram of the altitude controller.*

Figure 7.10 shows the structure of the altitude controller. An inner loop is controlling the z velocity, and an outer loop is controlling the altitude of the helicopter. The inner controller, $C_{\dot{z}}$, is a *PID* controller, and the ultimate gain and period is found to

$$K_{u,\dot{z}} = -80{,}6\,\mathrm{rad/m} \qquad P_{u,\dot{z}} = 1{,}20\,\mathrm{s} \tag{7.2}$$

Thus, the *PID* gains are

$$\begin{aligned} P_{\dot{z}} &= -48{,}4\,\mathrm{rad/m} \\ I_{\dot{z}} &= -80{,}6\,\mathrm{rad/m{\cdot}s} \\ D_{\dot{z}} &= -7{,}25\,\mathrm{rad{\cdot}s/m} \end{aligned} \tag{7.3}$$

The resulting step response of the inner loop is shown in Figure 7.11.

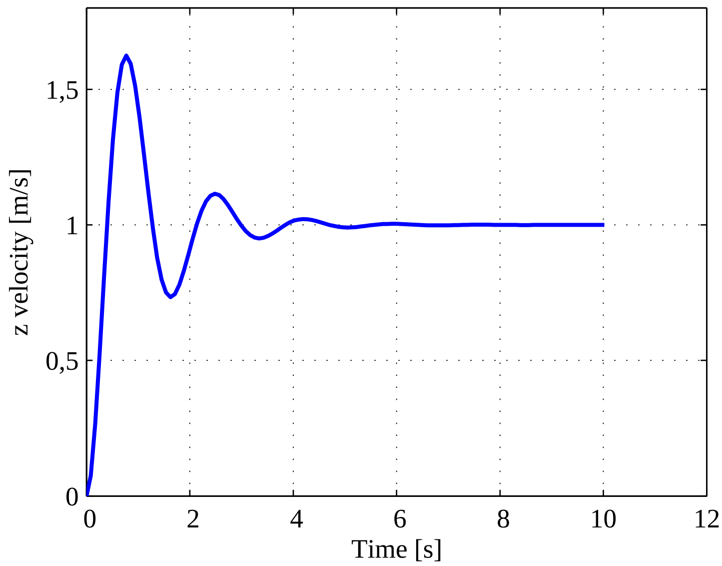


Figure 7.11: *The step response of the inner loop of Figure 7.10.*

The procedure is repeated for the outer loop, with the inner loop implemented. This is also a *PID* controller.

$$K_{u,z} = 1{,}71\,\mathrm{s}^{-1} \qquad \mathrm{P_{u,z}} = 1{,}56\,\mathrm{s} \tag{7.4}$$

$$
\begin{aligned}
P_z &= 1{,}03\,\mathrm{s}^{-1} \\
I_z &= 1{,}32\,\mathrm{s}^{-2} \\
D_z &= 0{,}20
\end{aligned}
\tag{7.5}
$$

The resulting step response of the outer loop is shown in Figure 7.12.

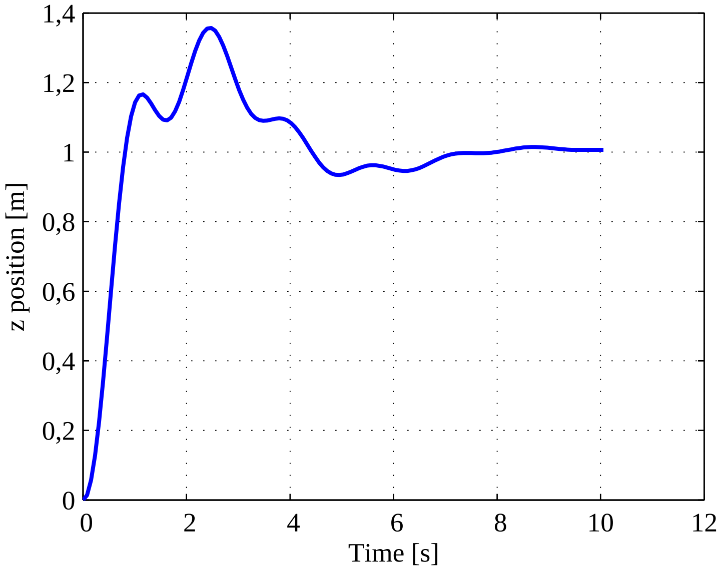


Figure 7.12: *The step response of the outer loop of Figure 7.10.*

Lateral and Longitudinal Controller

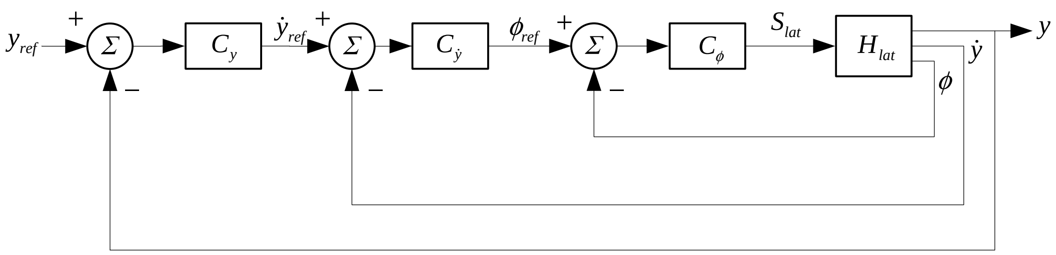


Figure 7.13: *A block diagram of the lateral controller.*

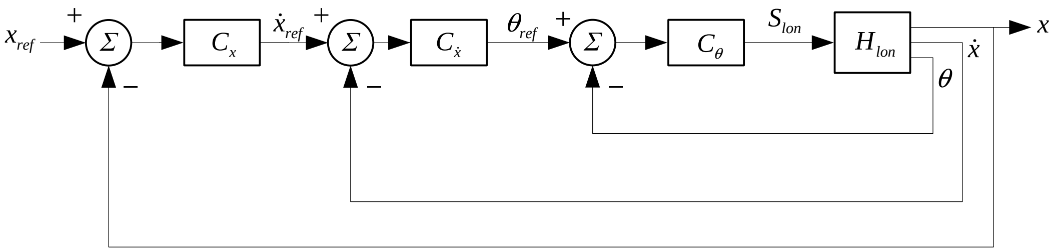


Figure 7.14: *A block diagram of the longitudinal controller.*

Figure 7.13 and Figure 7.14 shows the structure of the lateral and longitudinal controller, respectively. They consist of three loops, an inner loop controlling the cyclic pitch (ϕ and θ), a loop controlling the translatory velocity ($\dot{y}$ and $\dot{x}$), and an outer loop controlling the position (y and x).
The ultimate sensitivity method is not applicable for the inner loop, because it is a stable first order system with a single pole in $-P \cdot K$, and therefore no oscillations will occur when increasing the gain. The closed loop transfer functions are given by

$$H_{\phi,cl} = \frac{P_\phi K_{lat}}{s + P_\phi K_{lat}} \tag{7.6}$$

and

$$H_{\theta,cl} = \frac{P_\theta K_{lon}}{s + P_\theta K_{lon}} \tag{7.7}$$

The performance of this type of system gets better with increasing gain, so in theory P_ϕ and P_θ should be infinite. This is not feasible when considering the non-linear system, since this is a higher order system. Large values of P gains might lead to instability [Franklin et al., 2002, p. 216]. Therefore, P_ϕ and P_θ is set to $-1\,\mathrm{rad}^{-1}$, which yields a step response rise time of 1,01 s. The I and D term will be omitted in this controller, since it is not essential, that the attitude is controlled absolutely correct.
The next loop controlling the velocity is a stable second order system. The root locus of the closed loop transfer function with P gain as the parameter is shown in Figure 7.15. It shows that the poles never cross the imaginary axis, i.e. the ultimate sensitivity method does not work for this system either. Instead different values of $P_{\dot{y}}$ is tried, and a value which gives a slower rise time of the step response, than that of the inner loop, is chosen. The result is

$$P_{\dot{y}} = 0{,}05\,\mathrm{rad{\cdot}s/m} \tag{7.8}$$
$$P_{\dot{x}} = -0{,}05\,\mathrm{rad{\cdot}s/m} \tag{7.9}$$

which yields a rise time of 3,44 s. Again, the I and D gain of this loop will be omitted.
The outer loop gains can be found using the ultimate sensitivity method.

$$K_{u,y} = K_{u,x} = 2{,}00\,\mathrm{s}^{-1} \qquad \mathrm{P_{u,y}} = \mathrm{P_{u,x}} = 6{,}37\,\mathrm{s} \tag{7.10}$$

$$\begin{aligned} P_y = \; P_x &= \; 1{,}20\,\mathrm{s}^{-1} \\ I_y = \; I_x &= \; 0{,}38\,\mathrm{s}^{-2} \\ D_y = \; D_x &= \; 0{,}96 \end{aligned} \tag{7.11}$$

The resulting step response of the outer loop is shown in Figure 7.16.

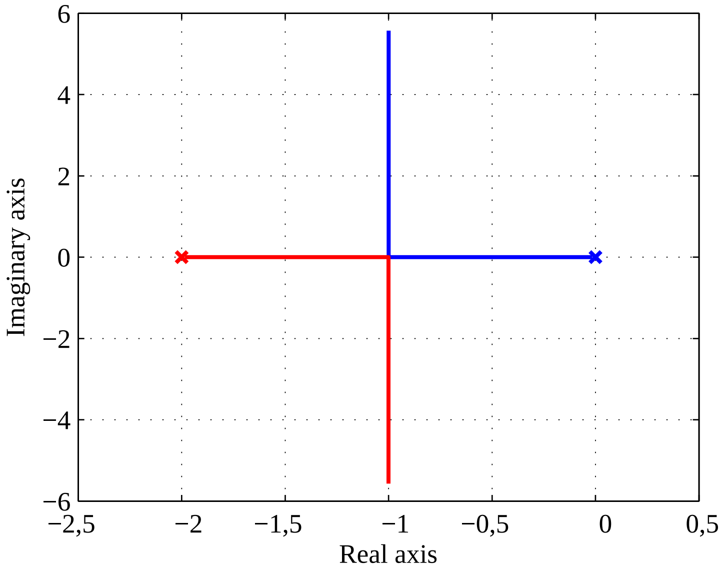


Figure 7.15: *The root locus of the $\dot{y}$ closed loop system.*

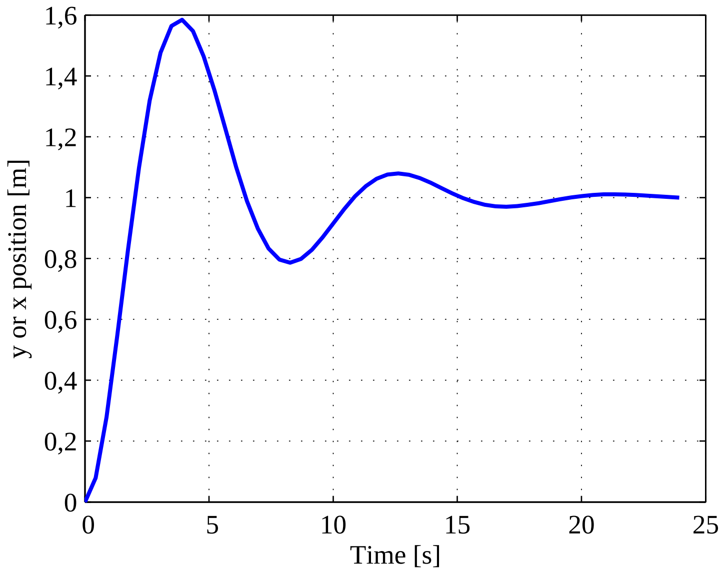


Figure 7.16: *The step response of the outer loop of Figure 7.13.*

Yaw Controller

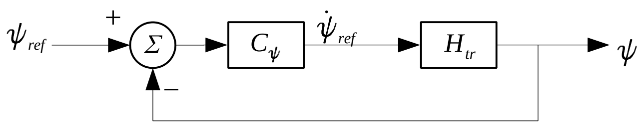


Figure 7.17: *A block diagram of the yaw controller.*

From the comparison of the linear and the non-linear model (performed in Appendix F), it is seen from Figure F.7(e) and Figure F.7(f), that the linear yaw model is very close at being equal to the non-linear yaw model. The reason for this is the gyro, which works as an inner control loop, stabilizing $\dot{\psi}$. The inner loop is shown in Figure 5.31 on page 77. The yaw controller, which is shown in Figure 7.17, is an outer loop with a *P* gain. As the system itself is an integration, there will be no steady state error. With a gain of

$$P_{\psi} = 2\,\mathrm{s}^{-1} \tag{7.12}$$

the step response will have a rise time of approximately 1,1 s. The step response is illustrated in Figure 7.18.

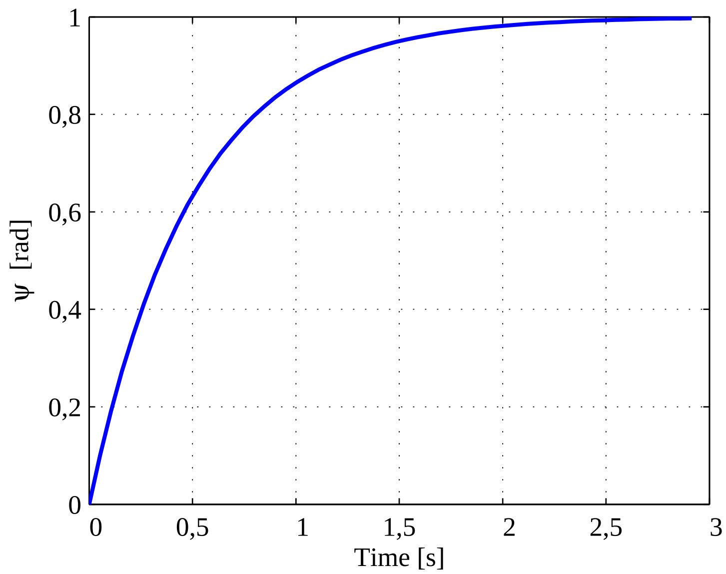


Figure 7.18: *The step response of the yaw controller.*

To summarize, all the obtained controller gains are listed in Table 7.2.

Control Variable	$P_{(\cdot)}$	$I_{(\cdot)}$	$D_{(\cdot)}$
$\dot{z}$	−48,4	−80,6	−7,25
z	1,03	1,32	0,20
ϕ	−1	0	0
$\dot{y}$	0,05	0	0
y	1,20	0,38	0,96
θ	−1	0	0
$\dot{x}$	−0,05	0	0
x	1,20	0,38	0,96
ψ	2	0	0

Table 7.2: *Table of all controller for the linear model. Units of the gains are omitted.*

7.2.3 Test of Controller on Nonlinear Model

The *PID* controllers have been implemented in SIMULINK with the gains found in the previous section. In this section it is applied to the non-linear model, also implemented

in SIMULINK, and a step test is performed for all four input. During the test, three signals are logged: the reference input (z_{ref}, y_{ref}, x_{ref} and ψ_{ref}), the output of the controller (S_{mr}, S_{lat}, S_{lon} and S_{tr}), and the state vector ($\boldsymbol{x}$). Saturation limits are applied to the controller output, such that they correspond to the real world input limits, $S_{mr,min} = 0\,\mathrm{rad/s}$, $S_{mr,max} = 190\,\mathrm{rad/s}$, $S_{lat,min} = -1$, $S_{lat,max} = 1$, $S_{lon,min} = -1$, $S_{lon,max} = 1$, $\dot{\psi}_{ref,min} = -1{,}5\,\mathrm{rad/s}$ and $\dot{\psi}_{ref,max} = 1{,}5\,\mathrm{rad/s}$.
The results of the step tests are given in Figures 7.19 through 7.22. Each figure has four subfigures, (a), (b), (c) and (d), containing the following information:

- (a) is showing the reference input step and the step response of the non-linear model in the considered direction. To be able to compare it with the step response of the linear model, this has been superimposed.

- (b) is showing the step response of the other five position and attitude states. Due to cross couplings, it is necessary to include these in the tuning considerations of the controller gains.

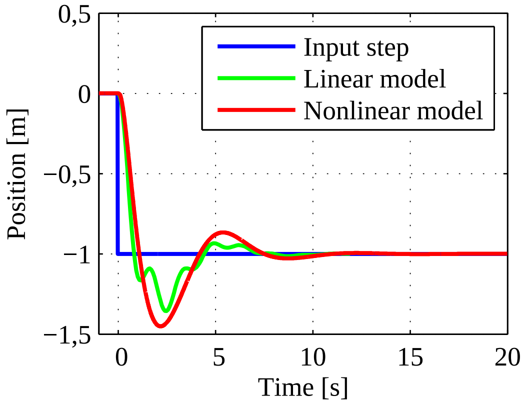
- (c) is showing the controller output for the main rotor.

- (d) is showing the controller output for the cyclic pitch and the tail rotor pitch.

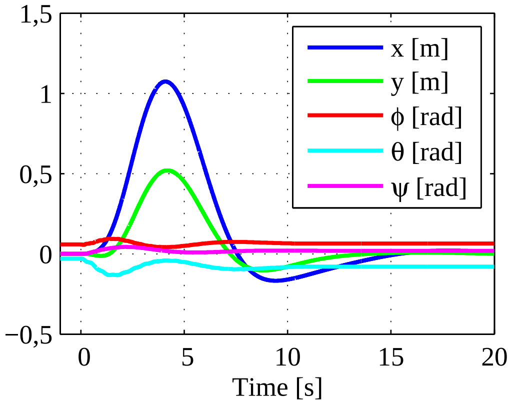
To be able to compare the individual figures, they all have the same scaling on the time axis.

The z Position Step Test

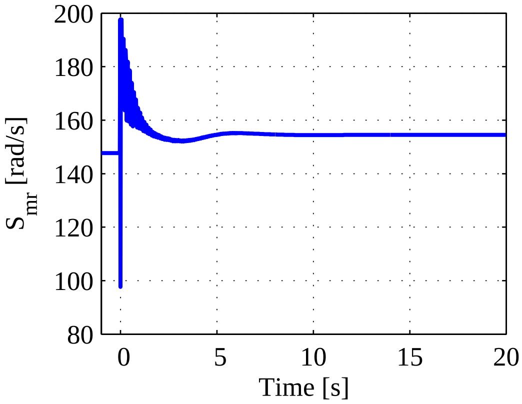
The step test in the z axis direction is applied as a negative step from 0 m to −1 m, to make the helicopter increase its altitude in the simulation. The non-linear response in the z direction is more smooth than the linear response, however, the step input leads to cross coupling effects in both the lateral and longitudinal direction. This is mainly due to the wire attached to the nose of the helicopter, which is included in the non-linear model, but not in the linear model. Looking at the control signal for the main rotor, we see that it reacts aggressively to the step. Zooming in on the time axis just after the step shows that the controller first increases S_{mr} $+50\,\mathrm{rad/s}$, in the next iteration it drops $-100\,\mathrm{rad/s}$, and then it increases $+100\,\mathrm{rad/s}$ again, from where it starts decreasing to its steady state value. This kind of behaviour is not wanted when testing on the real helicopter. Therefore the gains are decreased to avoid the aggressive behaviour.



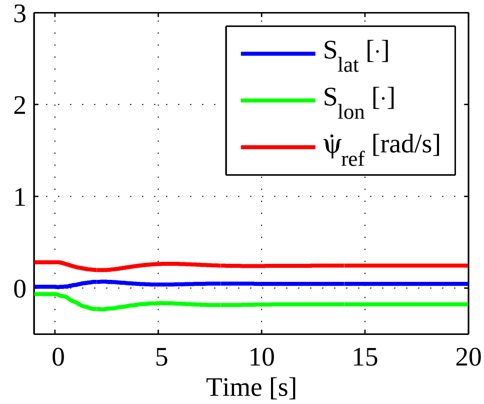
(a) The step response of the non-linear and linear model superimposed.



(b) Cross couplings effect from the z step test.



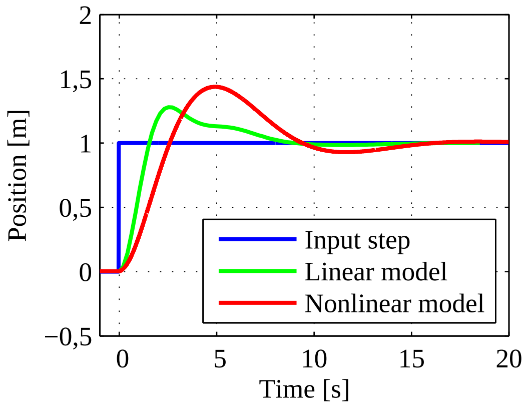
(c) Controller output S_{mr}.



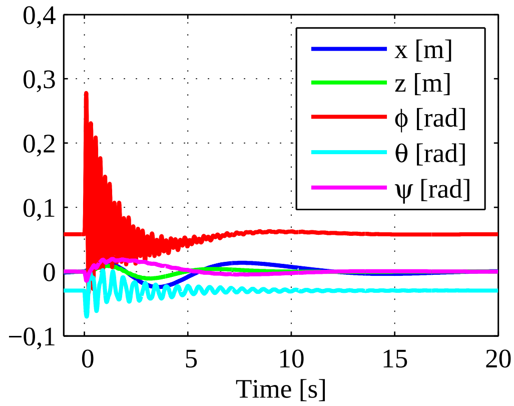
(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.19: *The z position step test.*

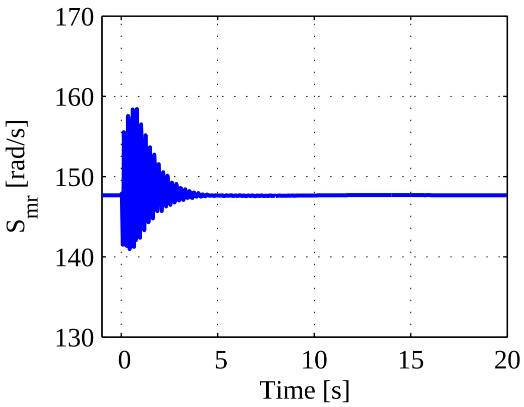
The y Position Step Test



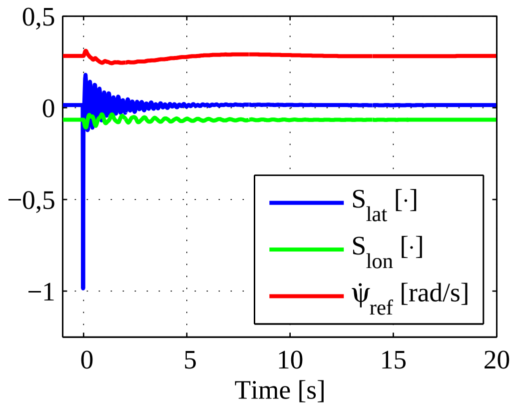
(a) The step response of the non-linear and linear model superimposed.



(b) Cross couplings effect from the y step test.



(c) Controller output S_{mr}.



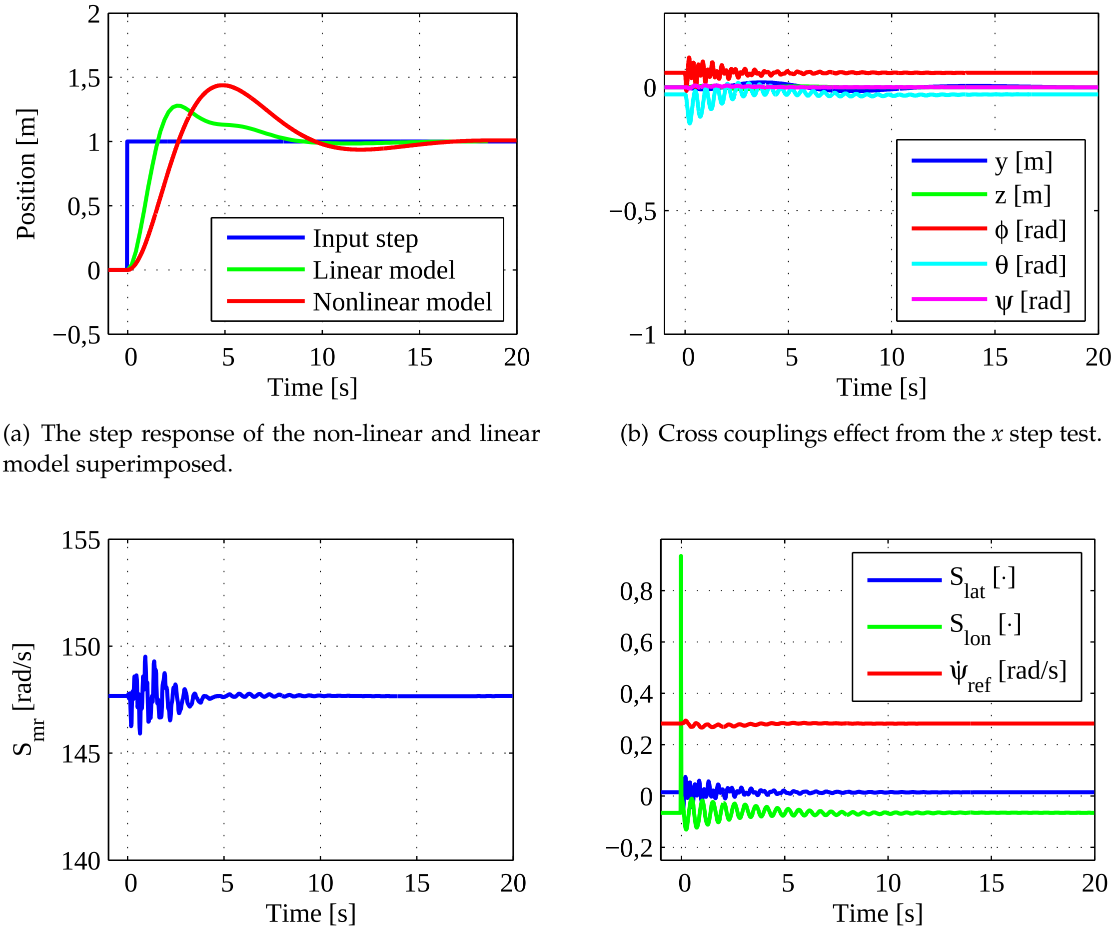
(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.20: *The y position step test.*

The lateral step response follows the reference, though the cross coupling effects include oscillations of both the roll and pitch of the helicopter. The cause can be found in the lateral controller output. The step input to the lateral pitch servo leads to heavy flapping of the main rotor blades, which is not included in the linear model. This again causes the helicopter to oscillate about the two horizontal axes. As with the z position controller, the oscillatory behaviour can be reduced by decreasing the gains to obtain a less aggressive controller.

The x Position Step Test

The considerations done for the y position step test also applies to the step test in the longitudinal direction.



(a) The step response of the non-linear and linear model superimposed.

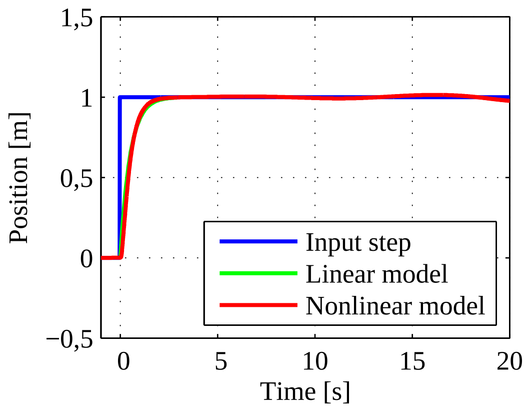
(b) Cross couplings effect from the x step test.

(c) Controller output S_{mr}.

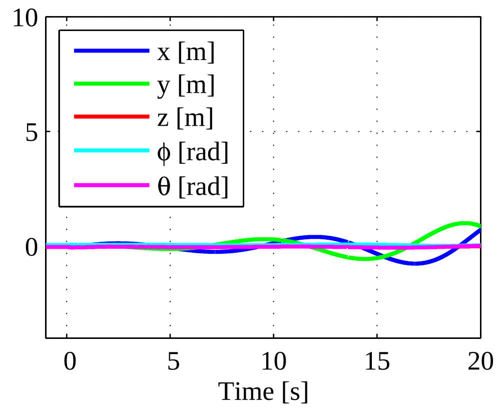
(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.21: *The x position step test.*

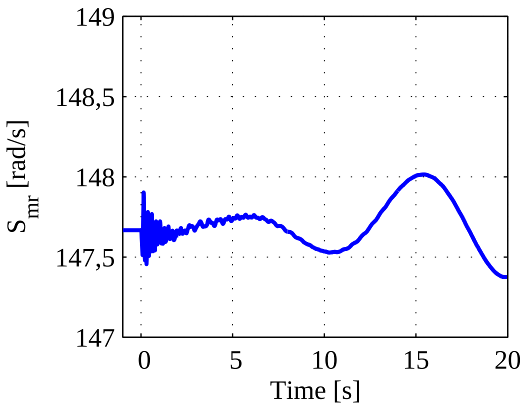
The ψ Rotation Step Test



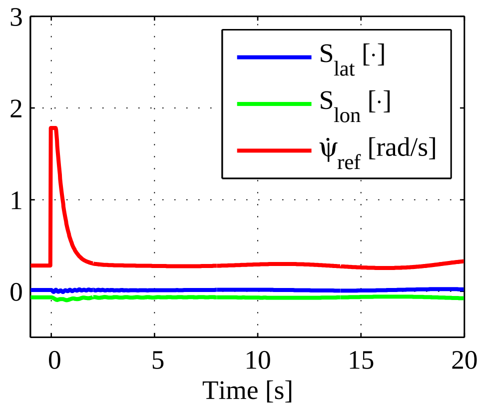
(a) The step response of the non-linear and linear model superimposed.



(b) Cross couplings effect from the ψ step test.



(c) Controller output S_{mr}.



(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.22: *The ψ rotation step test.*

The yaw angle step test results in a nicely shaped response with no overshoot and a rise time of approximately 1 s. After the step, the system starts to oscillate, and eventually becomes unstable in the x and y direction. This is explained by the fact that the helicopter has moved, such that $\{B\}$ and $\{E\}$ are not coinciding. The controller is not designed to handle this situation, it always tries to minimize errors in $\{B\}$ - if for example the helicopter is hanging a little to the right of the origin of $\{E\}$ (positive y), the controller reacts by making the helicopter fly to the left relative to $\{B\}$, which can be completely wrong (if e.g. $\psi = \frac{\pi}{2}$ rad this is positive ${}^{E}x$). In the simulation, the helicopter starts spiralling outwards when $\{B\}$ and $\{E\}$ are not aligned, and eventually it crashes. This should though not be a problem, since it is wanted to make the helicopter hover with $\psi = 0$ rad, and the problem will therefore not occur.

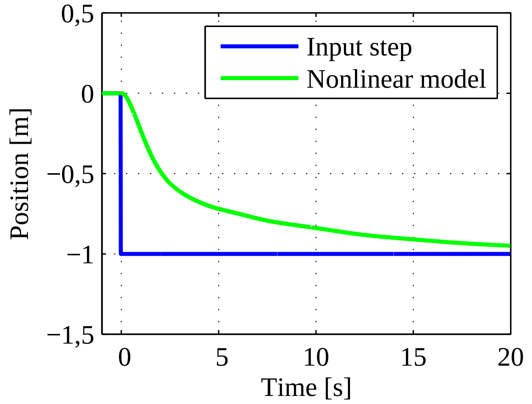
Tuning of Controller Gains

When tuning the controller it is not only a question of optimizing the performance of the non-linear model. It is also important to have the real helicopter in mind. This means that the controller must be tuned such that it does not react to aggressively. We do not want to see the controller output for the main rotor going from 110 rad/s to 300 rad/s in one time sample, even though this might work fine in the simulation. Therefore, the primary part of tuning is to make the control output look like a human pilots control signals - or at least keep them within a certain dynamic range. Optimizing settling times and overshoots is not important at this stage of the project.
The tuning procedure is not documented in detail, since it is mostly a question of common sense combined with trial and error. The general idea is to reduce the P and I terms while the D terms are omitted, as these makes the controller sensitive to noise on the state estimate. As an example, $P_{\dot{z}}$ is reduced by a factor 10, and omitted both $I_{\dot{z}}$ and $D_{\dot{z}}$. With these new gain values, the outer loop of the z position controller is tuned using the ultimate sensitivity method again. Then I_z is reduced by a factor 2, and D_z is set to zero. The final result of the tuning is given in Table 7.3.

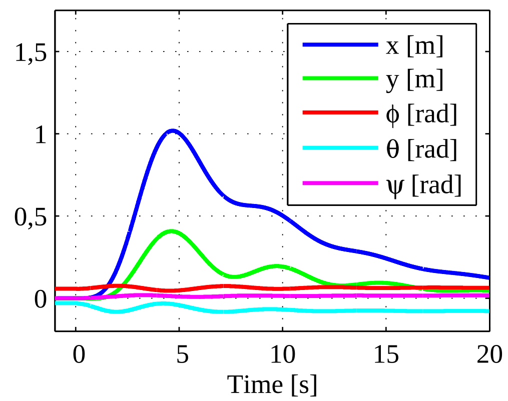
Control Variable	$P_{(\cdot)}$	$I_{(\cdot)}$	$D_{(\cdot)}$
$\dot{z}$	$-4{,}84$	0	0
z	1,41	0,28	0
ϕ	$-0{,}5$	0	0
$\dot{y}$	0,1	0	0
y	1,08	0,19	0
θ	$-0{,}5$	0	0
$\dot{x}$	$-0{,}1$	0	0
x	1,08	0,19	0
ψ	2	0	0

Table 7.3: *Table of all controller gains for the non-linear model. Units of the gains are omitted.*

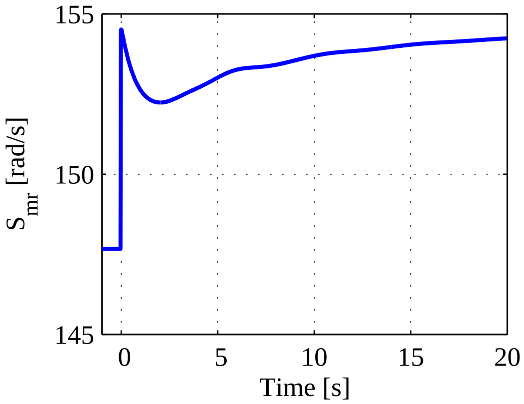
With these new controller gains, a series of step tests are performed again, in the same manner as described in Section 7.2.3. The result is shown in Figures 7.23 through 7.26, with the same configuration as in Figure 7.19 to Figure 7.22. The general conclusion is that the controller output has been reduced substantially in amplitude, and looks much more smooth now, than it did before the tuning. The cost for this "softening" of the controller is increased rise and settling times.



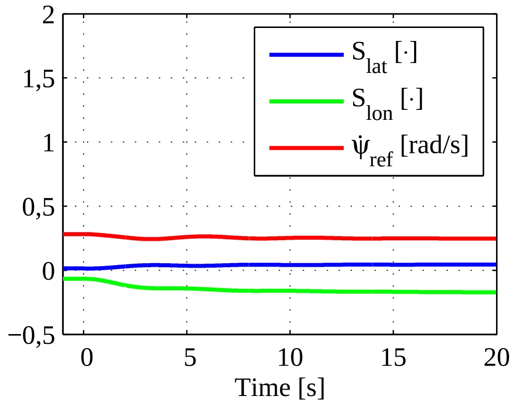
(a) The step response of the non-linear model with tuned controller gains.



(b) Cross couplings effect from the z step test.

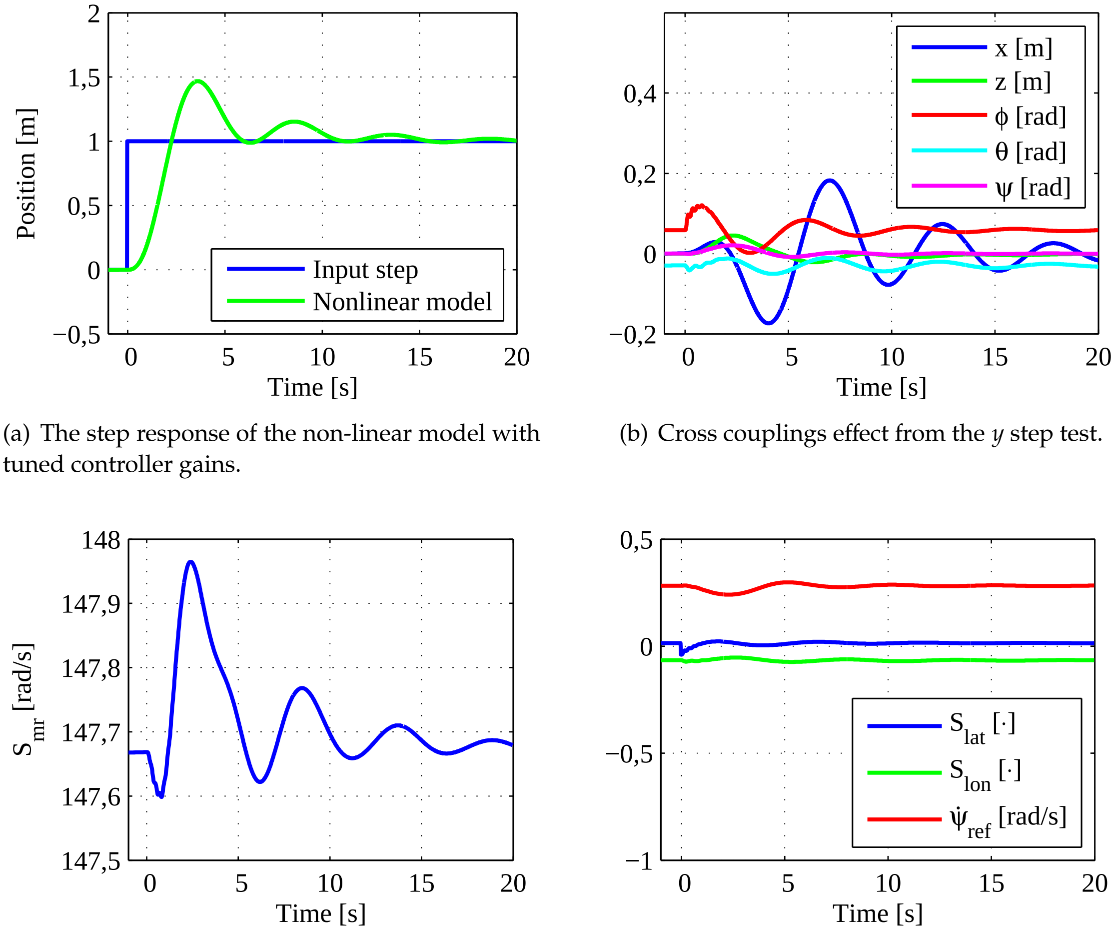


(c) Controller output S_{mr}.



(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.23: *The z position step test repeated with the tuned controller.*



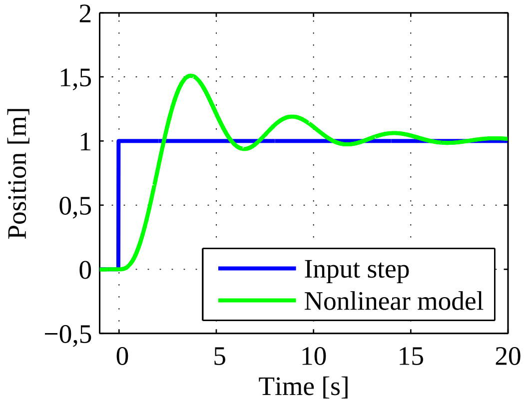
(a) The step response of the non-linear model with tuned controller gains.

(b) Cross couplings effect from the y step test.

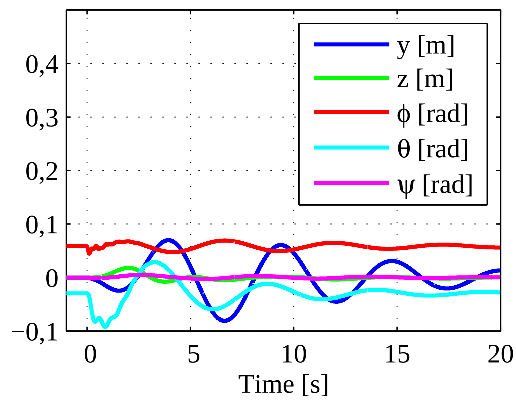
(c) Controller output S_{mr}.

(d) Controller output S_{lat}, S_{lon} and S_{tr}.

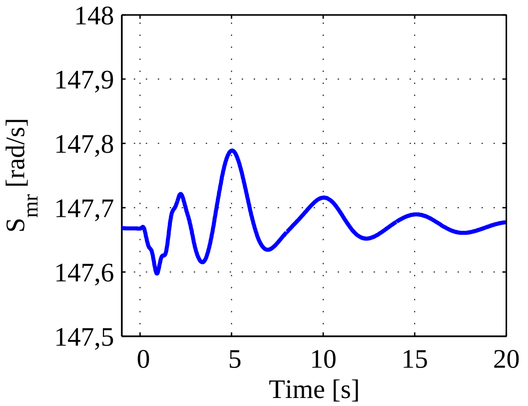
Figure 7.24: *The y position step test repeated with the tuned controller.*



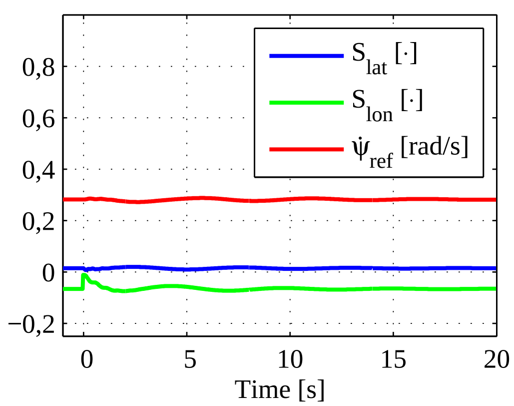
(a) The step response of the non-linear model with tuned controller gains.



(b) Cross couplings effect from the x step test.

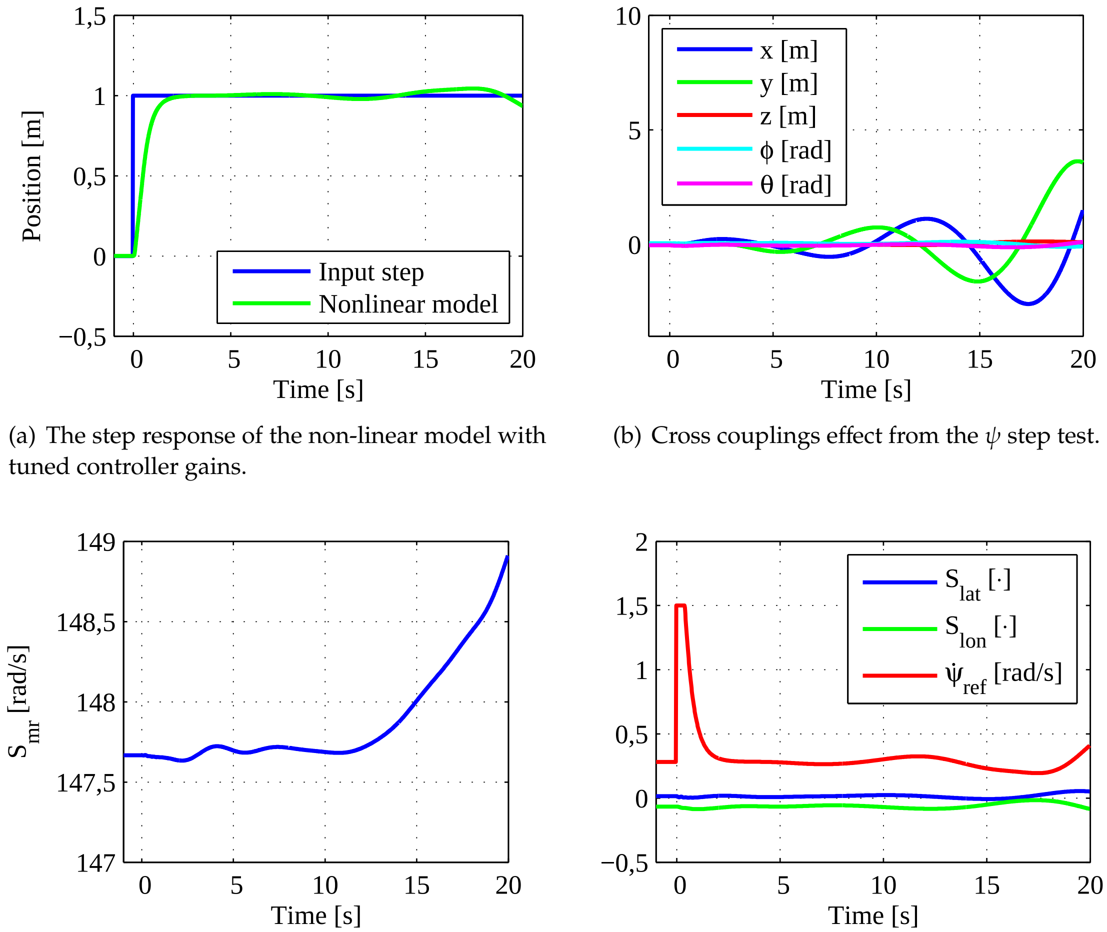


(c) Controller output S_{mr}.



(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.25: *The x position step test repeated with the tuned controller.*



(a) The step response of the non-linear model with tuned controller gains.

(b) Cross couplings effect from the ψ step test.

(c) Controller output S_{mr}.

(d) Controller output S_{lat}, S_{lon} and S_{tr}.

Figure 7.26: *The ψ rotation step test repeated with the tuned controller.*

7.2.4 Test of Controller on Real Helicopter

As described in the beginning of this section (page 132), the idea is to apply the controller on the helicopter stepwise for the four input signals z, y, x and ψ, in that order. In practice, this is done by removing the signal wires of the manual controlled servos from the servoboard output, and connecting them directly to the receiver. In that way, the controller is disconnected in the selected directions, and the servos can be controlled manually regardless of the select switch on the RC.

Before applying a controller, a manual flight is performed and the control input measured. This is done to be able to determine the approximate working point offset for the controllers.

At the time of writing, a functioning altitude controller has been obtained, but has not been possible to tune it very well. Furthermore preliminary tests of the lateral controller have been done. The reason for this status is hardware problems, which is described later. In the following, the logged data of these test flights will be presented. However, to reach the goal of autonomous hovering, more thorough tests will have to be done in all four directions.

The Altitude Controller

The altitude controller was as a preliminary feature implemented with a slider gain as a substitute for $P_{\dot{z}}$, $I_{\dot{z}}$ and P_z. Thus, it was possible to do some tuning of the gains during the autonomous flight. Table 7.4 shows the actual gain values, as they were throughout the considered test flight.

Control Variable	$P_{(\cdot)}$	$I_{(\cdot)}$	$D_{(\cdot)}$
$\dot{z}$	−24,3	0	0
z	0,2	0	0

Table 7.4: *Table of the altitude controller gains used for the first tests of the real helicopter.*

Figure 7.27 shows the z reference and the estimated z position during the whole flight from take off to landing. In between the two vertical dashed lines the helicopter was in autonomous mode, and had the altitude controller enabled. The bottom graph shows the controller output.
At first sight, it does not look like the position follows the reference very much. However, there is some correlation between the graphs. First it should be noted that it is a very difficult environment control the helicopter in (which is also verified by the pilot). The reason is that the helicopter presses a lot of air downwards, and as the test area is quite small (approximately $2{,}5 \times 3 \times 3\,\mathrm{m}^3$), the air has to move upwards along the edges of the test area. When it reaches the ceiling, it must return to the center of the area, thus pressing the helicopter downwards. But if moving, rolling or pitching the helicopter just a bit, this airflow can be completely changed, and this will also change the thrust. This is why oscillations is seen on the graph. When the controller is enabled, the altitude immediately starts to increase. It is expected that this is due to the main rotor velocity working point not being tuned correctly. After a few seconds the altitude is stabilized around 0,4 m. At around 50 s a step is given, and it is seen that the main rotor velocity increases at the same time. This affects the ${}^{E}z$ position as it should, though only to around −0,4 m. Later, at around 100 s a step is performed again, and again the tendency of ${}^{E}z$ following the reference, is seen. A big altitude drop is seen at time 72 s. This is caused by cross coupling effects. In Figure 7.28 we have zoomed in on the first reference step, and superimposed the estimated x position of the helicopter. Just before the helicopter drops in altitude, it has been moving forward. The longitudinal motion is caused by a pitch, and when pitching the helicopter the air column on which the helicopter "floats", is blown away.

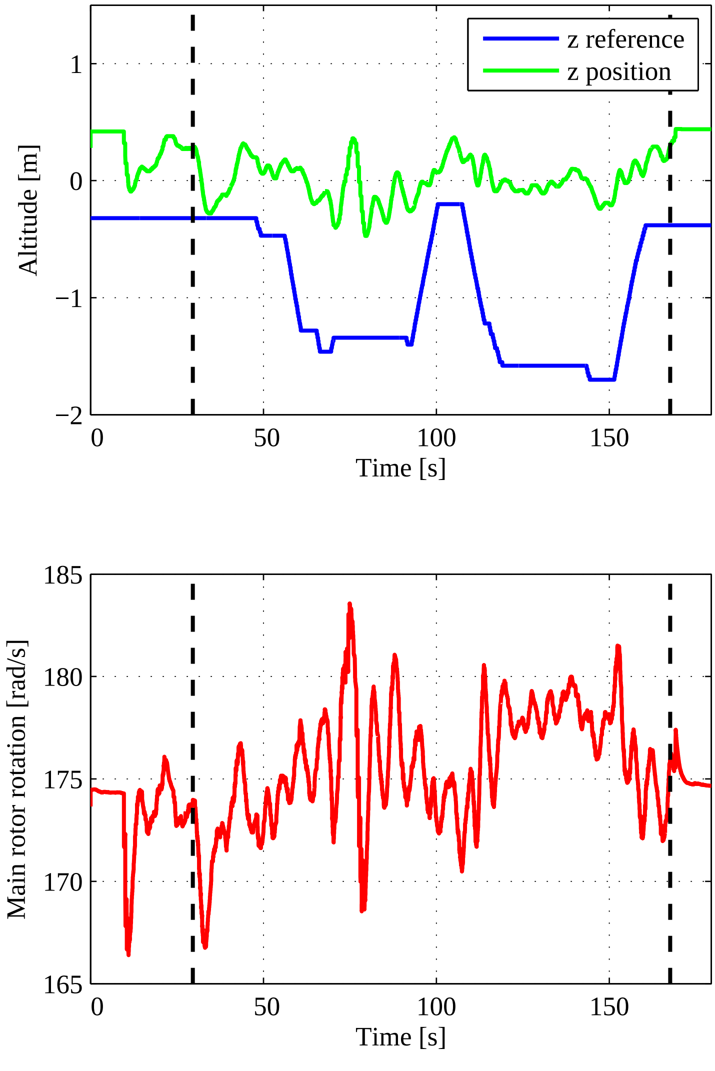


Figure 7.27: *Top: Preliminary test of the helicopter with altitude controller implemented. Bottom: The controller output during the flight test.*

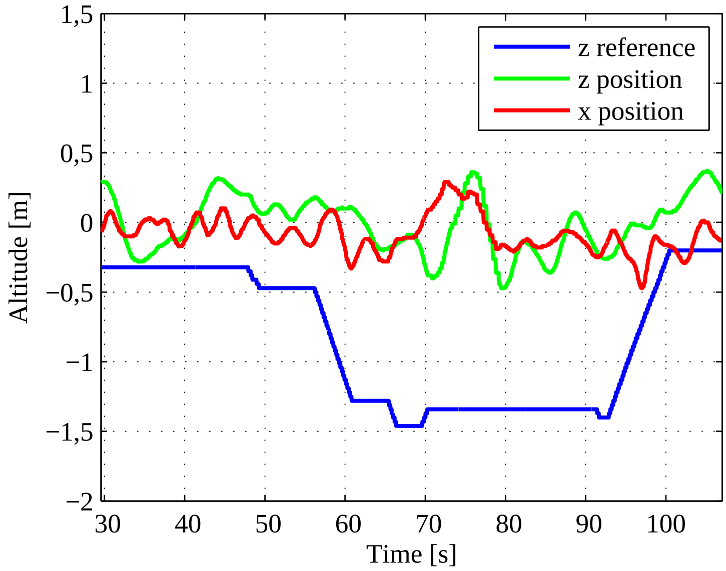


Figure 7.28: *Selected cut of Figure 7.27. The x position has been superimposed.*

The conclusion of the altitude controller test, is that it is able to control the helicopter altitude, even though some oscillation occurs due to air circulation in the test area. Furthermore, a steady state error is present, which is because the I_z gain is set to zero. The controller has to be tuned, to be able to compensate for these errors, but as the controller works, it has been prioritized to continue with the implementation of the horizontal controllers.
A video of a flight supports the documentation of a working altitude controller. The video can be found at [CD-ROM, 2007, `/videos/altitude_control.wmv`]. At time 0:01:03, it is seen that a red diode on the servoboard indicating autonomous flight, turns on.

The Lateral Controller

While keeping the altitude controller output connected, the wires are switched such that the lateral controller output is connected to the lateral pitch servo. When switching to autonomous mode, the pilot still has to control the longitudinal position and yaw rotation manually. The space, in which the helicopter can fly in the laboratory, is limited, and therefore the pilot has very short time to switch back to manual flight, if necessary. If the control input is allowed to change from minimum to maximum, this time would bee even shorter. In order to avoid huge controller output signals, a saturation of ±0,1 is therefore inserted at the output of the lateral controller. The gains used in this preliminary test of the lateral controller is given in Table 7.5.

Control Variable	$P_{(\cdot)}$	$I_{(\cdot)}$	$D_{(\cdot)}$
ϕ	−1	0	0
$\dot{y}$	0,1	0	0
y	1,2	0	0

Table 7.5: *Table of the lateral controller gains used for the preliminary tests of the real helicopter.*

Figure 7.29 shows a selected cut of a flight test. The vertical dashed line shows when the helicopter is switched to autonomous mode.

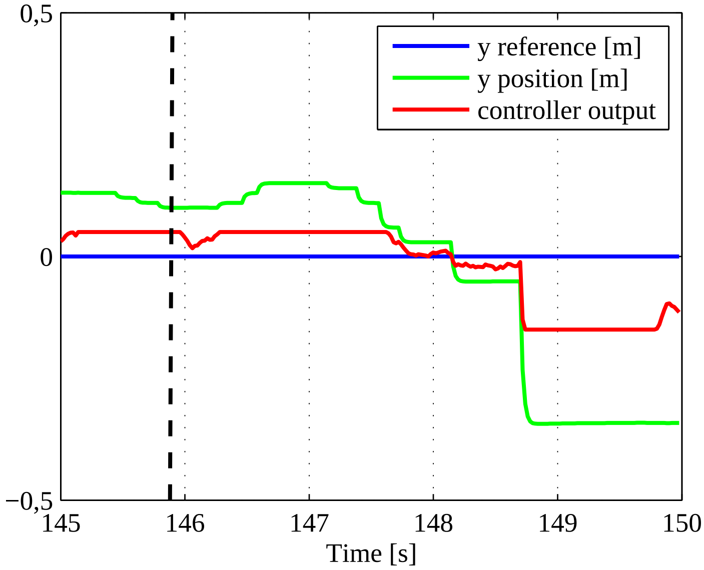


Figure 7.29: *Preliminary test of the helicopter with the altitude and lateral controller implemented.*

The reference is kept at zero. We can see, that the controller behaves as expected: a positive y axis position results in a positive lateral controller output, and when the helicopter crosses the 0 position, so does the controller output. Remember that a positive S_{lat} makes the helicopter move in the negative y axis direction. So the question is why the helicopter is not stabilized in the reference. First of all, we see that the control signal saturates most of the time. This means slower settling time. Furthermore, the update frequency of the y estimate decreases when the helicopter starts to move. This happens at time 147,8 s, where the width of the steps increases from approximately 0,2 s to nearly 0,4 s. In that time the helicopter has moved just over 15 cm, and it gets even worse from time 148,2 s. The problem is that the region of interest in the image processing software is to small. When the helicopter moves outside the ROI, the IPC must do a full image search to find the marker, and this slows the process down. The consequence is that the controller gets the position information to late. At time 148,7 s the position is updated, and the controller saturates immediately to compensate for the wrong position. The constant y position from this time and forward can be explained by the fact that the helicopter has left the ROO of the front camera, and the IPC returns the last known position of the marker to the CC. The controller's effort to get the helicopter back, was however too late. At time 149,7 s, the helicopter crashes into the wall.

Current *PID* Control Perspective

When the helicopter crashed, the main rotor and servoboard was unfortunately damaged. It was however possible to do a few more test flights before the main rotor and the servoboard broke down. The few tests indicated that the lateral controller worked, but had to be tuned further, because it was still too aggressive. Afterwards it has been discovered that the ROI was set way too low, which meant that a full update was done almost as soon as the helicopter started to move. On top of this the IPC was set to save the image to the hard disk when a full update was performed. This has contributed to the slow position update, and it is expected that when this is fixed, it will improve the controller considerably.

As written in Section 4.3.2 on page 30, the IMU acceleration output is saturated when using the current IMU. It can therefore not be used to estimate the state. It has also later been found out that a small error has been made when compensating for this issue. The error has affected the state estimate of $\dot{\Xi}$ to have an offset, and therefore a constant velocity is estimated. This makes it even harder for the controller.

Given the explained conditions, it is estimated that when the hardware problems are fixed, we are very close to obtain the goal of stabilizing the helicopter in hover.

7.3 Advanced Controllers

When hovering is achieved, it is desired to develop more advanced controllers for other flight trajectories than hover. Therefore this section is, with basis in some of the numerous articles on the subject, dedicated to analyze more advanced controller types. Apart from the conventional *PID* controller, basically five different types of controllers (or combinations thereof) has been found in the literature:

1. *PID* control
2. LQR control
3. Robust control
4. Fuzzy control
5. Adaptive control
6. Nonlinear control (in various forms)

Some of the different types are described in the following sections.

7.3.1 Other *PID* Type Controllers

A lot of literature treats more advanced or extended type *PID* structure controllers for helicopter and flight control. Liu [2003] uses several *PID* controllers, each tuned to satisfy one given performance criteria, and through convex optimization a combination of these controllers are calculated in the overall controller.

Many papers treats *PID* controllers in combination with neural networks (NN). In Park et al. [1999] the attitude of a helicopter model is controlled by using neural networks to automatically tune the gains in each *PID* controller. This is a way to improve the performance of the inner control loop in the *PID* control structure shown in Figure 7.2.

Also fuzzy logic controllers are used. For example in an altitude controller developed in Zein-Sabatto and Zheng [1997], where a *PID* controller is made to control the cyclic pitch (i.e. the attitude of the helicopter), and a fuzzy controller is made for controlling the speed

of the main rotor. Hereafter a NN model is trained based on real flight data, and based on this model the coefficients of the two controllers are derived.
The modular approach by neglecting cross couplings and having an inner and outer loop reduces the complexity of the controllers, but also reduces capabilities such as robustness and disturbance rejection of the controllers. In De Nardi et al. [2006] a NN based control structure is used to partly remove this modularity. In a working *PID* control structure, each loop is incrementally substituted by a NN based controller. For every substitution only one NN is allowed to evolve, while the others are frozen. First the NN is made to tune a P controller for the yaw motion. Subsequently the outer velocity loop is substituted by a multi-layer perceptron (MLP) controller, and at last the inner attitude loop is replaced with a MLP controller. Finally the different NN based controllers are allowed to evolve together so the inner and outer loop can co-adapt together, which is not possible for the independent *PID* controllers. According to De Nardi et al. [2006] this controller type performed very good and robust. One of the great advantages compared to this project is the possibility to gradually exchange a working *PID* controller with a more advanced controller.

7.3.2 Linear model Based Controllers

The above described controllers uses only a reference error to generate the control input for each actuator individually, and hence no cross coupling effects are taken into account. For more aggressive maneuvers than hover, cross coupling becomes more pronounced. To take into account all cross couplings and system non-linearities, knowledge of the model is necessary. The following described model based controller types do this to different extents. First some linear controllers are considered.

LQR

Linear quadratic regulation (LQR) or optimal control minimizes the weighted $\mathcal{H}_2$ norm of the state error plus the control input (performance vs. robustness) by use of a linearized state space system model. This is the controller type used in Hald et al. [2006]. When the linearized state space model and the weight function are obtained the control law can be calculated..
The initial task is however to determine the wanted state reference. This is not trivial, since a given reference may be obtained in several ways. For instance, the same translatory velocity in $\{E\}$ may be obtained by flying backwards, or by yawing 180 degrees and then fly forwards. The process of determining the wanted state is called trimming. This can be done by setting up some constraints and solve the non-linear system equations, which provides the wanted steady state solution together with the actuator control input. The easiest is to trim in hover, since many states have to be zero. Afterwards the non-linear model is linearized in this operating point, before a controller for this operating point can be obtained. The trimming and linearization method is described in [Hald et al., 2006, p. 91] and Bisgaard [2005]. For this method an additional integral effect can also be added to obtain zero steady-state error i.e. hovering at the same spot.
A more modular type of LQR has also been utilized [Gavrilets et al., 2002]. Here the control is split in an LQR controller for the combined longitudinal and vertical movement, and another for lateral movement and yaw rate. This method is simpler and has also showed

good results.

$\mathcal{H}_\infty$

Another type of linear control is robust control (minimizing the $\mathcal{H}_\infty$ norm). In Postlethwaite et al. [2005] different types of $\mathcal{H}_\infty$ controllers are tested. The controllers are developed from a linearized helicopter model and are primarily designed for robustness instead of performance, and tests show good results.

Gain Scheduling

Some difficulties arise when using linear control. The problem is that the linearized model is valid only for one specific operating point and in a small neighbourhood around this operating point. This is not a problem if the intention is to keep the helicopter at the same operating point at all times, but it is, if it is wanted to make a model that also supports more advanced manoeuvres, i.e. combinations of forward flight, curve flight, hover, loops, etc. One approach to obtain this performance is to use gain scheduling, and in fact gain scheduling originates from flight control systems [Khalil, 2000, p. 185]. Gain scheduling is a way to incorporate some of the non-linearities of the model by linearizing in several operating points instead of only one. For points not associated with a linear controller, control is obtained by interpolation of the linear controllers in the neighbourhood. The difficult thing here is to verify that the transition between controllers are smooth enough and does not cause instability [Kadmiry, 2002, p. 5]. Because of the often strong non-linearity of a helicopter, the linear controllers are only valid in a very small neighbourhood of the operating point.

7.3.3 Nonlinear and Adaptive Control

If it is wanted to make one overall controller, adaptive or non-linear control can be utilized, and much literature also concerns these types of controllers. Nonlinear control usually concerns removing the non-linear terms using knowledge of the system model. For adaptive control (which is also a kind of non-linear control), the problem of gain scheduling is avoided since the controller continuously adapt itself to the current operating point.

Kadmiry [2002] has made an extension to an already working UAV control system, which uses vision sensors (which is also of interest in this project). The goal was to obtain "aggressive" maneuverability using fuzzy logic gain scheduled control. Fuzzy gain scheduling is used as opposed to conventional gain scheduling because global stability and robustness can be guaranteed. The controller is still split into an inner and an outer control loop (though modified compared to Figure 7.1 on page 130 [Kadmiry, 2002, p. 42]). The inner loop uses a Takagi-Sugeno fuzzy inference system model for the attitude to make a fuzzy gain scheduled controller. For the outer position controller, a heuristic fuzzy model is made, and a mamdani type fuzzy controller is developed.

For most other non-linear and adaptive control strategies the method of feedback linearization is the most widely used, though also other methods such as backstepping has been used. Feedback linearization is a technique where undesirable (non-linear) dynamics are canceled out by using knowledge about the system model and current state estimate feedback. This results in a combined system which is linear and therefore a linear controller can be developed. The method is also known as dynamic inversion, since the model is dynamically inverted and used to continuously linearize the combined system. The basic idea is shown in the block diagram in Figure 7.30.

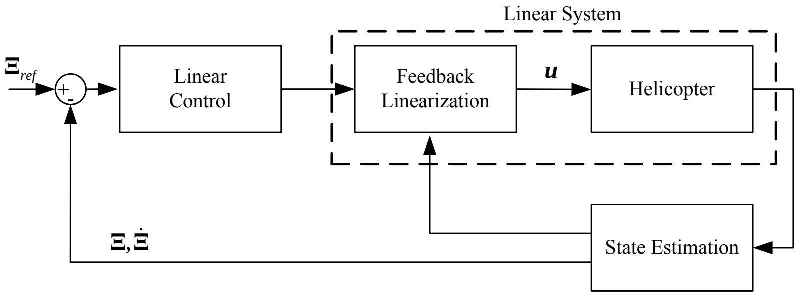


Figure 7.30: *Illustration of the basic idea behind non-linear control, in this case feedback linearization. By knowledge of the state and the system model, the non-linear terms are removed so the combined system can be seen and controlled as a linear system.*

Ito et al. [2002] gives a description of how to use dynamic inversion as a mean to control a UAV. In this case dynamic inversion is used as an inner loop controller combined with an outer loop controller using other control techniques e.g. PID control (an example of the technique applied to a simple system is given).

The problem of dynamic inversion for simple maneuvers as hover or forward flight, where the state operating point does not change, can also be solved offline. This is exactly what is utilized in the linear controllers in e.g. Hald et al. [2006]. This means that the dynamic inversion is actually a continuous trimming and linearization of the helicopter model.

The advantage of the feedback linearization technique is that when a complete and accurate mathematical description of the system model is given (which is the case for this project), then the linearization can be fully computerized, though it is more advanced.

In order to make this type of control work well, it is necessary to have a very accurate dynamic model to be able to exactly cancel all the undesired dynamics, otherwise this controller type can also result in instability. Another way to obtain this accurate model is to dynamically adapt the model to fit the real world, and hence make an adaptive model which is dynamically inverted. This is the technique used in the UAV projects on Georgia Tech. Here a neural network based adaptive control is made. The NN is trained offline to provide an approximate inversion model. An on-line learning neural network is then used to compensate for the inversion error which arises from the discrepancies between the off-line trained model and the real world. See Calise and Rysdyk [1998], Prasad et al. [1999] Johnson [2000], Johnson and Kannan [2002] and `http://www.ae.gatech.edu/people/ejohnson/papers.html`. This method is also further developed, using so called "hedging" to accommodate for robustness against actuator saturation.

7.3.4 Comparison of Controllers

In Shim et al. [1998] some of the different types of controllers are compared. Three different types of control designs for hover and near hover are studied. The controllers are tested on a non-linear simulation model performing a vertical and a horizontal manoeuvre with and without disturbances (wind) and model errors (helicopter mass error). The used measurements are GPS and IMU data, and input are collective, lateral and longitudinal pitch together with collective tail rotor pitch. The tree controller types are:

- Linear robust controller: μ synthethis is used to control the hover linearized model. Greater weight is associated with robustness than performance.
- Fuzzy Logic control design: A *PID* controller is used to control x, y, z and ψ respectively and independently. The fuzzy controller transfers the *PID* output to actuator input in an open loop.
- Nonlinear Tracking Control with input-output linearization: The model is the same as in Mustafic et al. [2005].

Results

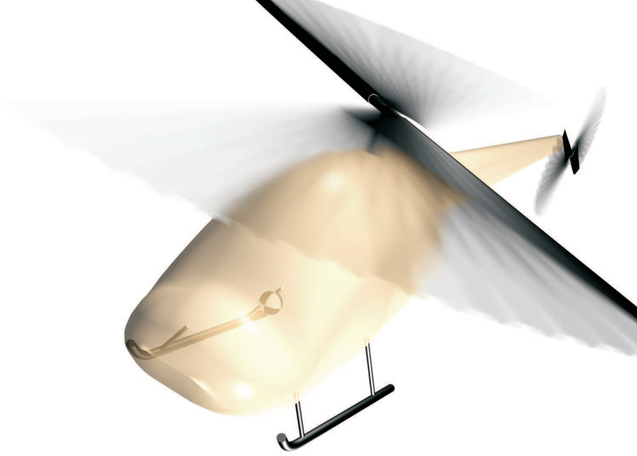
The non-linear controller is very fast and precise, compared to the other types of controllers. However, for even small disturbances and model uncertainties a steady state error occurs, which means that the controller is very sensitive to model inaccuracies. This, however, must be possible to minimize by introducing integral action. The robust controller shows poor transient response but is very robust, and a small steady-state error occurs. The fuzzy controller achieves good transient response and has no steady state error because of the utilized integral control.

7.4 Partial Conclusion

In this chapter it has been chosen to implement *PID* controllers to stabilize the helicopter in hover. It has been chosen to develop controllers for each direction (x,y,z,ψ) individually and disregarding cross-couplings. The *PID* controllers are designed with basis in a simplified linear model of the helicopter in hover. After this, the *PID* gains are tuned on the non-linear model, which they are able to stabilize. The altitude controller is implemented and is working. Due to a crash it has only been possible to make preliminary tests of the lateral controller. The test results have been promising, and it is expected that it is possible to make the helicopter hover in the near future.
To prepare for later projects on the helicopter, different types of more advanced controllers are investigated. The controller types are presented and evaluated, but no general conclusions are made regarding of which type to use.

8

EPILOGUE



Chapter Contents

This chapter finalizes the main report, by summarizing the report and presenting the project status at the time of writing. This leads to an overall conclusion of the project to answer the problem formulation in Section 2.3 on page 10. and finally some suggestions for further work are presented.

8.1 Summary of the Report

The purpose of this project is to make a small scale electric helicopter hover autonomously in the laboratory. To be able to obtain this goal, a number of intermediate objectives has been set up in Section 2.4 on page 10. These intermediate objectives are

1. hardware implementation,
2. modelling,
3. state estimation,
4. control design and implementation.

Each of the four objectives have been dedicated a chapter in the report. Focus has been on reaching the final goal, autonomous hover, and it has therefore been chosen to keep the development towards accomplishing each objective as simple as possible. In each individual objective, there is enough work for a whole project, so to accomplish all objectives, the fine tuning and optimization of each of the subsystems has been rated as lower priority. When a working solution for a subsystem has been obtained, it has been tested before integrated with the other subsystems. This procedure has resulted in a foundation with working subsystems, where each can be subject for further work and optimization.
The following sections summarizes the results obtained in each main chapter of the report, and the current status of the objectives are listed.

8.1.1 Hardware

The considerations and overview over the hardware is described in Chapter 4 on page 23.
A servoboard to control the servos on the helicopter has been available since the beginning of the project. The servoboard makes it possible to switch between manual and autonomous control of the helicopter. The servoboard is developed for a Bergen Industrial Twin helicopter used in other projects, and it is therefore not dedicated to the Corona implementation. Due to this, it was necessary to make some modifications to the servoboard.
It has been chosen to use cameras to estimate the location of the helicopter. Two cameras are placed on the wall, perpendicular to each other, and one camera is mounted on the helicopter. An Image Processing Computer (IPC) is used to locate the helicopter in the images. An IMU is mounted on the helicopter to obtain inertial measurements, but unfortunately the ± 2 g IMU saturates. Therefore a ± 10 g IMU has been ordered, but has not arrived yet.
To obtain an easy interface to control the helicopter, it has been chosen to use an external Control Computer (CC), with a serial connection to the helicopter. An external power supply is also used to have power for longer consecutive flight tests.
Because of a helicopter crash in the end of the project period, the servoboard is partly damaged. It might be possible to repair the servoboard, but the extend of the damage is not fully known. To ensure secure operability of the system, it is necessary to make a new servoboard, and therefore it is suggested to make a servoboard dedicated to the Corona 120 helicopter.

8.1.2 Modelling

The modelling (described in Chapter 5) has taken its starting point in a previous developed helicopter model for the Bergen helicopter. An overview over this existing model has been given, and the model has afterwards been adapted to the Corona helicopter. After implementation of the changes, simulations has verified that the model behaves as expected. Subsequently all parameters for the helicopter has been found by measurements and experiments. Adjustment of the parameters according to real flight data has not been done, since it is not expected to change the parameters significantly.

8.1.3 State Estimation

As written in Section 8.1.1 the utilized sensors are cameras and an IMU. The state estimation is divided into two parts. First, a method to find the position and attitude of the helicopter, from the image data obtained by the cameras, is developed. Then a sensor fusion algorithm is used to fuse the position and attitude data from the IPC, with the IMU data.
On the IPC, a Circular Hough Transform (CHT) is used to find circular markers placed on the helicopter and on the ground. An algorithm to calculate the position of the helicopter, from the pixel positions of the markers located on the helicopter, is developed. The imagery from the helicam together with the position of the helicopter is then used to calculate the attitude of the helicopter. The cameras, used to find the position of the helicopter, are seen as a substitution for a GPS and possible a magnetometer, which should be used when flying with the helicopter outside the laboratory in future projects.
It has been chosen to make a state estimate of the 12 rigid body states (position $\boldsymbol{\Xi}$, attitude $\boldsymbol{\Theta}$, velocity $\dot{\boldsymbol{\Xi}}$ and angular velocity $\dot{\boldsymbol{\Theta}}$), and use these for feedback control. This also makes it possible to use a simplified rigid body model to estimate the state, instead of the full dynamic model. The sensor fusion is done in an IMU driven extended Kalman filter, which uses the IMU measurements as input vector to the process model. As measurements, the position and attitude estimate from the IPC is used, and this data is transmitted from the IPC to the CC by a serial connection. Because of the implementation in a Kalman filter, these measurements can be easily substituted by GPS and magnetometer measurements.
The state estimation algorithm has been tested by moving the helicopter around manually. It has been verified that all the subsystems within the state estimation works as expected, though a small delay in the measurements coming from the IPC, is seen.

8.1.4 Control

Methods to control the helicopter has been investigated in Chapter 7. It has been chosen to implement decoupled *PID* controllers for each of the axes, x, y and z, and for the yaw angle. First a simple decoupled linear model has been developed, to use when designing the controllers. The derived controllers have then been tested and tuned on the non-linear model developed in Chapter 5. It showed possible to stabilize the helicopter in hover. Implementing the controller on the real helicopter was initiated, and flight tests with the altitude controller implemented was successfull, even though the gains were decreased to obtain a less aggressive controller. The lateral controller was also behaving as expected, however, the controller was not fast enough, because of a delay in the state estimate, and

the helicopter crashed. In spite of this, we expect that by tuning of the controllers it will be possible to stabilize the helicopter in hover.
Last, some more advanced controllers have been investigated, to find out how to continue when the helicopter is in hover using the simple controller.

8.1.5 Project Status

In Table 8.1, an overview of the status of each objective and its sub-objectives at the end of the project period is presented. For further information on the objectives, see Section 2.4.

Objective	Status	Remark
1. Hardware Implementation		
1.1 Consider hardware system structure	Done	
1.2 Sensor selection and implementation	Done	The IMU offset from CM is disregarded.
1.3 Implement actuator controlling devices	Done	Problem with the servoboard must be solved. Preferably, a new, redesigned servoboard which is dedicated for the helicopter, will have to be implemented.
1.4 Set up system computers	Done	
1.5 Choose and design interfaces between hardware parts	Done	
1.6 Provide proper power supply for the whole system	Done	
2. Modelling		
2.1 Adapt the existing model to the Corona 120	Done	
2.2 Verify the model	Done	
2.3 Determine the parameters for the Corona 120	Done	
3. State Estimation		
3.1 Design markers to mount on the helicopter and the floor	Done	
3.2 Develop image processing software to track the markers	Done	
3.3 Combine the redundant information from three cameras	Done	
3.4. Obtain state estimate	Done	
4. Control Design and Implementation		

Continued on next page

Table 8.1: *Current status of the objectives described in Section 2.4 on page 10.*

Continued from previous page

Objective	Status	Remark
4.1 Design a feasible controller	Done	
4.2 Implement and test the controller in the simulation environment	Done	
4.3 Test the controller on the real helicopter	Initiated	Altitude contorller works, but needs to be tuned. Preliminary tests have been conducted on the lateral controller.
4.4 Investigate and design advanced controllers	Initiated	Investigated, but not designed.

Table 8.1: *Current status of the objectives described in Section 2.4 on page 10.*

8.2 Conclusion

The overall goal for this project has been to make a small scale helicopter capable of hovering autonomously. In simulation, a *PID* controller has been tested on the non-linear model of the helicopter. The controller was not only capable of stabilizing the helicopter in hover, it was also able to follow a reference input in all three dimensions. Instability arose when yawing the helicopter, but this is believed to be a minor problem, caused by the fact that the body frame rotates, and thereby becomes unaligned with the earth frame.
On the real helicopter, preliminary tests of an altitude controller has been successfull, and it is believed that it is a question of tuning to stabilize the helicopter in a desired altitude. In the lateral direction, a controller was also tested, and the controller output was proved to behave as expected, even though it was not fast enough to stabilize the helicopter. The reason for the this behaviour, was mainly a delay in the state estimate.
Unfortunately, hardware problems with the servoboard occured after a crash during a test flight. This prevented us from finishing the implementation and test of the *PID* controller. However, based on preliminary flight test data gathered before the crash, it is believed to be possible to make the controller stabilize the helicopter in hover, by tuning the controller gains appropriatly.

8.3 Future Work

The initiating problem in Chapter 1 describes the HAMOC (Helicopter Aided Mapping Of Crops) system, and how we imagine the application could be used in a typical use scenario. Obviously, there is a lot of work to do before this scenario is possible. We will in this section look into the near future, and on the basis of our experience with the project suggest the imminent tasks to be done. After this, the longer term perspectives and tasks are identified.

8.3.1 Hardware

The problem with the servoboard has only been fixed temporarily at the time of writing, so a task for the near future is to find a more permanent solution. An optimal solution would be to redesign the servoboard layout, such that it is dedicated for the Corona helicopter used in this project. This includes changing the power supply, such that all low power circuits on the helicopter are powered through a supply, which is separated from the servo motor supply.
The helicam could advantageously be replaced with another webcam, which has a wider angle of view. With the present helicam, the helicopter has very little range of movement before the marker has left the helicam view. This deteriorates the attitude estimate of the helicopter significantly.

8.3.2 State Estimation

Regarding the state estimation, some delay in the image processing has been observed. This delay can be taken into account by implementing a smith predictor or compensator [Franklin et al., 2002, p. 609]. A smith predictor uses the process model to predict the current state, when it is known that the measurements are a certain amount of time old. Furthermore, as it is now, the IMU driven kalman filter does not take the control input (S_{mr}, S_{lat}, S_{lon} and $\dot{\psi}_{ref}$) into consideration. The control input contains valuable information about whereto the helicopter might move in the near future. Therefore, it is suggested to add the control input to the input vector (given in Equation (6.50)) of the used process model.

8.3.3 Control

Another task is obviously to finish the *PID* controller tuning, and hopefully make the helicopter hover autonomously. To achieve this, thorough tests of the individual controllers (altitude, lateral, longitudinal and yaw) will be necessary. In the tuning process, it may be an advantage to obtain a certain level of performance of one controller, before proceeding to the next.
Some improvements might be done to ease the tuning process, and to increase the performance of the controller. The linear model can be further developed by including the force of the attached wire, thus making the linear model correspond better to the non-linear model.

8.3.4 Long Term Tasks

To reach the goal of making the helicopter fly autonomously in an outside environment as intended in the HAMOC project, some more comprehensive tasks have to be solved.
When the helicopter is hovering in the lab, it is suggested to move the setup outside, and replace the cameras with a GPS and a magnetometer, but still with an external computer to control the helicopter. The external computer provides an interface, by which it is easy to develop more advanced controllers.
While hovering autonomously with GPS and magnetometer as position sensors, it will be an advantage to develop a non-linear state estimator for more advanced flight trajectories.

This will have to be used at later stages of the project.
When the helicopter hovers autonomously outside, with a well working and robust controller, it is suggested to continue with implementing an on-board computer. But to have an easy interface, still have the external computer connected. By experience, it is a very demanding task to implement an on-board computer. When it is implemented, it makes the system development more difficult because of reduced computing power and decreased system overview (e.g. you do not have the possibility of running SIMULINK on an on-board computer). With an on-board computer implemented, then in small steps the on-board computer can be handed over the control. First, with a fully operational on-board system, it is advised to cut the wired connection (i.e. power and serial interface) to the helicopter.
It is our hope, that the project is carried on in the future, and that the HAMOC system eventually will become reality. It is out of the scope of this master thesis to elaborate on the future perspectives of the application, but the system will hopefully ease the collection of imagery of crop growth and enhance the quality of the data at the same time.

Bibliography

Bak, T. (2002).
Modeling of mechanical systems.
Technical report, Aalborg University.
Lecture notes from the course Modelling of Mechanical Systems I.
[CD-ROM, 2007, `literature/modelling/bak-MMS_I.pdf`].

Ballard, D. H. (1981).
Generalizing the hough transform to detect arbitrary shapes.
Pattern Recognition, 13(2):111–122.
[CD-ROM, 2007, `literature/vision/ballard-generalizing....pdf`].

Bisgaard, M. (2005).
Autonomous Helicopter Control for Mine-Detection Operations.
PhD thesis, Aalborg University.
Preprint - [CD-ROM, 2007, `literature/modelling/bisgaard....pdf`].

Bisgaard, M., la Cour-Harbo, A., and Bendtsen, J. D. (2007).
Model based vs. model free state estimation for helicopter slung load system.
American Instute of Aeronautics and Astronautics.
Preprint - [CD-ROM, 2007, `literature/estimation/bisgaard....pdf`].

Bramwell, A. R. S. (1976).
Helicopter Dynamics.
Edward Arnold Ltd.

Brennan, S. N. (1997).
Modeling and Control Issues Associated with Scaled Vehicles.
PhD thesis, New Mexico State University.
[CD-ROM, 2007, `literature/modelling/brennan-....pdf`].

Calise, A. J. and Rysdyk, R. T. (1998).
Nonlinear adaptive flight control using neural networks.
Control Systems Magazine, IEEE.
[CD-ROM, 2007, `literature/control/calise-nonlinear....pdf`].

CD-ROM (2007).
The enclosed CD-ROM.

Craig, J. J. (2005).
Introduction to Robotics - Mechanics and Control, Third Edition.
Pearson Prentice Hall.

De Nardi, R., Togelius, J., Holland, O., and Lucas, S. M. (2006).
Evolution of neural networks for helicopter control: Why modularity matters.
Evolutionary Computation.
[CD-ROM, 2007, `literature/control/daNardi-....pdf`].

Duda, R. O. and Hart, P. E. (1972).
Use of the hough transformation to detect lines and curves in pictures.
Commun. ACM, 15(1):11–15.
[CD-ROM, 2007, `literature/vision/duda....pdf`].

Franklin, G. F., Emami-Naeini, A., and Powell, J. D. (2002).
Feedback Control Of Dynamic Systems, Fourth Edition.
Prentice Hall.

Gavrilets, V., Martinos, I., Mettler, B., and Feron, E. (2002).
Control logic for automated aerobatic flight of a miniature helicopter.
AIAA Guidance, Navigation, and Control Conference and Exhibit, Monterey, California.
[CD-ROM, 2007, `literature/control/gavrilets....pdf`].

Grewal, M. S. and Andrews, A. P. (2001).
Kalman Filtering - Theory and Practice Using MATLAB.
John Wiley and Sons.

Hald, U. B., Hesselbæk, M. V., and Siegumfeldt, M. (2006).
Nonlinear modeling and optimal control of a miniature autonomous helicopter.
Master's thesis, Aalborg University.
[CD-ROM, 2007, `literature/stud_report/05gr937a....pdf`].

Holm, E. (2006).
Spionfly skal holde saab i luften.
Ingeniøren, 32(40):22–23.
[CD-ROM, 2007, `literature/misc/ing-saab_uav.pdf`].

Holmgaard, J. T., Jensen, C. S., and Jakobsen, S. L. (2006).
Development and navigation of an autonomous uav.
Master's thesis, Aalborg University.
[CD-ROM, 2007, `literature/stud_report/05gr937b....pdf`].

Ito, D., Georgie, J., Valasek, J., and Ward, D. T. (2002).
Reentry vehicle flight controls design guidelines: Dynamic inversion.
Technical report, NASA.
[CD-ROM, 2007, `literature/modelling/nasa....pdf`].

Jensen, R. and Nielsen, A. K. N. (2005).
Robust control of an autonomous helicopter.
Master's thesis, Aalborg University.
[CD-ROM, 2007, `literature/stud_report/05gr1034a_student_report.pdf`].

Johnson, E. N. (2000).
Limited Authority Adaptive Flight Control.
PhD thesis, Georgia Institute of Technology.
[CD-ROM, 2007, `literature/control/johnson....pdf`].

Johnson, E. N. and Kannan, S. K. (2002).
Adaptive flight control for an autonomous unmanned helicopter.
In *AIAA Guidance, Navigation and Control Conference*, Monterey, CA.

[CD-ROM, 2007, literature/control/kannan...2002.pdf].

Johnson, E. N. and Kannan, S. K. (2005).
Adaptive trajectory control for autonomous helicopters.
Journal of Guidance, Control and Dynamics, 28(3).

Johnson, W. (1994).
Helicopter Theory.
Dover Books.

Kadmiry, B. (2002).
Fuzzy control for an unmanned helicopter.
Master's thesis, Linköping University.
[CD-ROM, 2007, literature/control/kadmiry....pdf].

Khalil, H. K. (2000).
Nonlinear Systems.
Prentice-Hall. Inc., third edition.

Kimme, C., Ballard, D., and Sklansky, J. (1975).
Finding circles by an array of accumulators.
Commun. ACM, 18(2):120–122.
[CD-ROM, 2007, literature/vision/ballard-circle....pdf].

La Civita, M. (2003).
Integrated Modeling and Robust Control for Full-Envelope Flight of Robotic Helicopters.
PhD thesis, Carnegie Mellon University, Pittsburgh, PA.
[CD-ROM, 2007, literature/modelling/civita-....pdf].

Liu, H. H. (2003).
"pid" type control for multiple performance: a flight control study.
In *American Control Conference*, volume Vol. 2, pages 1643–1648.
[CD-ROM, 2007, literature/control/hugh....pdf].

Marfeldt, B. (2005).
Høje ambitioner og små penge bag flyfiasko.
Ingeniøren, 31(23):12–13.
[CD-ROM, 2007, literature/misc/ing-taarnfalken.pdf].

Miller, M. P. and Soulé, H. A. (1933).
The experimental determination of the moments of inertia of airplanes.
Technical Report 467, NASA.
[CD-ROM, 2007, literature/modelling/nasa-experimental....pdf].

Mustafic, E., Petterson, R., and Fogh, M. (2005).
Nonlinear control approach to helicopter autonomy.
Master's thesis, Aalborg University.
[CD-ROM, 2007, literature/stud_report/05gr1033....pdf].

O'Gorman, F. and Clowes, M. B. (1973).
Finding pictures edges through collinearity of feature points.

In *Proc. of the 3rd IJCAI*, pages 543–555, Stanford, MA.
[CD-ROM, 2007, `literature/vision/gorman....pdf`].

Padfield, G. D. (1996).
Helicopter Flight Dynamics: The Theory and Application of Flying Qualities and Simulation Modeling.
AIAA.

Park, D., Lee, J., and Ha, H. (1999).
Attitude control of helicopter simulator using neural network based pid controller.
Fuzzy Systems Conference Proceedings, vol.1:465–469.
[CD-ROM, 2007, `literature/control/park....pdf`].

Pololu (2005).
Specifications for Pololu servo controller board.
[CD-ROM, 2007, literature/hardware/pololuServoController.pdf].

Postlethwaite, I., Prempain, E., Turkoglu, E., Turner, M. C., Ellis, K., and Gubbels, A. W. (2005).
Design and flight testing of various $\mathcal{H}^\infty$ controllers for the bell 205 helicopter.
Control Engineering Practice, 13(2):383–398.
[CD-ROM, 2007, `literature/control/postletwaite....pdf`].

Prasad, J. V. R., Calise, A. J., Corban, J. E., and Pei, Y. (1999).
Adaptive nonlinear controller synthesis and flight test evaluation on an unmanned helicopter.
In *IEEE International Conference on Control Applications*.
[CD-ROM, 2007, `literature/control/calise-adaptive...pdf`].

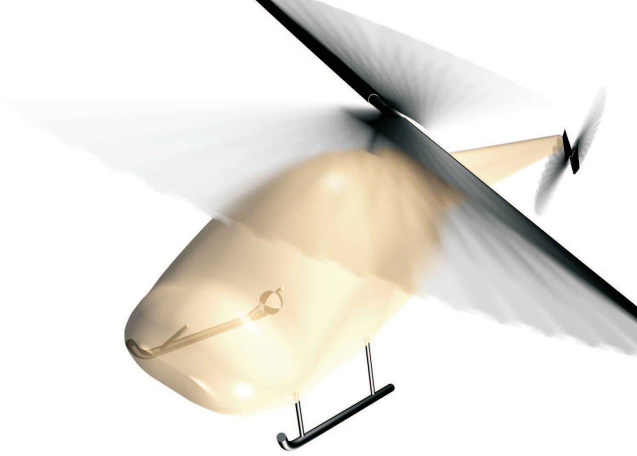
Prouty, R. W. (1990).
Helicopter Performance, Stability, and Control.
Krieger Publishing Company.

Shim, D. H., Koo, T. J., Hoffmann, F., and Sastry, S. (1998).
A comprehensive study of control design for an autonomous helicopter.
In *IEEE Conference on Decision and Control*.
[CD-ROM, 2007, literature/control/shim....pdf].

Weisstein, E. W. (2007).
Polytope, from mathworld–a wolfram web resource.
`http://mathworld.wolfram.com/Polytope.html`.

Zein-Sabatto, S. and Zheng, Y. (1997).
Intelligent flight controllers for helicopter control.
Neural Networks journal.
[CD-ROM, 2007, literature/control/sabatto....pdf].

Appendix



A

CD Contents

The content of the enclosed CD-rom is:

- **/autonomous_helicopter.pdf**
  A PDF version of the report.
- **literature/**
  - **/control**
    Articles about controlling helicopters.
  - **estimation/**
    The literature used for state estimation.
  - **hardware/**
    The datasheets of all the hardware used in this project.
  - **misc/**
    Miscellaneus literature.
  - **modelling/**
    The literature used for modelling of the helicopter.
  - **stud_report/**
    Student reports concerning UAV's from former semesters.
- **source/**
  - **/control**
    Files used for controller design.
  - **image_processing/**
    Files for the image processing algorithm on the IPC
  - **integration/**
    The combined SIMULINK *interface used on the CC.*
  - **interface/**
    S-functions for the serial connections on the CC.
  - **maple/**
    The maple scripts for deriving the model equations, both for clockwise and counter clockwise rotation.
  - **model/**
    The SIMULINK *model.*

- **parameters/**
  m-files used to determine the Corona parameters.
- **state_estimation/**
  Implementation of the sensor fusion algorithm.
- **test/**
  Testdata from various flight tests.
- **vision/**
  Articles about hough transform and vision.

- **videos/**
  Videos of the helicopter tests.

B

Servo Motor Control

This appendix describes the interface to the servo motors, and explains how the servo boards work. It defines the protocol used to control the servos. The source of information is the manual for the servo controller board. It can be found on the enclosed cd-rom [CD-ROM, 2007, `articles/hardware/pololuServoController.pdf`*].*

B.1 The Servo Controller Board

The control signal is generated by a servo driver board from *Pololu Micro Serial Servo Controller* (Pololu [2005]) acquired from Pololu Robotics and Electronics (http://www.pololu.com). It is possible to connect up to eight servo motors to one servo board. The input to the servo board is a serial bitstream with RS-232 levels. Up to 16 servo boards can be connected to the same serial input signal, enabling the user to control up to 128 servo motors with only one serial line. Each board has a size of 23 mm × 23 mm.
The supply voltage should be within 5 V and 16 V. The current consumption for one board is 5 mA in average. A picture of the board is shown in Figure B.1.

B.1.1 Setup of the Servo Board

The boards operate in two different modes: Mini SSC II mode and Pololu mode. In this configuration Pololu mode is used, as this supports a baud rate up to 38 400, whereas Mini SSC II mode only supports a baud rate of 2 400 or 9 600. Pololu mode also enables some more advanced features, e.g. setting the turning rate, altering the neutral position as well as the turning direction of the servos. The Pololu mode is set by removing the jumper in the top left corner on Figure B.1.
The serial line (TX and ground) is connected to the board from a computer serial port (e.g. COM1). If the serial cable has a DB9 male connector, TX is pin 3 and ground is pin 5.

B.1.2 Controlling the Servos

To control the servo motors, the following protocol is used. A string of five or six bytes are transmitted to the boards:

<startbyte>	<device ID>	<command>	<servo number>	<data 1>	<data 2>

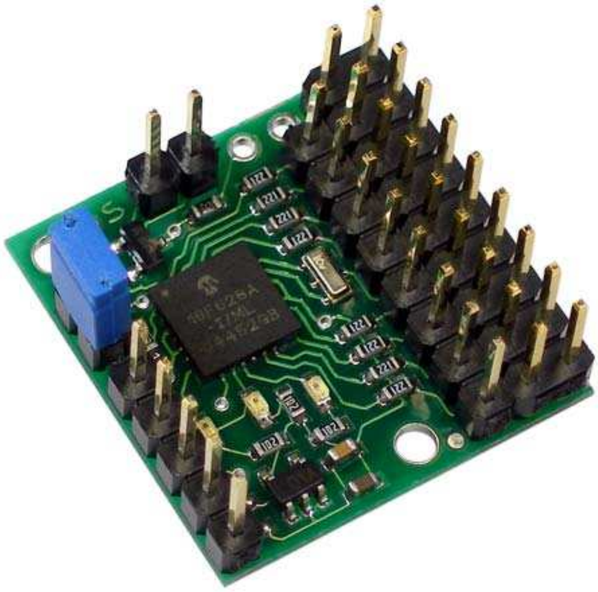


Figure B.1: *The Pololu Micro Serial Servo Controller. The connectors for the eight servo motors are placed on the right side. The jumper in the top left corner should be removed to set the board to "Pololu mode".*

- <startbyte>: Always 128.
- <device ID>: 1 for the Pololu Micro Serial Servo Controller.
- <command>: One of six different commands to send. An integer between 0 and 5, both included.
- <servo number>: The number of the servo motor to control - in this case an integer between 0 and 20, both included.
- <data 1>: First data byte. An integer between 0 and 127, both included.
- <data 2>: Second data byte (not used for <command> = 0, 1, and 2). An integer between 0 and 127, both included.

The Command numbers and data bytes are setup according to the following specifications:

<**command**> **= 0:** Set parameters (servo on/off, direction,range)
Only <data 1> is used. Bit 7 (the most significant bit, MSB) shall always be 0 (if not, the byte is interpreted as a <startbyte>). Bit 6 specifies whether a servo is on or off-if off there is no power delivered to the servo. 0 (default) turns off the servo, and 1 turns it on.
The following bit explanations will be easier understood when all the <commands> are read. Bit 5 sets direction, which applies when <command> = 2 and 3. If the bit is 0 (default) a larger number causes the output pulse to get longer; if the bit is 1, a larger number causes the pulse to get shorter. Seeing the servo motor from the front, i.e. with the shaft pointing towards yourself, the rotation will be clockwise, if the direction bit is set to 0 and the position value is increased.
Bits 0 - 4 sets the range through which the servo moves in position command 2 and 3. 0 makes the servo stay in neutral regardless of the position command, and 31 sets the range to its maximum (and even outside the mechanical restrictions for command 3). Default is 15, which makes the range approximately ±45° for command 2 and ±90° for command 3.

<**command**> **= 1:** Set turning rate
Only <data 1> is used. If the value is 0, the output pulse will change instantly to the new value. If nonzero, the pulse changes gradually from the old value to the new value. 1 is the slowest turning rate, and 127 is the fastest.

<**command**> **= 2:** Set 7-bit position
Only <data 1> is used. The position is set with reference to the range and direction specified with <command 0>, and the neutral position specified with command 5. The servo is automatically turned on, when setting a position.

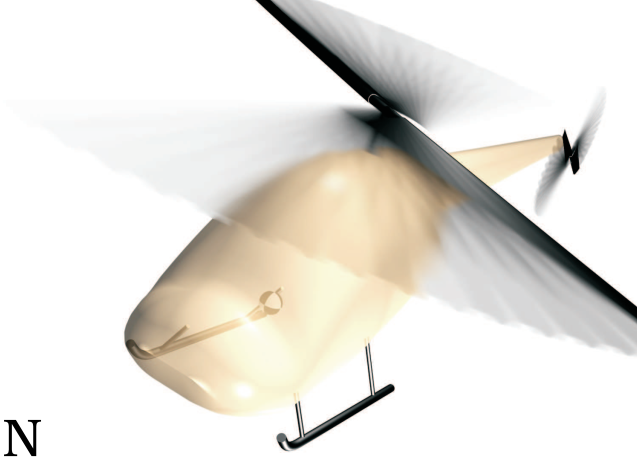
<**command**> **= 3:** Set 8-bit position
Both data bytes are used. In <data 1> only bit 0 is used, and in <data 2> only the seven least significant bits are used. Bit 7 of both data bytes must always be 0. <data 1> contains the most significant bit, and <data 2> contains the seven least significant bits. Together, the eight bits set a position value between 0 and 255, both included. The servo is automatically turned on, when setting a position. The angle extend (range) from 0 to 255 is determined with <command 0>.

<**command**> **= 4:** Set absolute position i.e. the pulse width
This <command> is the one used in this project to set the position. Both data bytes are used. <data 1> contains the upper bits, and <data 2> contains the lower seven bits. Bit 7 of the data bytes must always be 0. The range of the position is from 500 to 5 500, corresponding to a pulse width of 250 μs and 2 750 μs, respectively. As it shows, the absolute position value can be calculated from the pulse width in μs by multiplying with 2:

$$\text{absolute pos. value} = 2 \cdot \text{pulse width } [\mu\text{s}]$$

However, due to mechanical restrictions within the servo motor, the feasible range is approximately delimited to the values from 1 400 through 4 900. The servo is automatically turned on, when setting a position.

<**command**> **= 5:** Set neutral position
Works similar to command 4. The absolute position set with this command, will be the neutral setting for command 2 and 3. When setting the neutral position, the servo moves to this position. Default is 3 000.



C

MODEL IMPLEMENTATION OVERVIEW

This appendix serves as an overview of the implementation of the model. First it is described how the changes compared to the former model are implemented. Then a few notes on abstraction level of the input and output from the actuators are given, and finally the structure of the implementation in SIMULINK *is presented.*

C.1 Implementation of Model Modifications

The other modifications described in Section 5.2 are implemented in the model and can be switched on and off by using the following defines:

Counter clockwise rotation of the main rotor :
`#define COUNTER_CLOCKWISE`

Fixed collective pitch on the main rotor and dynamic rotation velocity :
`#define SWITCH_THETA_0_STATE`

Wire attached in the nose of the helicopter :
`#define WireAttached`

Stabilizer bar collective pitch :
`#define CollectiveSB`

The two last defines could also be implemented by setting the collective pitch and wire mass variables to 0 in the Bergen parameters. This approach is not used since it is easier not tho have the variable at all when they are not used.

C.2 Abstraction Level for Actuator Input and Output

The actuator input is a 50 Hz PWM signal with a duty cycle of $1-2$ ms. To make measurements and inputseasier to understand, this is in the servoboard interface converted to an input between -1 and 1. Therefore the input and output in the model is also between -1 and 1. Further the lateral and input S_{lat} is reversed so that a positive input corresponds to a positive movement in the body frame. The input and output from the main rotor is decided to be rad/s.

C.3 Implementation Structure

The model is implemented in an S-function in SIMULINK. The important input to the S-function containing the model is the input to the actuators (the others are mentioned below). The output is the state and derivative of the state. Before SIMULINK starts the simulation , all necessary variables are initialized. This is done by running the init function `init.m`. Following in this section, the implementation structure of the model S-function is presented. This is done by describing the execution order of functions and in which files the corresponding functions are located. Because the model is comprehensive, only the most important functions are described. For complete overview see the actual model files on [CD-ROM, 2007, `source/model/sf-files/`]. Basically the model is implemented in an S-function called `Model.cpp`, which contains some initialization functions and then the function called at every timestep (`mdlOutputs()`) by SIMULINK. In Figure C.1 the execution order in a model simulation as performed by SIMULINK is shown. First the model is initialized, and then the model runs in a loop until a stop is requested either from the code, when the simulation is done or if the stop button is pressed.
The structure of the `mdlOutputs()` function is shown in Figure C.2. As can be seen, the function consists of four steps. First the input data (actuator input, wind disturbances, if load is attached) is fetched from SIMULINK. Then the actual model call is performed, and the output state obtained from the model call, is send back to the output port of the S-function block, in SIMULINK. Finally, if real time simulation is enabled, the function suspends itself until a whole sample time has passed.

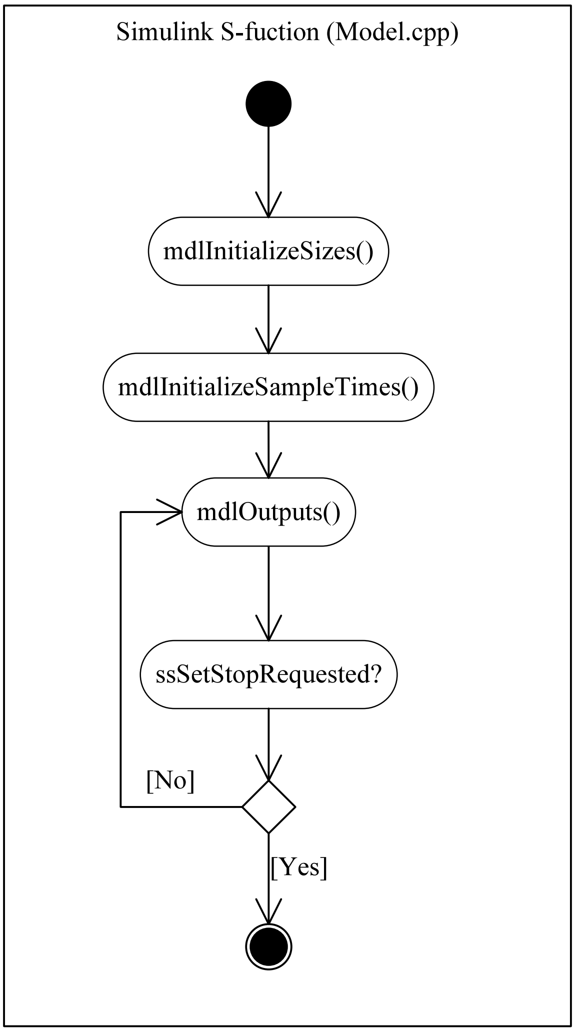


Figure C.1: *Overall S-function as it is used by* Simulink *during a model simulation.*

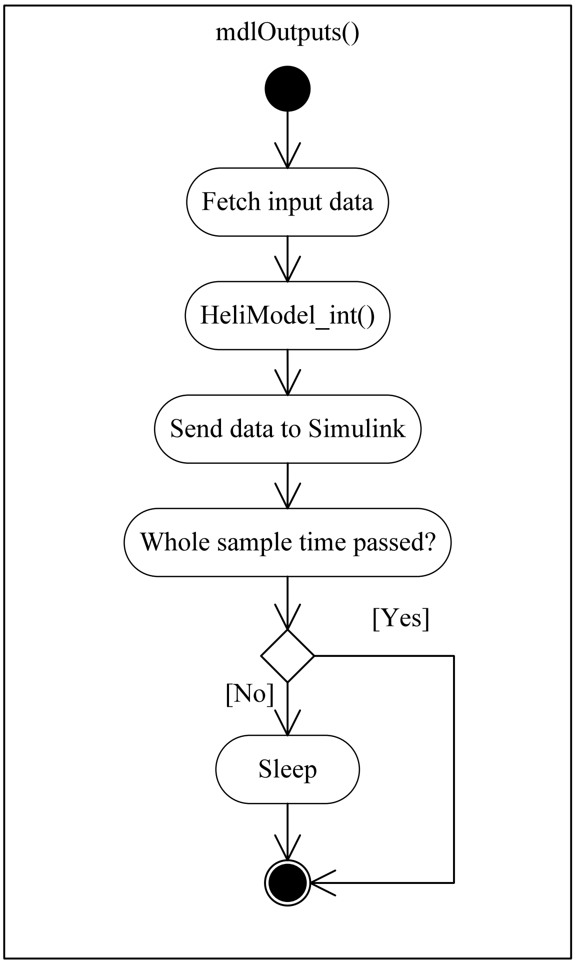


Figure C.2: *mdlOutputs().* Simulink *runs this function for each sample time.First the input data is fetched from Simulink, then the model is integrated and send to the output port and finally the function suspends until a whole sample time has elapsed.*

The function, which call the model `HeliModel_int()` is located in another file `HeliModel.cpp`. This is done so it is possible to use this function call from an ordinary C++ program instead of only SIMULINK. This is useful when later implementing the model on e.g. an on-board computer. The name refers to that the derivative of the model is calculated and then an integration is performed to obtain the next state. The structure of the function is shown in Figure C.3, where first the derivative of the model is found, the state is updated and at a new intermediate time step, the derivative is found again. This is done four times to obtain enough state information to calculate the fourth order Runge-Kutta integration (described in Section 5.3).

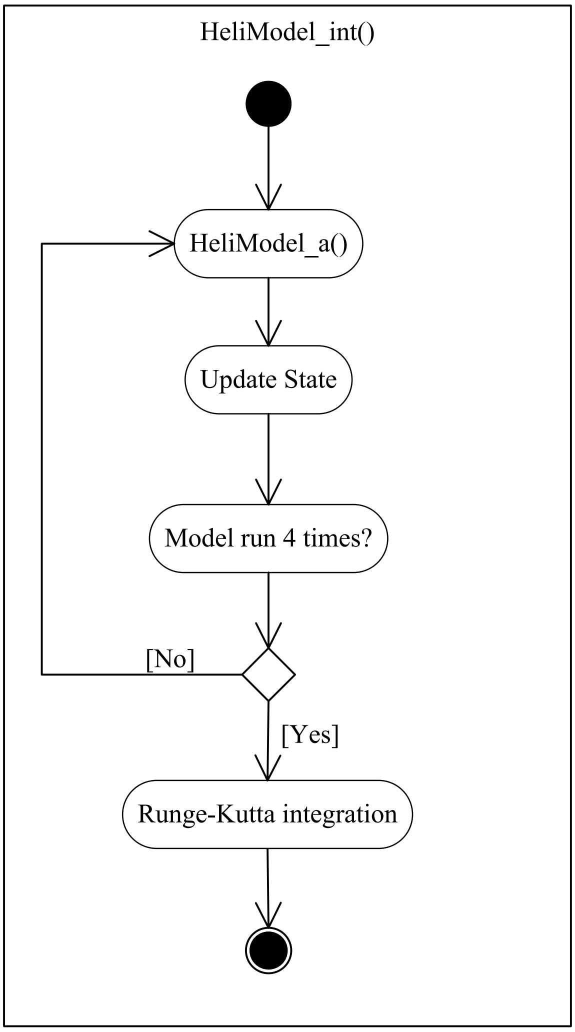


Figure C.3: *The mdlOutputs() function, shown in Figure C.2, calls HeliModel_int() each time to perform a one step Runge-Kutta integration of the model.*

The derivative of the current state is calculated in the function `HeliModel_a()`. The derivative is found by several subsequent calls to functions located in the file `HeliFunctions.cpp`. Each of the functions contains some of the model equations presented in the model overview Section (see 5.1 on page 39). The execution order of the functions is shown in Figure C.4, and the purpose of are described briefly in the following. Most of the functions are based on the output from the previous functions.

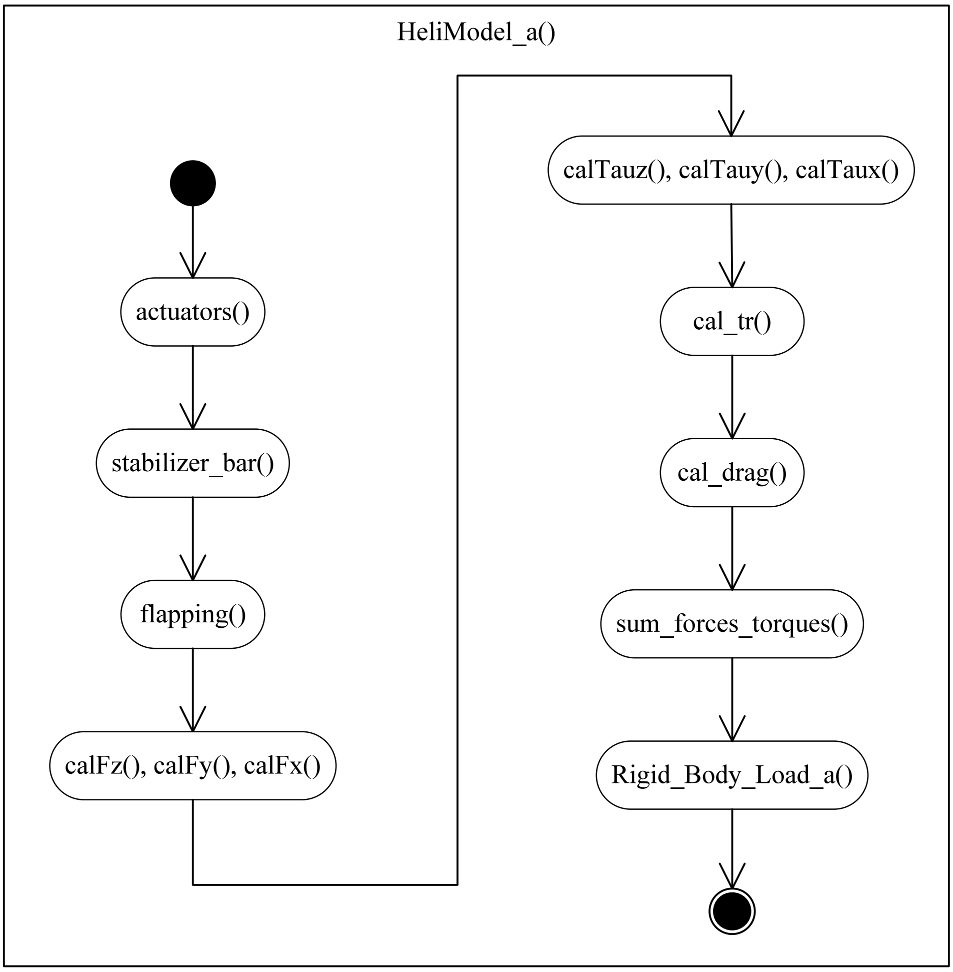


Figure C.4: *Helimodel_a() calculates the derivative of the model, which is used in the Runge-Kutta integration. The figure shows the execution order of the functions used to calculate the output of the equations given in Section 5.1.*

`actuators()` calculates the acceleration and velocity of each of the servos and the main motor - ($\dot{\Upsilon}$ and $\ddot{\Upsilon}$ from the transfer functions given in Equations (5.18)-(5.21)).

`stabilizer_bar()` calculates the flapping of the stabilizer bar based on the output from `actuators()` - ($\dot{\boldsymbol{a}}_{sb}$ and $\ddot{\boldsymbol{a}}_{sb}$ from Equation (5.27)).

`flapping()` calculates the flapping of the main rotor blades based on the output from `actuators()` and the flapping of the stabilizer bar. ($\dot{\boldsymbol{a}}_{mr}$ and $\ddot{\boldsymbol{a}}_{mr}$ from Equation (5.25)).

`calFz(),calFx(),calFy()` calculates the forces provided by the main rotor in the three directions. In `calFz()` the function `DirectLambda()` is used to calculate the analytical solution to the inflow λ_{mr} - (${}^{H}F_{z,mr}$, ${}^{H}F_{x,mr}$ and ${}^{H}F_{z,mr}$ from Equation (5.33)).

`calTauz(),calTaux(),calTauy()` calculates the torques provided by the main rotor around the three axis. This includes the torques provided by the forces calculated above - (${}^{B}\tau_{x,mr}$, ${}^{B}\tau_{y,mr}$, ${}^{B}\tau_{z,mr}$ from Equation (5.41)).

`cal_tr()` calculates the inflow ratio of the tail rotor by using `DirectLambda()`. From this the thrust and the torques provided by the tail rotor, is calculated - (${}^{B}F_{y,tr}$, ${}^{B}\tau_{x,tr}$, ${}^{B}\tau_{y,tr}$ and ${}^{B}\tau_{z,tr}$ from Equations (5.42) and (5.43)).

`cal_drag()` calculates the forces and torques provided by drag on the different parts of the helicopter - (${}^{B}\boldsymbol{F}_{d}$ and ${}^{B}\boldsymbol{\tau}_{d}$ from Equations (5.49) and (5.50)).

`sum_forces_torques()` sums the different forces and torques calculated in the above functions - (${}^{B}\boldsymbol{F}$ and ${}^{B}\boldsymbol{\tau}$ from Equations (5.51) and (5.52)).

`Rigid_Body_Load_a()` calculates the rigid body dynamics and kinematics from the obtained forces and torques - (${}^{B}\ddot{\boldsymbol{\Xi}}$ and ${}^{B}\ddot{\boldsymbol{\Theta}}$ from Equations (5.55) and (5.54), ${}^{B}\dot{\boldsymbol{\Xi}}$ and ${}^{B}\dot{\boldsymbol{\Theta}}$).

Some of the useful variables used in the code are listed below. Note that due to the generic nature of the mode, some variables like e.g. the state vector, is not entirely the same as in the model for the Corona helicopter:

$$^{*}MR = \begin{bmatrix} MR[0] \\ MR[1] \\ MR[2] \\ MR[3] \\ MR[4] \\ MR[5] \\ MR[6] \\ MR[7] \\ MR[8] \\ MR[9] \end{bmatrix} = \begin{bmatrix} \lambda_{mr} \\ \mu_x \\ \mu_y \\ \mu_z \\ {}^{H}F_{x,mr} \\ {}^{H}F_{y,mr} \\ {}^{H}F_{z,mr} \\ {}^{B}\tau_{x,mr} \\ {}^{B}\tau_{y,mr} \\ {}^{B}\tau_{z,mr} \end{bmatrix} \quad , \quad ^{*}TR = \begin{bmatrix} TR[0] \\ TR[1] \\ TR[2] \\ TR[3] \\ TR[4] \\ TR[5] \\ TR[6] \\ TR[7] \end{bmatrix} = \begin{bmatrix} \lambda_{tr} \\ \mu_{x,tr} \\ \mu_{y,tr} \\ \mu_{z,tr} \\ {}^{B}F_{y,tr} \\ {}^{B}\tau_{x,tr} \\ {}^{B}\tau_{y,tr} \\ {}^{B}\tau_{z,tr} \end{bmatrix} \tag{C.1}$$

Current input, state vector and output derivative:

$$^{*}Ui = \begin{bmatrix} \S_{mr} \\ S_{col} \\ S_{lat} \\ S_{lon} \\ \text{not used} \\ S_{tr} \end{bmatrix} \quad , \quad ^{*}Xi = \begin{bmatrix} \boldsymbol{\Xi} \\ \boldsymbol{\Theta} \\ \boldsymbol{\Xi}_{load} \\ \boldsymbol{\Theta}_{load} \\ {}^{B}\dot{\boldsymbol{\Xi}} \\ {}^{B}\dot{\boldsymbol{\Theta}} \\ {}^{B}\dot{\boldsymbol{\Xi}}_{load} \\ {}^{B}\dot{\boldsymbol{\Theta}}_{load} \\ \boldsymbol{a}_{mr} \\ \dot{\boldsymbol{a}}_{mr} \\ \boldsymbol{a}_{sb} \\ \dot{\boldsymbol{a}}_{sb} \\ \boldsymbol{\Upsilon} \\ \dot{\boldsymbol{\Upsilon}} \end{bmatrix} \quad , \quad ^{*}Xo = \begin{bmatrix} \dot{\boldsymbol{\Xi}} \\ \dot{\boldsymbol{\Theta}} \\ \dot{\boldsymbol{\Xi}}_{load} \\ \dot{\boldsymbol{\Theta}}_{load} \\ {}^{B}\ddot{\boldsymbol{\Xi}} \\ {}^{B}\ddot{\boldsymbol{\Theta}} \\ {}^{B}\ddot{\boldsymbol{\Xi}}_{load} \\ {}^{B}\ddot{\boldsymbol{\Theta}}_{load} \\ \dot{\boldsymbol{a}}_{mr} \\ \ddot{\boldsymbol{a}}_{mr} \\ \dot{\boldsymbol{a}}_{sb} \\ \ddot{\boldsymbol{a}}_{sb} \\ \dot{\boldsymbol{\Upsilon}} \\ \ddot{\boldsymbol{\Upsilon}} \end{bmatrix} \tag{C.2}$$

To define which helicopter is used, the parameters are included from a general parameter file `Parameters.h`. In this project it is defined to include the `CoronaParameters.h` file, which again includes `Parameters.cpp`. This order is shown in Figure C.5.

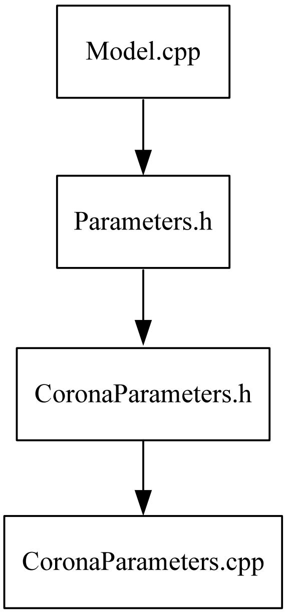


Figure C.5: *The Figure shows the order in which the parameter files are included. The helicopter type is determined in Parameters.h, by using e.g.* `#define Corona` *for the Corona helicopter.*

Finally Figure C.6 shows in which files to find the functions described above.

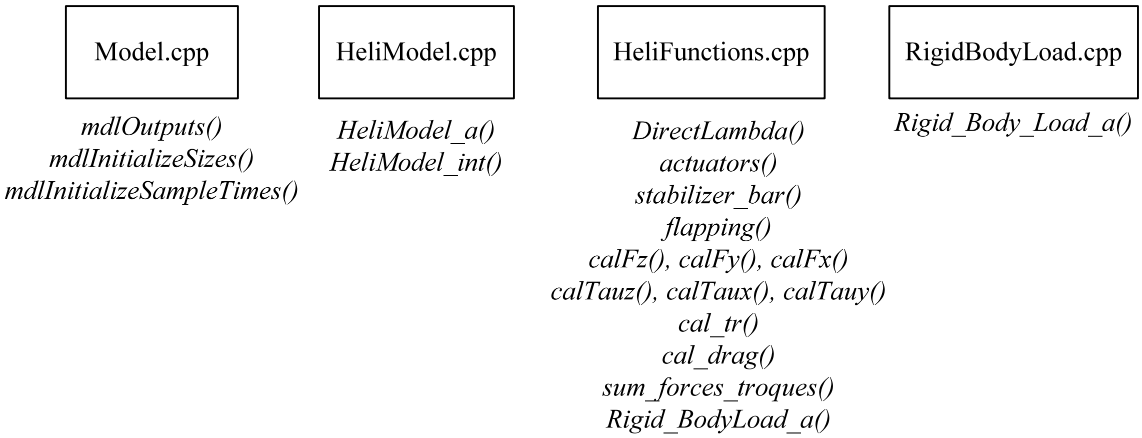
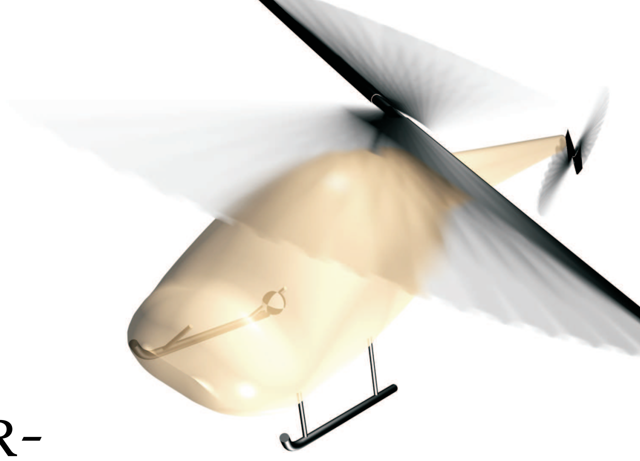


Figure C.6: *This Figure shows in which files to find the most relevant functions.*



D

Marker Colour Experiment

To be able to determine which colours are best to use as markers on the helicopter, an experiment with filming different colours, has been made.

D.1 Experiment and Results

A paper sheet with white markers and different colours as background (see Figure D.1) has been made.

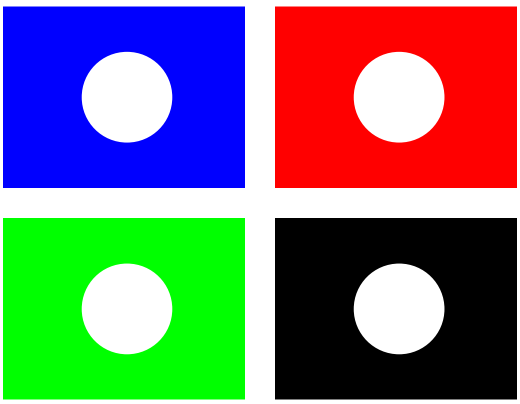


Figure D.1: *The marker background colours used for testing the contrast.*

Then pictures of the sheet has been taken with and without a floodlight projector enlightning the paper. Then the RGB-values of the output pictures has been analyzed. The average value over a 10 by 10 pixel area of each colour has been calculated at what seemed to be the lightest and darkest areas of the colours. The results can be seen in Table D.1.

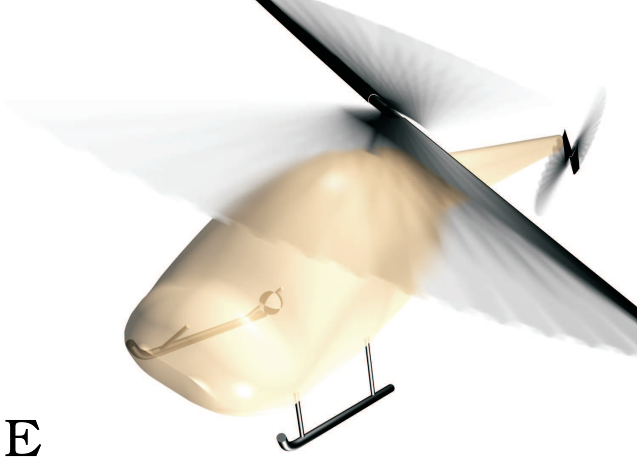
Note that the measurements without light actually have higher colour values than the measurements with light. It is assumed that this is because the camera auto ajusts the white balance, and therefore brightens the picture when there is no light. By inspecting the table it can be seen that the target colours are close to the desired limit (255), especially for the colour white. But for the non-target colours, then the value is rather high and

	Light			No Light		
	R	**G**	**B**	**R**	**G**	**B**
Red:	253	74	48	241	68	59
	254	115	91	254	121	112
Green:	99	192	158	100	202	178
	104	198	158	112	210	180
Blue:	80	120	213	92	138	245
	50	80	170	41	89	200
Black:	69	74	74	64	78	91
	65	66	67	57	67	83
White:	254	254	254	254	255	254
	253	253	253	254	255	252
Statistics:	Target	Other		Target	Other	
Average	233	98		240	96	
Std:	31	32		22	39	

Table D.1: *Measured RGB values for different colours printed on a paper sheet, with and without projektor light. Below in the table the averages and standard deviations for the target colour and other colours are calculated. The target colour for black is* $0-0-0$*, and white is* $255-255-255$*.*

has an average of almost 100. But not only a high value is observed, there is also a large deviation of the measurements. This leads to the conclusion that white is a good colour for the marker, but a background with lower values are desired. It is assumed that the high values are because the glossy surface of the paper reflects a lot of the light. Therfore a non-glossy red cloth has been tried as well. This yielded the following results, which is substantially better than the paper. Therefore a red cloth is chosen as marker background.

	Light			No Light		
	R	**G**	**B**	**R**	**G**	**B**
Red Cloth:	255	38	0	231	27	0



E

Verification of the Process Model

In the sensor fusion in Section 6.3, a rigid body process model is used instead of the full dynamic non-linear model derived in Chapter 5. In this appendix the process model is tested to verify that it is correct.

E.1 Test Procedure

The developed process model in Equation (6.67) consists of the following formulas:

$$
\begin{aligned}
{}^{E}\hat{\boldsymbol{\Xi}}_{k|k-1} &= T_s \cdot \left({}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}\, {}^{B}\hat{\dot{\boldsymbol{\Xi}}}_{k-1|k-1} \right) + {}^{E}\hat{\boldsymbol{\Xi}}_{k-1|k-1} && \text{(E.1)}\\
{}^{E}\hat{\boldsymbol{\Theta}}_{k|k-1} &= T_s \cdot \left({}^{E}_{B}\hat{\mathcal{R}}_{k-1|k-1}\, {}^{B}\hat{\dot{\boldsymbol{\Theta}}}_{k-1|k-1} \right) + {}^{E}\hat{\boldsymbol{\Theta}}_{k-1|k-1} && \text{(E.2)}\\
{}^{B}\hat{\dot{\boldsymbol{\Xi}}}_{k|k-1} &= T_s \cdot \left(\ddot{\boldsymbol{\Xi}}_{IMU,k-1} + {}^{B}_{E}\hat{\mathcal{R}}_{k-1|k-1}\, {}^{E}\boldsymbol{g} - {}^{B}\hat{\dot{\boldsymbol{\Theta}}}_{k-1|k-1} \times {}^{B}\hat{\dot{\boldsymbol{\Xi}}}_{k-1|k-1} \right) + {}^{B}\hat{\dot{\boldsymbol{\Xi}}}_{k-1|k-1} && \text{(E.3)}\\
{}^{B}\hat{\dot{\boldsymbol{\Theta}}}_{k|k-1} &= \begin{bmatrix} \dot{\boldsymbol{\Theta}}_{IMU,k-1} & {}^{B}\hat{\dot{\boldsymbol{\Theta}}}_{k-1|k-1} \end{bmatrix} \begin{bmatrix} q \\ 1-q \end{bmatrix} && \text{(E.4)}
\end{aligned}
$$

- ${}^{E}\hat{\boldsymbol{\Xi}}$ is the position of the helicopter given in $\{E\}$.
- ${}^{B}\hat{\dot{\boldsymbol{\Xi}}}$ is the translatory velocity of the helicopter given in $\{B\}$.
- ${}^{B}\hat{\ddot{\boldsymbol{\Xi}}}$ is the translatory acceleration of the helicopter given in $\{B\}$.
- ${}^{E}\hat{\boldsymbol{\Theta}}$ is the attitude of the helicopter given in $\{E\}$.
- ${}^{B}\hat{\dot{\boldsymbol{\Theta}}}$ is the angular velocity of the helicopter given $\{B\}$.
- T_s is the sampling time.
- ${}^{E}_{B}\hat{\mathcal{R}}$ is the rotation matrix converting a vector given in $\{B\}$ to a vector given in $\{E\}$.
- q is a weighting factor, which is a scalar between 0 and 1.
- The subscript e.g. $k|k-1$ is the value at time k given $k-1$.

In this appendix the rigid body model is verified by the following trial-and-error procedure.

1. A trajectory is created, and the state vector for this trajectory is written as a function of time ($\boldsymbol{x}(t)$)

2. The initial state vector $\boldsymbol{x}_0$ is noted.

3. The IMU output for the trajectory is calculated and written as a function of time ($\boldsymbol{u}(t)$).

4. $\boldsymbol{x}_0$ and $\boldsymbol{u}(t)$ is used as input to the process model. The resulting state vector is compared with the original state vector from the first step. By using a small step time (T_s = 0,01 is used), they should be equal, and if that is not the case the process model is incorrect.

13 different tests has been conducted, starting with simple one dimensional trajectories with only translatory or angular motion; then continuing to combined one-dimensional translatory and one-dimensional angular motion. Finally, trajectories of three-dimensional translatory movement combined with one-dimensional angular motion are tested. It would be too time-consuming to document it all. However, to convince the reader that the process model is correct, two of the final tests are covered here.

E.2 The Roll

The roll trajectory starts out with the helicopter in zero position ($\boldsymbol{\Xi} = \boldsymbol{0}$ and $\boldsymbol{\Theta} = \boldsymbol{0}$). While moving forward with constant velocity of 1 m/s, it performs a 360° uniform circular motion in the lateral plane. The radius of this circle is 1 m and the center is placed directly underneath the CM. After 8 seconds, the helicopter has finished the roll. Figure E.1 illustrates the full roll trajectory.

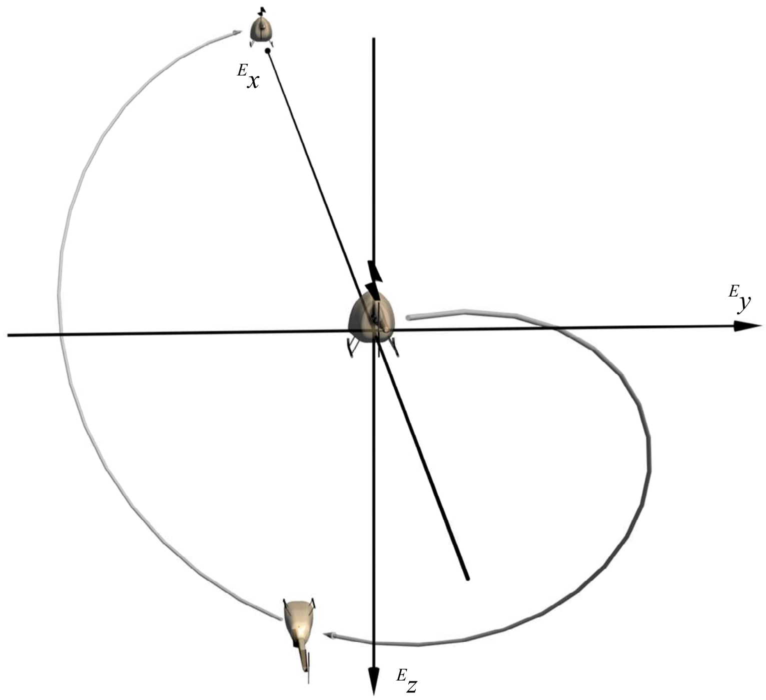


__Figure E.1:__ The trajectory of the roll motion used to test the process model. The helicopter does not get hurt in the operation.

1. The state vector function for the roll motion is given by

$$\boldsymbol{x}(t) = \begin{bmatrix} t & \sin(\frac{\pi}{4}t) & 1-\cos(\frac{\pi}{4}t) & \quad \frac{\pi}{4}t & 0 & 0 & \quad 1 & \frac{\pi}{4} & 0 & \quad \frac{\pi}{4} & 0 & 0 \end{bmatrix}^T$$

2. The initial values are

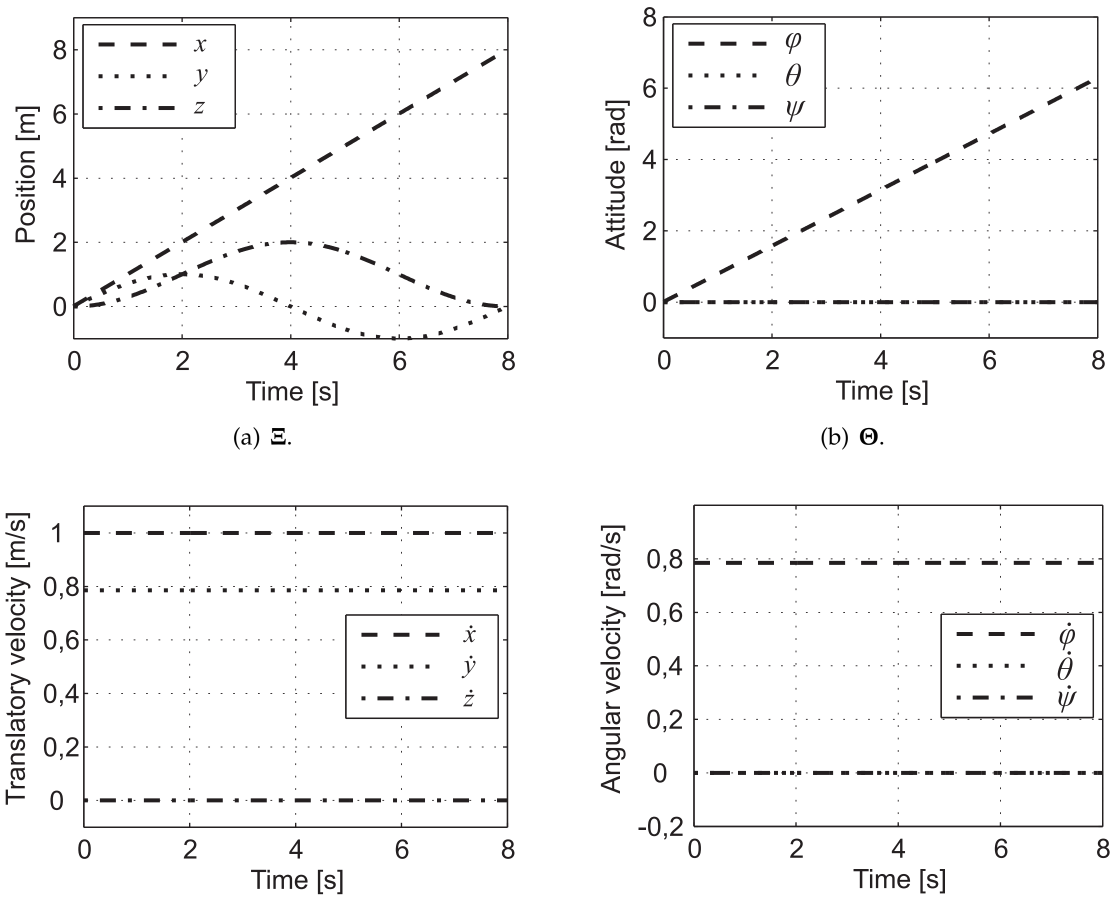
$$\boldsymbol{x}_0 = \begin{bmatrix} 0 & 0 & 0 & \quad 0 & 0 & 0 & \quad 1 & \frac{\pi}{4} & 0 & \quad \frac{\pi}{4} & 0 & 0 \end{bmatrix}^T$$

3. The IMU output is then found to be

$$\boldsymbol{u}(t) = \begin{bmatrix} 0 & -9{,}82\sin(\frac{\pi}{4}t) & \frac{\pi^2}{16} - 9{,}82\cos(\frac{\pi}{4}t) & \quad \frac{\pi}{4} & 0 & 0 \end{bmatrix}^T$$

   The sin and cos functions arise due to the gravitational acceleration, which is converted from $\{E\}$ to $\{B\}$.

4. The output of the process model is depicted as graphs in Figure E.2.



(a) $\boldsymbol{\Xi}$.

(b) $\boldsymbol{\Theta}$.

(c) $\dot{\boldsymbol{\Xi}}$.

(d) $\dot{\boldsymbol{\Theta}}$.

Figure E.2: *The state vector output for the roll motion.*

E.3 The Loop

The second trajectory of the verification is a straight loop motion. The helicopter starts out in zero position and attitude, and then while moving with constant speed in the ${}^{B}x$-axis, it revolves with constant angular speed around the ${}^{B}y$-axis. It does a 360° turn, and ends up in the same position as it started. Figure E.3 shows the loop trajectory.

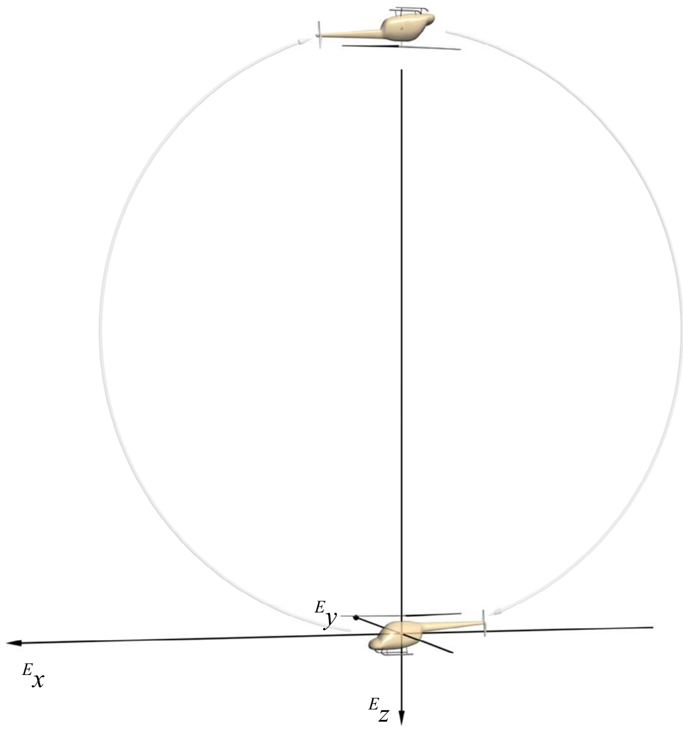


Figure E.3: *The trajectory of the loop motion used to test the process model.*

1. The state vector function for the loop motion is given by

$$\boldsymbol{x}(t) = \begin{bmatrix} \sin(\frac{\pi}{4}t) & 0 & \cos(\frac{\pi}{4}t) - 1 & & 0 & \frac{\pi}{4}t & 0 & & \frac{\pi}{4} & 0 & 0 & & 0 & \frac{\pi}{4} & 0 \end{bmatrix}^T$$

2. The initial values are

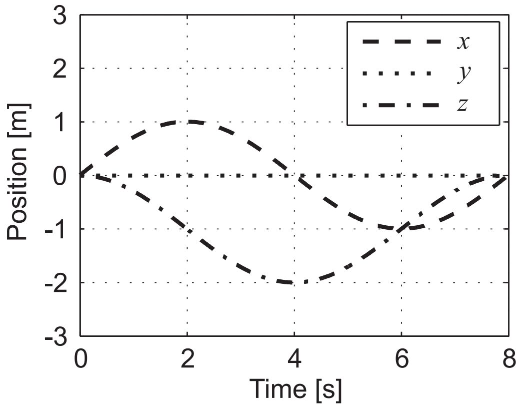
$$\boldsymbol{x}_0 = \begin{bmatrix} 0 & 0 & 0 & & 0 & 0 & 0 & & \frac{\pi}{4} & 0 & 0 & & 0 & \frac{\pi}{4} & 0 \end{bmatrix}^T$$

3. The IMU output is then found to be

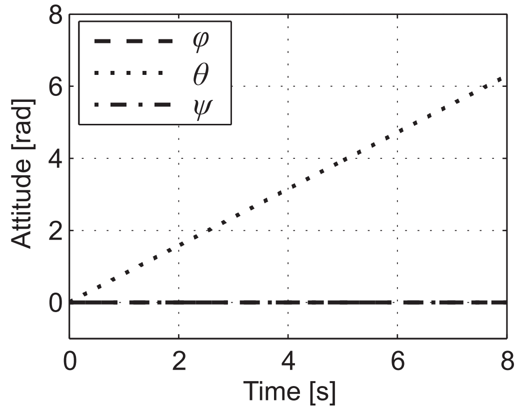
$$\boldsymbol{u}(t) = \begin{bmatrix} 9{,}82\sin(\frac{\pi}{4}t) & 0 & -\frac{\pi^2}{16} - 9{,}82\cos(\frac{\pi}{4}t) & & 0 & \frac{\pi}{4} & 0 \end{bmatrix}^T$$

   Again, the sin and cos functions arise due to the gravitational acceleration, which is converted from $\{E\}$ to $\{B\}$.

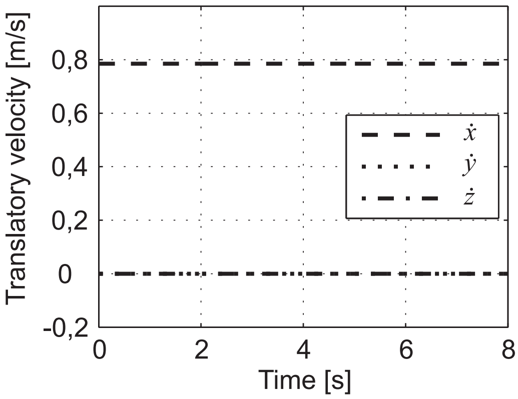
4. The output of the process model is depicted as graphs in Figure E.4.



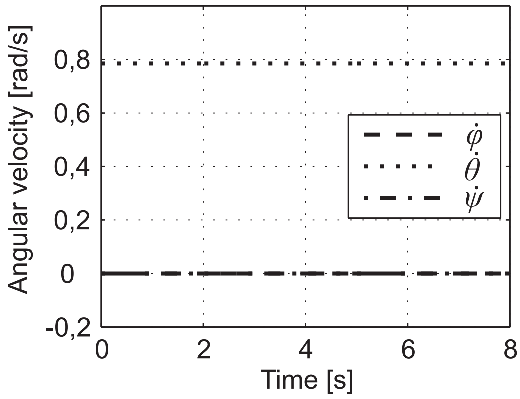
(a) $\boldsymbol{\Xi}$.



(b) $\boldsymbol{\Theta}$.



(c) $\dot{\boldsymbol{\Xi}}$.

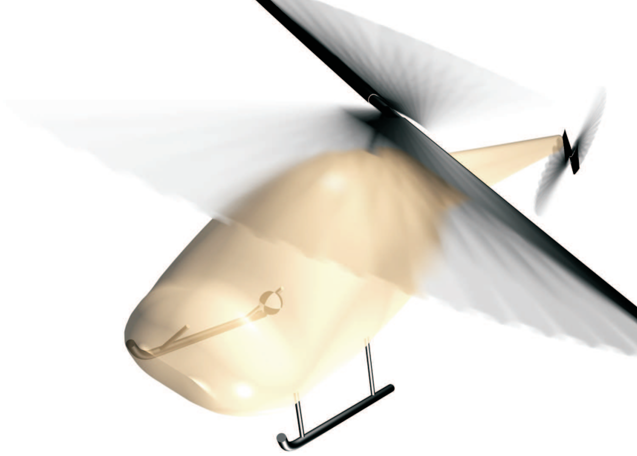


(d) $\dot{\boldsymbol{\Theta}}$.

Figure E.4: *The state vector output for the loop motion.*

E.4 Test Results

By inspection of the roll and loop graphs in the figures E.2 and E.4, it can be seen that they are in accordance with the 12 state functions given under item 1. This was the case for all the conducted tests. The conclusion is that the process model is correct.



F

LINEARIZED MODEL

In this appendix, the simplified linearized model, which in Chapter 7 is used to develop controllers on, is developed. The model is derived disregarding cross couplings. That means that four individual controllers are derived, one for altitude, one for lateral movement, one for longitudinal movement and one for yaw. All derivation in the appendix is kept in the s-domain, since this it is the way it is implemented as blocks in SIMULINK.

F.1 Altitude Model

The altitude position transfer function $H_z(s)$ is given by

$$H_z(s) = \frac{\tilde{z}}{\tilde{S}_{mr}} \quad . \tag{F.1}$$

- z is the vertical position of the helicopter given in the s-domain.
- $\tilde{S}_{mr}$ is the input to the transfer function controlling the rotational speed of the main rotor.

The working point is in hover, so the input to the main rotor resulting in steady hover is denoted by $\bar{S}_{mr}$. This is where the sum of the lift created by the main rotor and the gravitational force is 0 N. Thus, $\tilde{S}_{mr}$ is the deviation from this point, such that

$$S_{mr} = \tilde{S}_{mr} + \bar{S}_{mr} \quad . \tag{F.2}$$

S_{mr} is the input to the DC motor, which has a second order transfer function, as described in Section 5.5.12 on page 78. Denoting this transfer function as H_{mr}, we have (according to the parameters Section 5.5.12).

$$H_{mr}(s) = \frac{\Omega_{mr}}{S_{mr}} = \frac{25}{s^2 + 10s + 25} = \frac{25}{(s+5)^2}$$

$$\Updownarrow$$

$$\Omega_{mr} = \frac{25}{(s+5)^2}(\tilde{S}_{mr} + \bar{S}_{mr}) = \frac{25\tilde{S}_{mr}}{(s+5)^2} + \bar{S}_{mr} = \tilde{\Omega}_{mr} + \bar{S}_{mr} \quad . \tag{F.3}$$

Ω_{mr} is the rotational velocity of the main rotor, and the relation of Ω_{mr} and the lift thrust L, has been found in Section 5.5.14, and Figure F.1 shows that it can be approximated by a straight line, given by

$$L = K_L \Omega_{mr} + L_{offset} = K_L \tilde{\Omega}_{mr} + (K_L \bar{S}_{mr} + L_{offset}) = \tilde{L} + \bar{L} \quad , \tag{F.4}$$

- L is the lift thrust of the main rotor.
- $K_L = -0{,}126\,\mathrm{Ns/rad}$ is the gain from Ω_{mr} to $\tilde{L}$.
- $L_{offset} = 8{,}26\,\mathrm{N}$ is the intersection with the abscissa of the affine function $L(\Omega_{mr})$.
- $\tilde{L}$ is the deviation of the lift thrust from the working point value.
- $\bar{L}$ is the working point of the lift thrust function.

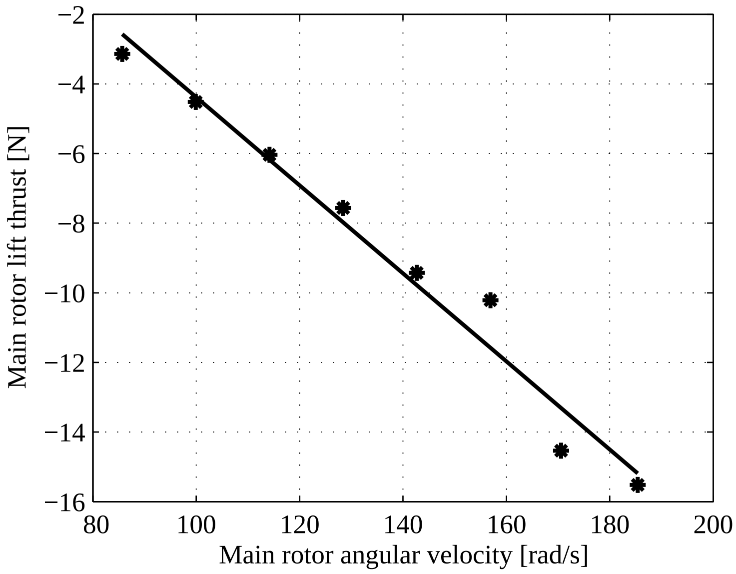


Figure F.1: *The relation between the rotational velocity of the main rotor and the lift thrust.*

Having the lift thrust, the acceleration in vertical direction of the helicopter is calculated by

$$\ddot{z} = \frac{L}{M} + g = \frac{\tilde{L} + \bar{L}}{M} + g = \frac{\tilde{L}}{M} + \left(\frac{\bar{L}}{M} + g\right) = \tilde{\ddot{z}} + \bar{\ddot{z}} \tag{F.5}$$

- $\ddot{z}$ is the acceleration of the helicopter along the ${}^{E}z$ axis. Note that we assume the helicopter to be in or close to hover, such that a small angle approximation is valid.
- M is the mass of the helicopter.
- g is the gravitational acceleration.
- $\tilde{\ddot{z}}$ is the deviation of the acceleration from g.
- $\bar{\ddot{z}}$ is the working point acceleration, which is equal to 0.

Last, the acceleration is integrated twice to get the position of the helicopter. As the helicopter is assumed to be positioned in the working point from the beginning, the initial conditions of both integrations are zero.

$$z = \frac{\ddot{z}}{s^2} = \frac{\tilde{\ddot{z}} + \bar{\ddot{z}}}{s^2} = \frac{\tilde{\ddot{z}}}{s^2} = \tilde{z} \tag{F.6}$$

Combining all the working point values in Equations (F.2) to (F.6) yields the altitude position transfer function $H_z(s)$

$$H_z(s) = \frac{\tilde{z}}{\tilde{S}_{mr}} = \frac{25K_L}{Ms^2(s+5)^2} \tag{F.7}$$

and collecting all the working point values results in the overall working point for the altitude model, which are set to 0 m

$$\bar{H}_z = \frac{K_L\bar{S}_{mr} + L_{offset}}{M} + g = 0$$
$$\Updownarrow$$
$$\bar{S}_{mr} = \frac{-gM - L_{offset}}{K_L} = 145\,\text{rad/s} \tag{F.8}$$

In other words, when the input to the main rotor is 145 rad/s, and the helicopter model is initiated in steady hover, the helicopter will stay hovering.
Figure F.2 shows a block diagram of the altitude model, H_{alt}, which corresponds to the SIMULINK implementation. Note that both the z position and z velocity is used as output, as they are needed for feedback control. The velocity is given as the signal before the last integration, and the transfer function $H_{\dot{z}}$ is found by differentiating H_z.

$$H_{\dot{z}}(s) = \frac{\tilde{\dot{z}}}{\tilde{S}_{mr}} = H_z(s)s = \frac{25K_L}{Ms(s+5)^2} \tag{F.9}$$

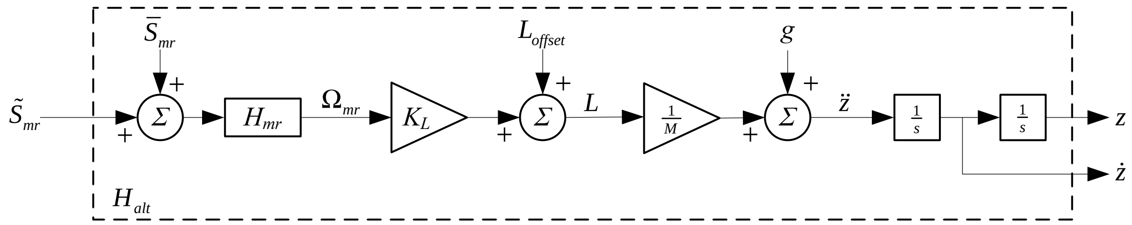


Figure F.2: *Block diagram of the altitude model, H_{alt}, which shows how it has been implemented in* SIMULINK.

F.2 The Lateral Model

Lateral motion of the helicopter is mainly caused by rolling, as this tilts the lift thrust of the main rotor. This results in a projection of the lift thrust down on the ${}^{E}y$ axis, which creates an acceleration of the helicopter.
The model is initiated with a rough assumption about the relationship between the input S_{lat} and the roll angle, ϕ, which says that the angular roll velocity $\dot{\phi}$ is proportional to S_{lat}.

$$\phi = \frac{K_{lat}}{s}S_{lat} \tag{F.10}$$

- ϕ is the roll angle of the helicopter.

- $K_{lat} = -2\,\mathrm{rad/s}$ is the DC-gain from S_{lat} to ϕ. It is negative because a negative S_{lat} results in a positive $\dot{\phi}$ (see Table 3.3 on page 22).
- S_{lat} is the control input for the lateral pitch of the main rotor.

The projection of the lift thrust vector on the ^{E}y axis is given by

$$L_y = |L|\sin(\phi) \approx gM\phi \tag{F.11}$$

- L_y is the projection of L at the y axis.
- L is the lift thrust of the main rotor.
- g is gravitational acceleration.
- M is the mass of the helicopter.

To avoid cross couplings, it is assumed that the lift thrust is always equal to $-gM$. This makes sense since the model is only valid in or close to hover. For the same reason, a small angle approximation is also applied. The acceleration of the helicopter in the y axis direction can now be calculated as

$$\ddot{y} = L_y/M \tag{F.12}$$

- $\ddot{y}$ is the acceleration of the helicopter in the y axis direction.

The position of the helicopter is now found by integrating the acceleration twice.

$$y = \frac{\ddot{y}}{s^2} \tag{F.13}$$

Combining Equations (F.10) to (F.13) yields the final transfer function for the lateral motion.

$$H_y(s) = \frac{y}{S_{lat}} = \frac{gK_{lat}}{s^3} \tag{F.14}$$

Figure F.3 shows a block diagram of the lateral model, H_{lat}, which corresponds to the SIMULINK implementation. Note that both y, $\dot{y}$, ϕ and $\dot{\phi}$ is used as output, as they are needed for state feedback. The transfer functions are

$$H_{\dot{y}}(s) = \frac{gK_{lat}}{s^2} \tag{F.15}$$

$$H_{\phi}(s) = \frac{K_{lat}}{s} \tag{F.16}$$

$$H_{\dot{\phi}}(s) = K_{lat} \tag{F.17}$$

F.3 The Longitudinal Model

The longitudinal motion of the helicopter is modelled in exactly the same way as the lateral, so we end up with the same transfer functions, with one exception: a positive pitch angle (θ) results in a negative longitudinal acceleration, such that

$$L_x = -|L|\sin(\theta) \approx -gM\theta \tag{F.18}$$

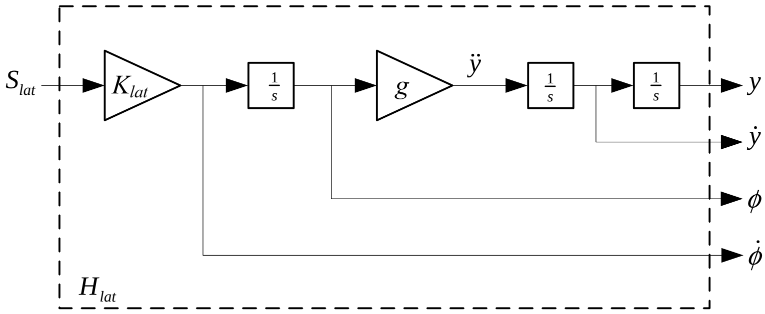


Figure F.3: *Block diagram of the lateral model, H_{lat}, which shows how it has been implemented in* SIMULINK.

- L_x is the projection of L at the x axis.
- L is the lift thrust of the main rotor.
- g is gravitational acceleration.
- M is the mass of the helicopter.

Thus, the longitudinal transfer functions are given by

$$H_x(s) = \frac{-gK_{lon}}{s^3} \tag{F.19}$$

$$H_{\dot{x}}(s) = \frac{-gK_{lon}}{s^2} \tag{F.20}$$

$$H_\theta(s) = \frac{K_{lon}}{s} \tag{F.21}$$

$$H_{\dot{\theta}}(s) = K_{lon} \tag{F.22}$$

- $H_x(s)$ is the transfer function from S_{lon} to x.
- g is the gravitational acceleration.
- $K_{lon} = -2$ rad/s is the DC gain from S_{lon} to ϕ. It is negative because a positive input yields a negative $\dot{\phi}$.
- $H_{\dot{x}}(s)$ is the transfer function from S_{lon} to $\dot{x}$.
- $H_\theta(s)$ is the transfer function from S_{lon} to θ.
- $H_{\dot{\theta}}(s)$ is the transfer function from S_{lon} to $\dot{\theta}$.

Figure F.4 shows a block diagram of the longitudinal model, H_{lon}, which corresponds to the SIMULINK implementation. Note that both x, $\dot{x}$, θ and $\dot{\theta}$ is used as output, as they are needed for state feedback.

F.4 The Yaw Model

The yaw motion is simply modelled as a DC-gain from the input to the yaw velocity.

$$H_{\dot{\psi}} = \frac{\dot{\psi}}{\dot{\psi}_{ref}} = K_{yaw} \tag{F.23}$$

- $H_{\dot{\psi}}$ is the transfer function from $\dot{\psi}_{ref}$ to $\dot{\psi}$.
- $\dot{\psi}_{ref}$ is the control input to the tail rotor pitch.

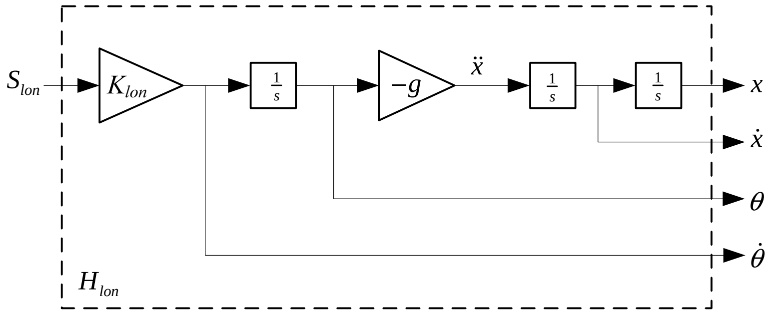


Figure F.4: *Block diagram of the longitudinal model, H_{lon}, which shows how it has been implemented in* SIMULINK.

- $K_{yaw} = 1$ is the DC gain from $\dot{\psi}_{ref}$ to ψ.

To get the yaw position, an integration with zero initial conditions is applied.

$$H_{\psi} = \frac{K_{yaw}}{s} \tag{F.24}$$

- H_{ψ} is the transfer function from $\dot{\psi}_{ref}$ to ψ.

Figure F.5 shows a block diagram of the yaw model, H_{yaw}, which corresponds to the SIMULINK implementation. Note that both $\dot{\phi}$ and ϕ is used as output, as they are needed for state feedback.

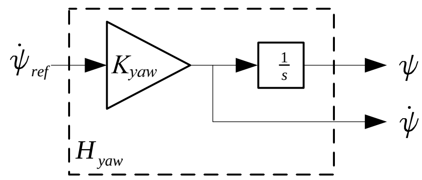


Figure F.5: *Block diagram of the yaw model, H_{yaw}, which shows how it has been implemented in* SIMULINK.

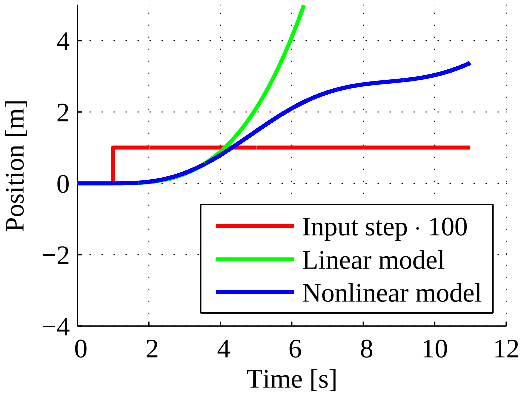
F.5 Comparison of Linear and Nonlinear Model

To verify that the linear model is behaving as expected, a series of step tests has been conducted on the linear and the non-linear model described in Chapter 5. The goal with the step test is to verify that the linear model and the non-linear model behaves roughly similar around the working point, which is hovering. Therefore, the input is calibrated such that the helicopter is hovering. This is simple for the linear model, as all input are just set to 0. For the non-linear model, it requires some more tuning of the initial input values, as hovering is an unstable equilibrium. In practice, it is done by using a temporarily designed PI-controller to stabilize it, whereafter the input values are hold steady and the controller is disabled.

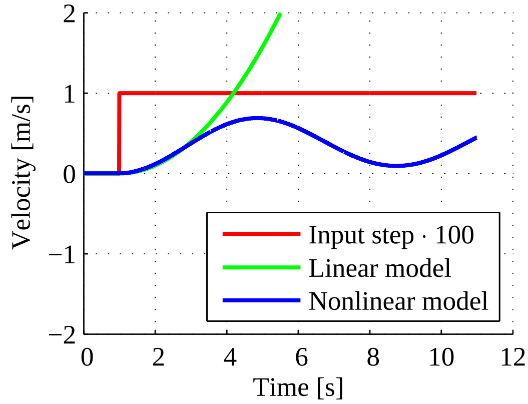
A step of suitable amplitude is now applied to each input, one at a time. The amplitude of the step is chosen such that the reaction of the helicopter is not too aggressive. As it is unstable, it requires very small steps, and by experiment it has been found that it is feasible to use

- $1\ {}^{\mathrm{rad}}/_{\mathrm{s}}$ for S_{mr}.
- 0,01 for S_{lat} and S_{lon}.
- $0{,}1\ {}^{\mathrm{rad}}/_{\mathrm{s}}$ for $\dot{\psi}_{ref}$.

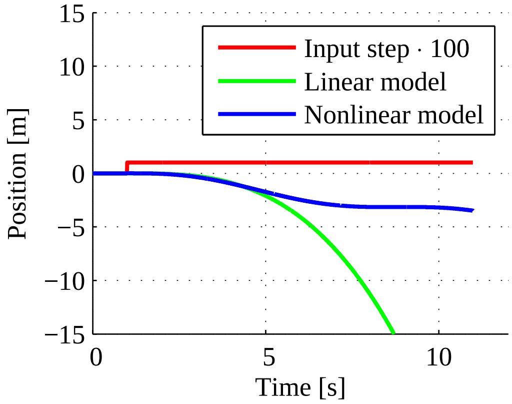
Figure F.6 and Figure F.7 show the result of the step tests. In all tests, the step is applied at time 1 s, as illustrated with a red graph in each figure. A green graph shows the step response of the linear model, and a blue graph shows the step response of the non-linear model. To be able to compare the graphs, all the step responses are illustrated with the same time axis going from 0 s to 11 s. If the helicopter is actuated in such a way that the attitude is kept within a few degrees of the working point, then the small angle approximations are valid, and the green and the blue graph should be coinciding. When applying a step this is not the case, and the helicopter will eventually move outside the small signal range. However, as can be seen on the graphs of the step responses, the linear and non-linear model should approximately follow each other for a few seconds, until the helicopter has moved outside the valid range. This is the case for all of the step responses, though $\dot{z}$ follows for a shorter period, and $\dot{\phi}$ and $\dot{\theta}$ exhibits oscillations. Regarding $\dot{z}$ the deviation results from the fact, that the wire attached to the nose is not included in the linear model. For $\dot{\phi}$ and $\dot{\theta}$ the fast oscillations occurring right after the step is due to flapping dynamics. The slow oscillations happens because of cross couplings from the lateral and longitudinal velocity, respectively. However, these deviations are not considered to be vital, and it is concluded that the linear model is feasible for developing a controller for hover.



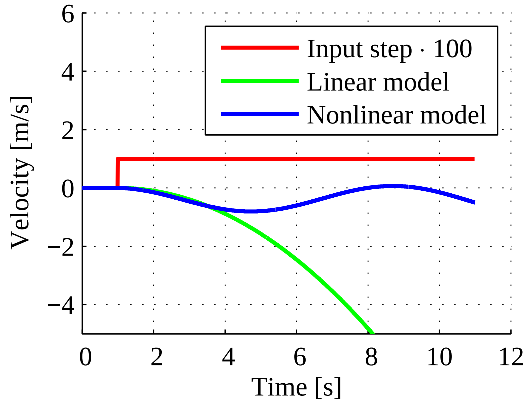
(a) Step test for x. Note that the input step is scaled by a factor 100 in order to make it visible on the current axes.



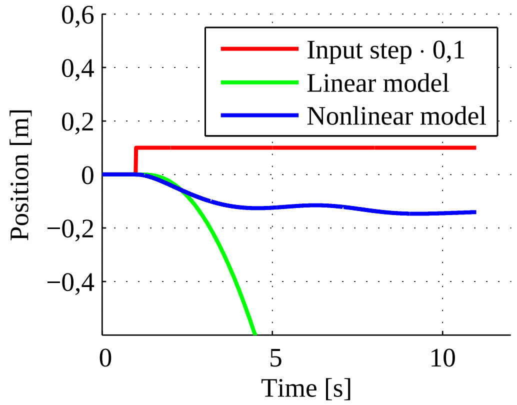
(b) Step test for $\dot{x}$. Note that the input step is scaled by a factor 100 in order to make it visible on the current axes.



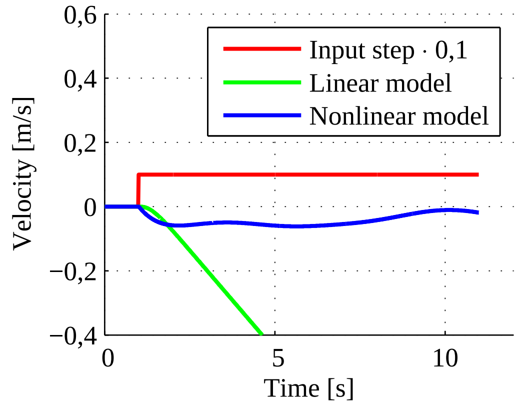
(c) Step test for y. Note that the input step is scaled by a factor 100 in order to make it visible on the current axes.



(d) Step test for $\dot{y}$. Note that the input step is scaled by a factor 100 in order to make it visible on the current axes.

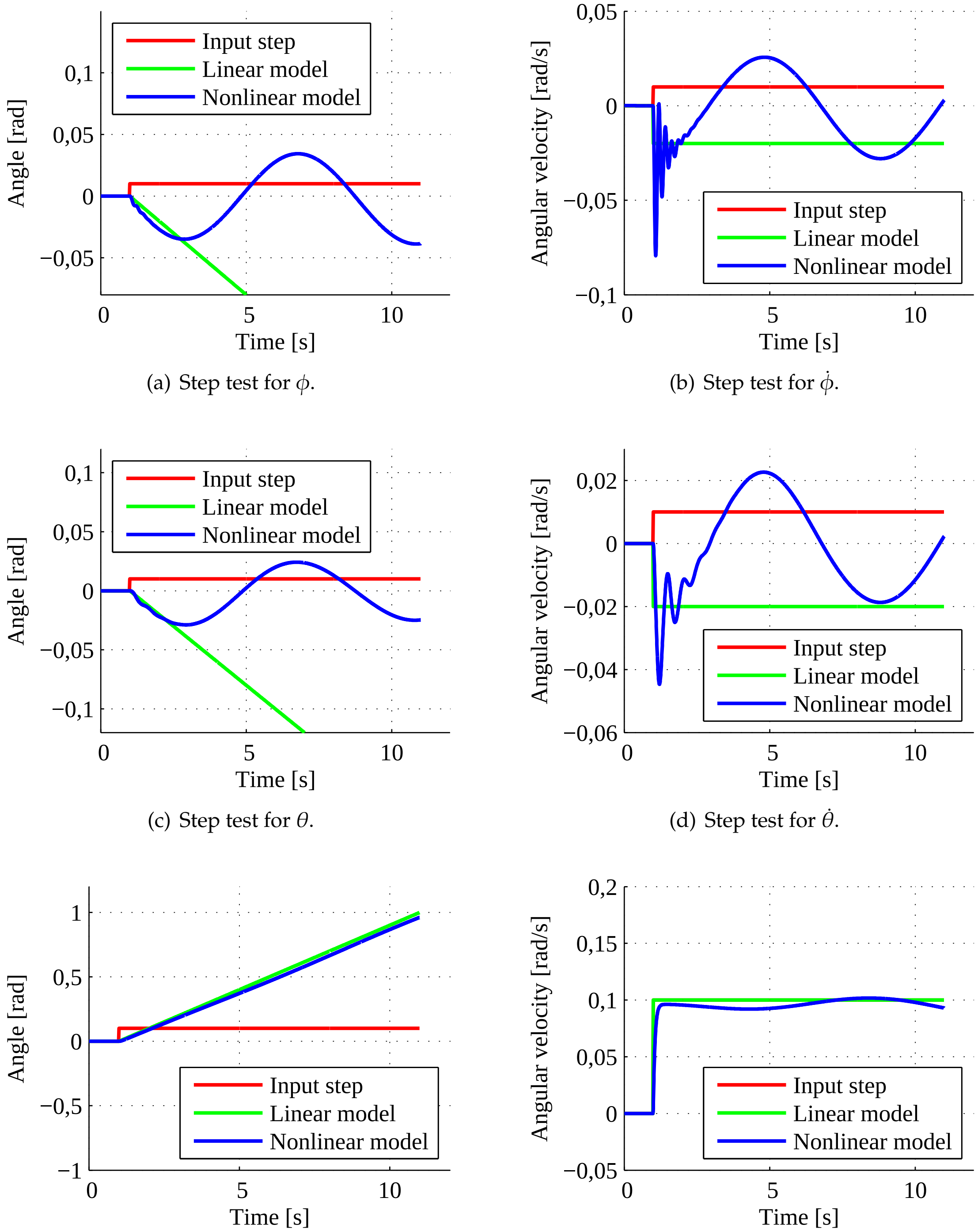


(e) Step test for z. Note that the input step is scaled by a factor 0,1 in order to make it visible on the current axes.



(f) Step test for $\dot{z}$. Note that the input step is scaled by a factor 0,1 in order to make it visible on the current axes.

Figure F.6: *Step test comparison of the position and velocity states.*



(a) Step test for ϕ.

(b) Step test for $\dot{\phi}$.

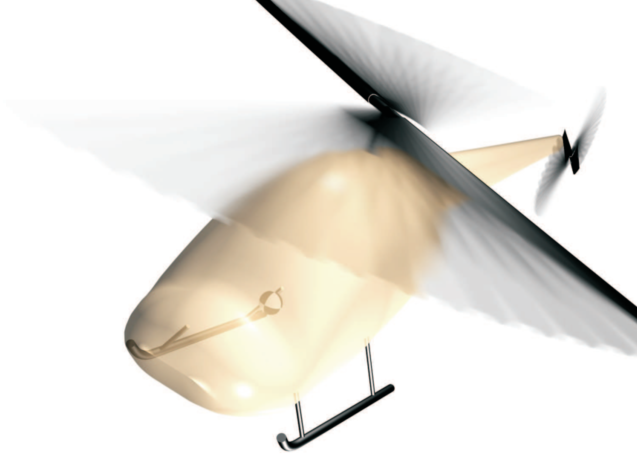
(c) Step test for θ.

(d) Step test for $\dot{\theta}$.

(e) Step test for ψ.

(f) Step test for $\dot{\psi}$.

Figure F.7: *Step test comparison of the attitude and angular velocity states.*



G

Users guide/Instruction Manual

This appendix contains a guide on how to use the system. This includes how to start up the individual parts of the system and how to check for correct operation.

G.1 Operating the IPC

This section describes how to start up and use the IPC. Further, if having trouble with the system, a troubleshooting section is also included.

G.1.1 Startup and System Check

Use the following steps to startup the IPC and check if everything works as expected. The procedure is documented in the video : [CD-ROM, 2007, `videos/IPC_init.wmv`]. Avoid wearing any red clothes, as it might "confuse" the algorithm when filming both you and the helicopter. Further, avoid removing the USB plugs for the front and side cameras when the IPC is turned on, since this, for some reason, makes the computer crash.

1. Start up
   - 1.1 Place the helicopter where the front and side camera views are coinciding. $\{B\}$ and $\{E\}$ are now coinciding. Check that all three cameras have a clear vision towards the markers.
   - 1.2 Connect the three cameras to the IPC, turn on the IPC in Windows and start MATLAB.
   - 1.3 Switch directory to `d:\hamoc\source\image_processing\`. Mex the S-functions and open the SIMULINK diagram, by typing the following in the command window:
     - `mex wallcam_Sfun.cpp`

- `mex helicam_Sfun.cpp`
- `open position_determination.mdl`

1.4 Remove the "to instrument" block, which sends the output to the CC, to avoid simulation error when the CC is not running.

2. Calibration of marker software

2.1 Open `marker_to_position.m` and adjust the camera positions given in $\{E\}$, to correspond to where you want the origin of $\{E\}$ to be. Also adjust the marker positions given in $\{B\}$. Make sure that the cameras are not tilted so that a horizontal line in the image corresponds horizontal in reality.

2.2 Start the simulation. After initialization, three video displays, from each of the cameras, should occur.

2.3 Verify that the markers are detected correct in all the three video displays.

2.4 Stop simulation again. To calibrate the helicam, enter the x and y position (x_offset and y_offset) of the floormarker position into `marker_to_position.m`.

2.5 Start the simulation again, and look at the output pixel positions from the S-functions, and move the helicopter so that the x-position for both the front and side camera are close to 320 (equal to the center of the image).

2.6 Stop simulation again. To calibrate the frontcam and sidecamb, enter the x and y position (x_offset and y_offset) of the marker positions into `marker_to_position.m`.

3. Testing the marker location algorithm

3.1 Start the simulation and verify that the markers are still found.

3.2 Look at the MATLAB prompt to see if nothing is written to screen (this should indicate that the marker location works properly).

3.3 Try to walk in between the cameras and the markers. The markers should stay at the same place until you move again (note that when standing in front of the marker, warning messages about not being able to find the marker, should be printed to the MATLAB prompt).

3.4 Move the helicopter (slowly) beyond the edge of one or both of the cameras, to see if the marker stays at the edge of the image (and does not find an arbitrary point which looks a bit like a marker.) until the marker returns into view.

3.5 Move the helicopter around, both far away from and near the cameras, to verify that the markers are found everywhere. This test shows if the calculation of the size of the markers are correct.

4. Testing the position and attitude calculation

4.1 Stop the simulation and remove the video displays to speed up simulation, which makes it possible to move the helicopter a bit faster and still not get warning messages.

4.2 Start the simulation again and verify that the position error estimate (PEE) is close to zero (should at least be below 0,05 m).

4.3 Open the position scope and move the helicopter in each direction to verify that the algorithm works as desired.

4.4 Place the helicopter at the origin of $\{E\}$ again, and verify the same way that calculation of ϕ and θ is correct as well.

4.5 Check if the error estimate have stayed below a satisfying limit during the whole simulation.

4.6 Stop the simulation.

5. Reinsert the "to instrument" to make ready to send data to the CC.

When the above test has been performed, there should be some images in the directory. These are saved each time the algorithm was not able to find the marker. For example when walking in front of the marker. Furthermore, the following variables are saved to workspace: `pixel_front`, `pixel_side`, `pixel_hcam` and `marker_state`.

G.1.2 Troubleshooting on IPC

The simulation will not finish initializing.	Try unplug and replug the USB connector for the helicam at the helicopter side. Close all explorer windows which displays the web camera images
The update rate from the IPC is slow	The marker might move too fast. Try increasing the ROI in the image processing software (this has to be don both in `camera_hough_Sfun.cpp` and `helicamprocessing.cpp`).
There is warning messages about not finding the right background colour.	If it is dark or rainy outside, the colours are not the same because of the missing sunlight. Try turning the floodlight on or set a manual background colour in the code.
The position error is more than 0,05 m most of the time.	The cameras are not calibrated correct in `marker_to_position.m`.
The marker is not found	Try looking at the saved images to find the error. Maybe adjust the PixelMax condition in `camera_hough_Sfun.cpp`

G.2 CC Startup

This section describes how to use the CC. How to initialize the system and the controllers, and how to check if everything is working as intended. The procedure is documented in the video : [CD-ROM, 2007, `videos/CC_init.wmv`]

G.2.1 Startup and System Check on the CC

This section describes the steps to start up the CC and check if everything is working as intended.

1. Start up

1.1 Turn on the CC in Ubuntu (user name and password are "hamoc"), start MATLAB and switch directory to `/home/hamoc/source/integration`.

1.2 Initialize system by running `mex_system.m`. This compiles the S-functions for the servoboard input, servoboard output, IMU output and input from the IPC. Further the parameters for the controller and a trimming condition are set.

1.3 Open the SIMULINK interface to the system: `open system_integration.mdl`.

1.4 Enable the "Pseudo Realtime" by using the switch in the top left corner.

1.5 Disconnect the main motor from the main rotor controller on the helicopter (to avoid that the main rotor starts to rotate uncontrollably, it is first checked if the system works).

1.6 Connect all the cables to the helicopter: servoboard, IMU, camera and power cable.

1.7 Turn on the remote control and switch on power to the helicopter. Now the pololu board should light yellow to indicate that it is ready to receive data.

2. Verify operation of the servoboard input and output.

2.1 Start the simulation. The yellow diode should turn off to indicate that the pololu board is ready.

2.2 Check if remote control is working and if the output on the "Servoboard output scope" corresponds to the input.

2.3 Check the MATLAB prompt. The sample frequency for the servoboard and the IMU should be shown, and no other messages should be printed. The sample frequencies should be close to 50 Hz and 100 Hz respectively.

2.4 Switch to manual control on the "Autonomous/Manual" switch.

2.5 Switch to autonomous on the remote control and the red diode on the servoboard should turn on.

2.6 Check if it is possible to control the servos from the CC, by using the manual input. Check again if the input and output matches.

2.7 Stop the simulation.

2.8 Switch back to manual on the remote and disconnect the power to the helicopter.

2.9 Reconnect the main rotor, and then reconnect the power to the helicopter.

2.10 Start the simulation again and spin up the main rotor to around 50 rad/s (the helicopter starts to lift off around 100 rad/s).

2.11 A display is showing the difference between the input velocity from the remote control and the input velocity the CC wants to give. Check that the difference is between $-10 - 10$ (It is important always to do this, especially when flying, to avoid sudden jumps in main rotor input). When this is the case, then switch to autonomous and verify that it is possible to control the main rotor rotation velocity, and that the output from the servoboard is still correct.

2.12 Switch back to manual control on the remote. Remember to check the difference again.

2.13 Slowly, decrease the rotational speed of the main rotor until it has stopped.

3. Verify that the IMU works
   3.1 Disconnect the power from the helicopter.
   3.2 Disconnect the main rotor from the main rotor controller again, and switch on power to the helicopter.
   3.3 Move the helicopter in different directions to see variations in the accelerometer data on the IMU scope.
   3.4 Turn the helicopter around the three axes to see if the rotation is measured by the IMU.
4. Check data from IPC and kalman filter.
   4.1 Turn on the simulation on the IPC (note that the simulation on the CC must be running first to avoid errors on the serial connection between the two computers).
   4.2 Try moving the helicopter around while watching if the state estimate, on the state scope, follows the movements.
   4.3 Try changing the attitude and watch if the state estimate is correct here also.
5. Verify that the controller is working correct.
   5.1 Start the simulation on both the IPC and CC and switch to autonomous mode on the simulation.
   5.2 Hold the helicopter around the origin of $\{E\}$ and move the helicopter in different directions while watching the servoboard input scope, which is the control signals.
   5.3 Verify if the control input is correct (note that there is an offset of the control signals to compensate for the helicopter trimming, and when flying manually the servo inputs has a peak-peak value of maximum 0,2):
      - Lift the helicopter $\Rightarrow$ lower S_{mr}
      - Move the helicopter to the right $\Rightarrow$ positive S_{lat}
      - Move the helicopter to the forwards $\Rightarrow$ negative S_{lon}
   5.4 stop the simulation on the CC and the IPC.
   5.5 turn off the helicopter and connect the main rotor again.

If all of the above works as intended, then the whole system should be working.

G.2.2 Troubleshooting on the CC

The kalman filter estimates a constant velocity.	There might be a small offset error on the attitude estimate, this results in the acceleration vector not pointing exactly the right way, and therefore the Kalman filter thinks an acceleration is experienced. Try adjusting the noise matrices in the Kalman filter to trust the attitude estimates from the IPC less.

G.3 Testing Autonomous Controllers

Calibrate the trimming conditions first by watching the output from a manual flight.
When flying autonomously, remember to switch to autonomous mode in the SIMULINK diagram before switching in the air.
Pres ctrl-s to save the test data to a file

Helicopter Aided Mapping Of Crops (HAMOC) is a project, which aims at making a small scale electric helicopter able to obtain imagery of a crop field. This report considers the first steps towards this, by the use a Corona 120 electric helicopter. The project goal is to make the helicopter hover autonomously in the laboratory.

The project has been divided into four main parts; hardware implementation, modelling, state estimation, and control development. Regarding hardware, it has been chosen to control the helicopter by an external computer, and use an external power supply. The computer is interfaced to a servoboard on the helicopter by a serial connection. An existing helicopter model has been described, adapted to the Corona 120 helicopter, and the parameters have been determined. The 12 rigid body states of the helicopter have been estimated using image processing of camera data and an inertial measurement unit (IMU), which as fused in an extended kalman Filter. Four decoupled PID controllers have been developed to control the helicopter (for the z, y and x axes, and for the yaw angle, respectively). Hereafter, more advanced controllers have been researched.

The hardware works as expected, and it is possible to control the helicopter from the external computer. The developed controllers are able to control the nonlinear model in a simulation. At the end of the project period, autonomous control of the altitude of the real helicopter has been reached, and preliminary tests of a lateral controller have also been done. A helicopter crash has damaged some of the hardware, and has thus prevented further tests of the horizontal controllers, but it is expected that autonomous hover is close to be obtained. It is believed that the developed subsystems form a solid basis for further work on the project.