AIRCRAFT SYSTEM SIMULATION FOR PRELIMINARY DESIGN

Petter Krus, Robert Braun, Peter Nordin, and Björn Eriksson
Linköping University, SE-58183 Linköping, Sweden
petter.krus@liu.se

Keywords: *aircraft conceptual design, system modeling, mission simulation*

Abstract

Developments in computational hardware and simulation software have come to a point where it is possible to use whole mission simulation in a framework for conceptual/preliminary design. This paper is about the implementation of full system simulation software for conceptual/preliminary aircraft design. It is based on the new Hopsan NG simulation package, developed at the Linköping University. The Hopsan NG software is implemented in C++. Hopsan NG is the first simulation software that has support for multi-core simulation for high speed simulation of multi domain systems.

In this paper this is demonstrated on a flight simulation model with subsystems, such as control surface actuators.

1 Introduction

Traditionally aircraft conceptual design involves very few aspects of system design. There is, however, a great advantage if system subsystem design also can be involved early on. There are several reasons for that. Modern aircraft are very compact so system installation is important to take into consideration, and this cannot properly be done unless there is a preliminary design of the systems. Furthermore, energy efficiency and the impact of subsystem on the energy efficiency of the whole aircraft are becoming important in order to select the proper subsystem concept. The aim here is to allow the whole aircraft to be simulated with its subsystem as early in preliminary design as possible, and then use this model to develop the design further. In this way the simulation model becomes a point of convergence for the aircraft conceptual design as well as the conceptual design of the subsystem as well as a tool for further development in preliminary design.

Using simulation performance characteristics at the whole aircraft level can be evaluated very straightforwardly, and things like trim drag are accounted for as a byproduct. Furthermore, as the design progress into sub system design, systems such as actuation systems, fuel systems etc, can also be included and verified together in the actual flight profile. Using simulation models of whole aircraft with subsystems, it is then possible to do design analysis, e.g. sensitivity analysis and trade of analysis, as well as design optimization. In this paper this is demonstrated on a flight simulation model with subsystems, such as control surface actuators.

In recent years there has been a lot of development in methods and tools suitable for simulation of systems. One example is the Hopsan-NG simulation package developed at Linköping University. This means for example that it is possible to model basic aircraft systems, such as hydraulic system, air system and fuel system, much more efficiently than before and that a lot of systems can even be simulated in real time or faster than real time. In this paper it is shown how subsystem models also can be coupled to models of flight dynamics, propulsion, and flight control, to produce a more complete aircraft system model. Such a model can be used already in preliminary design, thus allowing the preliminary subsystem designs to be designed concurrently with the aircraft layout.

One problem when dealing with large complex systems, however, is that most simulation packages rely on centralized integration algorithms that scale rather poorly with respect to system size. For large-scale systems it is an advantage if the system can be partitioned in such a way that the parts can be evaluated with only a minimum of interaction. The reason that centralized solver dominates is most likely, that until recently, the typical system simulation has been of a moderate size. In this paper, distributed solvers with linear scaling properties is used, which means that simulation speeds, orders of magnitude higher than real time, can be achieved for system simulation Furthermore, development in single processor performance is leveling out. On the other hand, multi-core architectures have become the norm. However, using conventional simulation software, the simulation can only use one core for the simulation, thus not exploiting that potential. The distributed modeling approach used in here, however, makes it intrinsically suitable for multi-core simulation, and Hopsan already has that capability implemented.

2 Distributed modelling

A very suitable method for modelling and simulation of large complex dynamic systems is represented by distributed modelling using transmission line elements. The origin of this concept goes back at least to Auslander 1968 [1] who first introduced transmission lines (or bi-lateral delay lines). This method evolves naturally for calculation of pressures when pipelines are modelled with distributed parameters. This approach was adopted for simulation of fluid power systems with long lines in the HYTRAN program [2]. already in the seventies. The method can be generalised to both mechanical and electrical systems.

A related method is the transmission line modelling method (TLM) presented by Johns and O'Brien (Ref. [2]) for simulation of electrical networks.

Johns and O'Brien pointed out that an important aspect of modelling using transmission line elements is that most of the numerical errors introduced by an ordinary solver are avoided. The errors made due to the introduction of transmission line elements, are better described as modelling errors.

An attractive feature with this is that laws of conservation of mass and energy still hold for the solution, since there always exist a plausible physical system for the model although the line lengths may vary compared to the original system. This also implies that the user may tolerate a larger numerical error since, generally, quite large modelling errors are present anyway (errors of the order of 10% are generally considered acceptable from an engineering point of view).

A key feature of the transmission line is the finite signal propagation speed (speed of sound) of the signal travelling through the line. This means that events at one component do not affect another immediately. Using this approach the subsystems can be solved independently in each time step in contrast to using conventional solver. The implementation of the numerical solver of differential algebraic equations can therefore be implemented in the subsystem rather than at a central level. This approach is sometimes referred to as distributed modelling for several reasons. The simulation of waves in transmission lines means that distributed parameters are used in the lines, secondly it allows for distributed solver of differential algebraic equations in component and subsystems. There is also the possibility of using distributed processing by allocating subsystems to different processors.

In this way the system is divided into subsystems that generally are of limited size and the very robust method for DAE:s described earlier, can be used.

2.1 The unit transmission line

In transmission line modelling the basic dynamic element is the unit transmission line. In the Hopsan package this is used to connect different components to each other. In the general case it can be used to model both capacitances and inductances. In the Hopsan-package, however, it is used only to represent capacitances (oil volumes and mechanical springs).

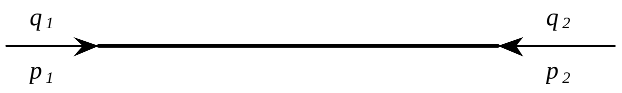


Figure 1. Transmission line

The complete set of equations that describes a lossless transmission line are:

$$p_1(t) = p_2(t-T) + Z_c q_1(t) + Z_c q_2(t-T) \quad (1)$$

$$p_2(t) = p_1(t-T) + Z_c q_2(t) + Z_c q_1(t-T) \quad (2)$$

Here Z_c is the characteristic impedance of the line, p and q are pressures and flows respectively. T is the time delay in the line. Note that the main property of these equations is the time delay they introduce in the communication between the ends.

Introducing

$$c_1(t) = p_2(t-T) + Z_c q_2(t-T) \quad (3)$$

$$c_2(t) = p_1(t-T) + Z_c q_1(t-T) \quad (4)$$

Here c is the wave variables that represent information that has been transmitted from the other side of the transmission line. With these, the following set of equations is obtained.

$$p_1(t) = c_2(t) + Z_c q_1(t) \quad (5)$$

$$p_2(t) = c_1(t) + Z_c q_2(t) \quad (6)$$

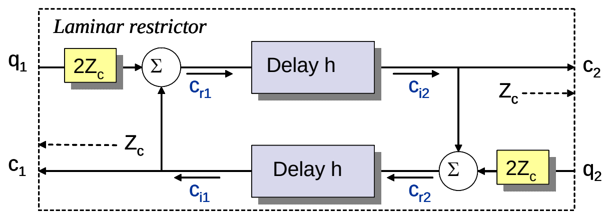


Figure 2. Block diagram of transmission line.

With this method the system can be partitioned since there is no direct communication between the two sides of the transmission line, there is always a time delay that can be used to partition the model. This also means that it is possible to use it for parallel simulation as in [15], [5], and [4].

An interesting observation is found if c_2 in Eq. (18) is substituted with Eq. (19) and the outlet at 2 is blocked.

$$p_2(t) = p_2(t-2T) + Z_c[q_1(t-2T) + q_1(T)] \quad (7)$$

Compared to the trapezoidal method for integration (h is the time step).

$$y_{h+t} = y_t + \frac{1}{2} h\left(f(u_t, t) + f(u_{h+t}, h+t)\right) \quad (8)$$

where

$$\dot{y} = f(u, t) \quad (9)$$

These equations are the same if $T = h/2$

The relationship between flow entering a volume and the pressure can be written as:

$$\dot{p} = \frac{q}{C} \quad (10)$$

where C is the capacitance. Identification yields

$$Z_c = \frac{h}{C} \quad (11)$$

The implication of this is that if we use the trapezoidal method to integrate pressure in a volume (capacitance) between two components, this corresponds to introducing a short pipe instead of a pure capacitance. This can also be extended to other domains such as mechanical and electric.

3 Differential algebraic systems

In order to solve the dynamics of the individual components and subsystems any type of solver can be used. A general approach to represent a system is to represent it as a differential algebraic system. This also allows for algebraic loops.

$$F(x, \dot{x}, t) = 0 \quad (12)$$

where x is the variable vector.

However, Eq. (1) implies that the system essentially has to be written in state space form, something that may be considered as too limited. Many relationships are usually given in transfer function form, which makes it more natural to allow for higher derivatives. The system can then instead be expressed as

$$F\left(y, \frac{dy}{dt}, \frac{d^2y}{dt^2}, \ldots, \frac{d^n y}{dt^n}, t\right) = 0 \qquad (13)$$

This also has the advantage that the variable vector is reduced, since y is shorter than x, y contains a subset of the states in x. It should, however, be pointed out that it is only possible to impose strong non-linearities (such as limitations on the state variables) represented in the y vector. Also all variables that are of any interest must be included in the y vector, otherwise they will not be computed explicitly.

In order to solve the dynamic part of the system in a numerically stable way, the trapezoidal rule can be used. Using the trapezoidal rule (or bilinear transform) the time differential is solved as:

$$x(h+t) = x(t) + \frac{1}{2}h(x(t) + x(h+t)) \qquad (14)$$

A more effective way of using the trapezoidal rule is to reformulate it in the form known as the bilinear transform.

$$\frac{d}{dt} = \frac{2(1-q^{-1})}{h(1+q^{-1})} \qquad (15)$$

where q in this context represents the time displacement operator such that:

$$qy = y(h+t) \qquad (16)$$

Using the bilinear transform in Eq. (4) means that it can be rewritten as a function G of y and old states.

$$G(y(t), y(t-h), \ldots, y(t-nh)) = 0 \qquad (17)$$

When solving the system all the old values y(t-h)... y(t-n h) can be regarded as constants since they have already been established in previous time steps. It is therefore rewritten as:

$$G(y(t), t) = 0 \qquad (18)$$

In order to solve this system of equations in a numerically stable way, the Jacobian matrix is needed, which is defined as:

$$J_{\mathrm{ijk}} = \frac{\partial G_i(y_k(t))}{\partial y_j} \qquad (19)$$

The equation can then be solved numerically using Newton-Raphson iteration.

$$y_{k+1}(t) = y_k(t) - J_k(t)^{-1} G(y_k(t)) \qquad (20)$$

Since an iterative procedure is used, there is a potential for performance loss due to the number of iteration needed to solve the system. However, since the values from the previous time step can be used as start values.

$$\begin{aligned} y_0(t) &= y(t-h) \\ J_0(t) &= J(t-h) \end{aligned} \qquad (21)$$

If the system is linear, the system can be solved in only one iteration, and it is usually sufficient with only one iteration even for non-linear systems, especially if a small time step is used. There are, however, situations when input signals changes suddenly, e.g. a valve is changed step wise during one time step that requires more than one iteration. In practice, however, it has been found that two iterations increase the tolerance against non-linearities dramatically, while a further increase to three iterations gives only minor improvement. Two iterations have therefore been found to be something near to an optimum for almost all situations. For implementation it is better to use LU-decomposition rather than using the matrix inverse of the Jacobian.

Provided the system is reasonably linear (slow variation of J, Eq. (9) is an A-stable method. However, in reality, rather large variations of J can be tolerated. Even pure discontinuities can also be handled satisfactory using the above approach, when fixed-time step is used (as in real-time simulation).

Eq. (9) also illustrates a dilemma associated with all numerically stable methods. They need knowledge of the Jacobian, and if the system is stiff and highly non-linear this must be updated (and inverted) very often. We also realize that the computational burden is much more than linearly dependent on the size of the system. This makes these methods unsuitable for large problems. Eq. (9) is, however, very effective for solving small systems which makes it very suitable for solving subsystems.

Instead of using equation (9) directly it is wise to replace the inverse of the Jacobian by using LU-decomposition instead, there are also a few other actions that can be done in order to further enhance the efficiency of the solver.

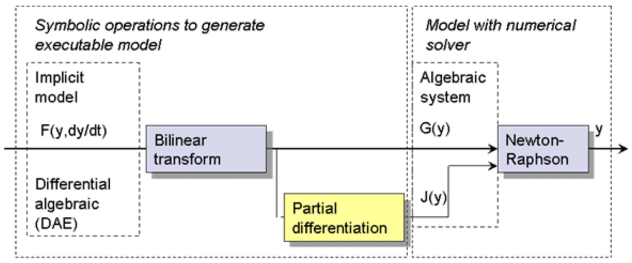


Figure 3. Scheme of transformations

Figure 3. shows a scheme of the transformations involved. The differential algebraic system DAE is transformed into time discrete form using bilinear transform. This method needs the Jacobian of the system. This can, however, be obtained analytically using symbolic math packages such as Mathematica, and a model generator has been written in the Mathematica symbolic manipulating language called Compgen, which takes the acausal component description similar to Modelica, [8][12], and transforms it into a component model implementation in C++ that can be used for simulation. This approach is described in more detail in Ref. [5] (although for generating Fortran models for the old Hopsan package). Work is also progressing to integrate this modeling capability based on Modelica and free software directly into the Hopsan package.

Packages for symbolic math can also be used to facilitate the modelling work in other ways, so that the user can concentrate on formulating the equations rather than solving them. The 6DOF-flight dynamics model is an ideal example since it involves a lot of co-ordinate transforms that can be set up entirely in the Mathematica environment. Using the Compgen tool a model for the Hopsan-NG simulation package can then be generated.

4 Modelling and simulation of an aircraft

As an example a small unmanned jet power aircraft is simulated. The control surfaces are powered by conventional hydraulic servo actuators. There is an attitude control unit that is fed by an external altitude controller. There is a fully non-linear six degree of freedom flight dynamics model with a non-linear aerodynamic model.

4 Flight dynamics and aerodynamic model

The flight dynamics model is here based on a 6 degree of freedom rigid body model that is connecteted to a aerodynamic model. The aerodynamic model can have different number of wings, with an arbitrary number of control surfaces, and a body with its characteristics. Alternatively all wings can be lumped into one wing if data for a whole aircraft is available. However, for this case the stall characteristics will not be correct, since there is only one lift curve for the whole configuration. The control surfaces are modeled both with a linear increase of lift force with deflection and the corresponding increase in induced drag. There is also a cross coupling effect of drag for control surfaces on the same wing e.g. ailerons and flaps.

Data for the aerodynamic model can be obtained in the early design phases from panel codes and/or handbook formulas, or some panel code. It is here based on a static version of the model presented in [9], although the unsteady effects can of course also be included.

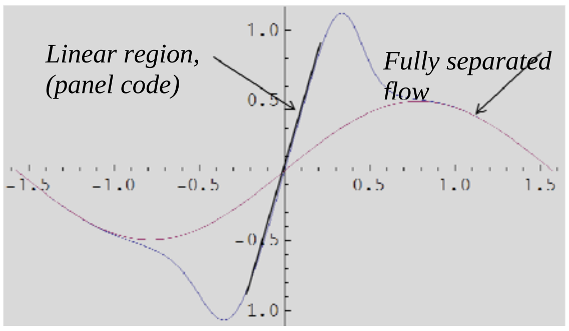


Figure 4. Non-linear aerodynamic model.

In this way effect of trim drag on range is automatically included, and the effect of reduced weight as fuel is consumed.

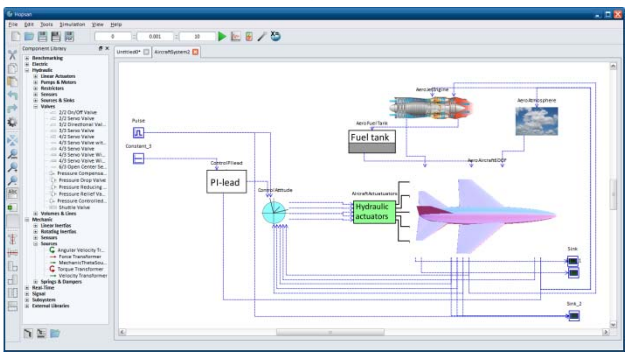


Figure 5. Screen shot of Aircraft system simulation model.

The connection between the actuators and the aircraft control surfaces are through transmission line elements that models the aerodynamic stiffness of the control surfaces. The moment at the actuator link for the elevator, can be calculated as:

$$M_1(t) = c_\delta(t-h) + Z_{c\delta}\dot{\delta}(t) \qquad (22)$$

$$c_\delta(t) = M_2(\delta,\theta,\alpha,q) \qquad (23)$$

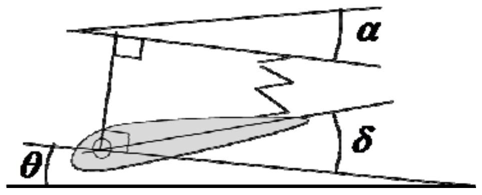


Figure 6. Model of aero load on rudder as a (non-linear) spring.

Here M_2 is the moment from the aerodynamic forces, which is a function of surface angle δ, aircraft pitch angle θ, and angle of attack α, and dynamic pressure *q*. M_1 is the corresponding moment at the actuator hinge, Ideally they are the same, but through the separation with the transmission line element they becomes slightly different. M_2 is a non-linear function and hence not a purely linear spring. The characteristic impedance is calculated as

$$Z_{c\delta} = hk_\delta \qquad (24)$$

where the linearized aerodynamic stiffness and is calculated as

$$k_\delta = \frac{\partial M_2}{\partial \delta} \qquad (25)$$

Note that this linearization is only used for the numerical solver, and does not affect the steady state value of the moment.

4.1 Flight control

For simulation in preliminary design of an unmanned aircraft the flight control system is not defined yet. However, in order to perform a flight simulation it is necessary to have some control system in and/or flight control system, or to represent the pilot in case of a manned aircraft. The requirement on this is that it should be easy to set up while still give reasonable performance to evaluate the aircraft at the level of detail present in preliminary design.

The flight control system is divided into an attitude control system and an attitude control system. The attitude control system is cascaded with the attitude control system. From the point of view of the attitude control system the plant will be an integral action system as the altitude becomes the integral of the attitude time speed (for small angles), in the low frequence region. For this kind of system there exist a very simple controller that is based on a PI controller but with a lead filter in the feedback path. This is here referred to as a *PI-lead* controller. This produce a pole placement controller with only real poles it is then only necessary to have the speed as a design parameter, which is known.

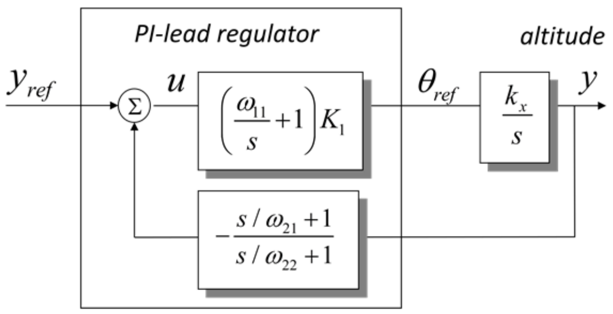

PI-lead regulator
altitude
y_{ref}
u
Σ
$\left(\frac{\omega_{11}}{s}+1\right)K_1$
θ_{ref}
$\frac{k_x}{s}$
y
$-\frac{s/\omega_{21}+1}{s/\omega_{22}+1}$

Figure 7. Altitude controller

The parameters are set as:

$$
\begin{aligned}
K_1 &= \omega_a / k_x \\
\omega_{11} &= \omega_a \\
\omega_{21} &= \omega_a / 2 \\
\omega_{22} &= \omega_a
\end{aligned}
\tag{26}
$$

This yield the closed loop response as:

$$y = \frac{1}{\frac{s}{\omega_a} + 1} y_{ref} \tag{27}$$

Furthermore, for small attitude angles:

$$h = \frac{v}{s} \theta \tag{28}$$

Which means that

$k_x = v$

However, in order to avoid very high gains at low speeds a minimum value of k_x is set.

Furthermore, to have realistic altitude changes, it is useful to put limitations on the requested pitch angle, both in positive and negative direction.

The attitude controller is defined as:

$$
\begin{aligned}
u_{ailerons} &= K_v (K_\phi \varepsilon_\phi + \text{limit}[K_{\phi\psi} \varepsilon_\psi, -\phi_{\max}, \phi_{\max}]) \\
u_{elevator} &= K_v K_{elev} \varepsilon_\theta \\
u_{rud} &= K_v (K_{rud}(-\beta) - K_{d\beta} R_b)
\end{aligned}
\tag{29}
$$

where ε represents the error in the respective attitude angles. R_b is the rotational speed around the vehicle vertical axis that is used for yaw damping. Of course damping terms on the other axis can also be introduced if needed, such as for the pitch angle if the aircraft is an unstable configuration. Since the gain of the plant (the aircraft) varies with the dynamic pressure. K_v is a speed dependent term that is defined as:

$$K_v = \frac{U_0^2}{U_0^2 + v^2} \tag{30}$$

where U_0 is a low speed that prevents the gain from becoming very large.

4.2 Subsystem modelling

In order to deal with complexity it is necessary to be able to model in a hierarchical way. This is done by introducing subsystems. In this example the hydraulic actuation system is one subsystem representing the control surface actuators and the hydraulic supply unit. The hydraulic supply system is represented by the pressure controlled pump, a pressurized tank (represented by an asymmetric piston in the model), and an accumulator.

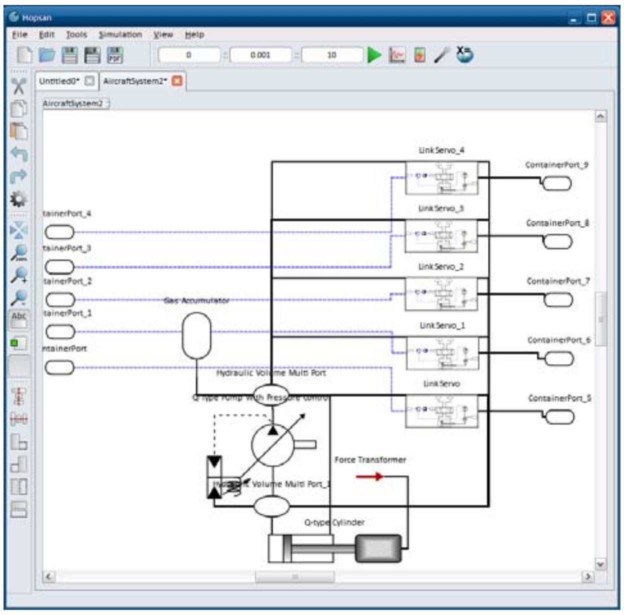


Figure 8. Model of the hydraulic system with a constant pressure controlled pump with pressurized tank (piston) and accumulator.

The servo actuators are also modeled as subsystems with servo valve, piston and linkage.

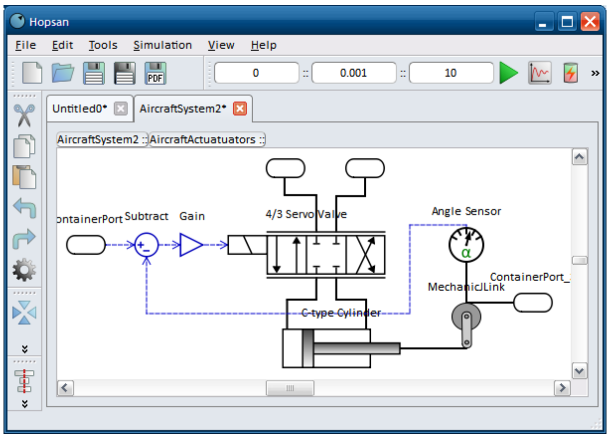


Figure 9. Model of the servo actuator subsystem.

The redundancy has not been modeled here, but it is very straightforward to include as well.

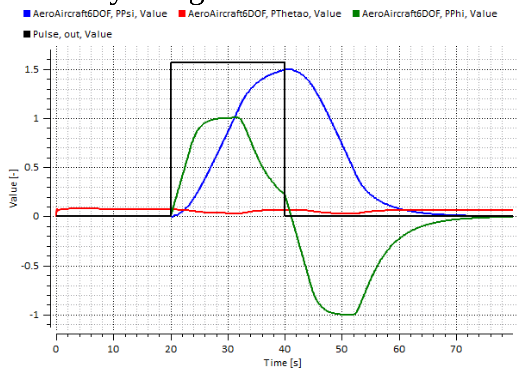


Figure 10. The aircraft attitudes during an S-maneuver.

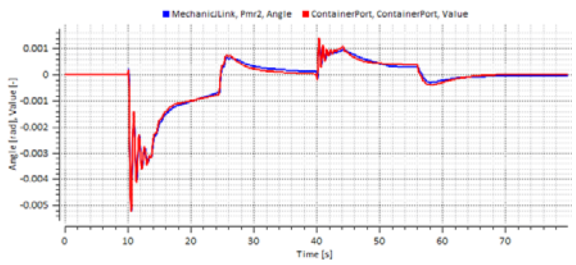


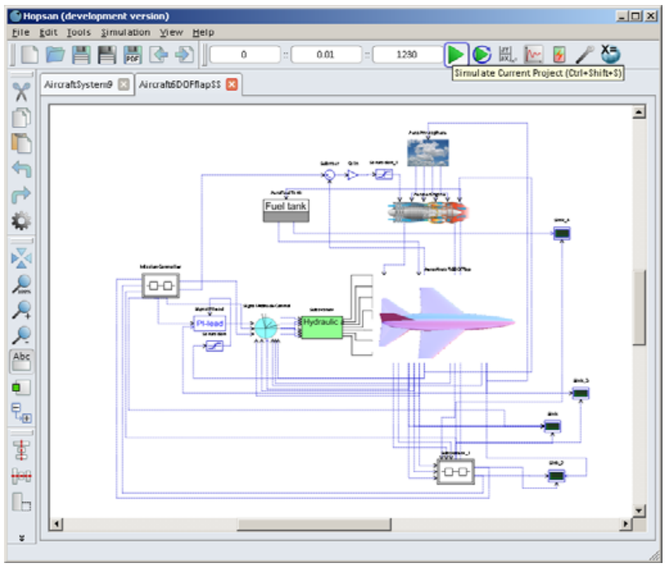
Figure 11. Angular position and reference position of the rudder actuator.

This simulation model is part of an integrated framework for aircraft design developed at Linköping University. In order to be part of a work flow, the system simulation needs to be able to integrate with other software. Hopsan-NG models can be exported for simulation within Matlab Simulink. Furthermore, the flight dynamics model has a parameter set that is very general, which means that coefficients can be obtained from other sources. Data is stored in XML for simple conversion and import of data.

4.2 Whole mission simulation

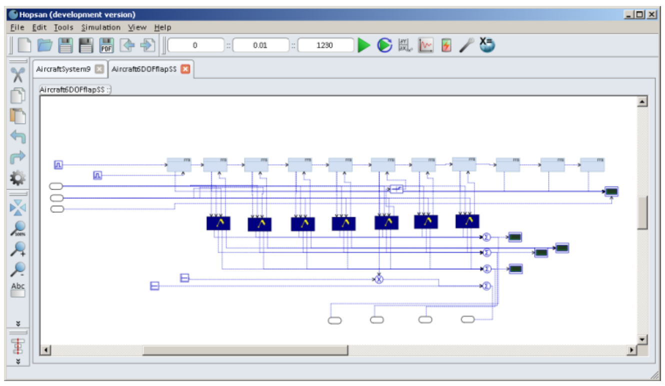
This model can also be simulated throughout a mission. In addition mission simulation also need a mission control unit where the flight is controlled using waypoints. In this way also performance measures such as fuel consumption can be assessed. Furthermore, with this kind of simulation full mission system analysis can be performed, including sensitivity analysis and studying the influence of uncertainty in parameters.

To simulate a whole mission another layer of control have to be implemented in the form of a mission controller. This is essentially a event based model based on *Functional Flow Block Diagram* (FFB). In this way it is possible to include a hierarchical description of event based systems. A transition to the next state is triggered if the aircraft gets close enough to a way point. This will generate reference heading, reference altitude and velocity for the next stage of the mission.



Hopsan (development version)
File Edit Tools Simulation View Help
0
0.01
1230
Simulate Current Project (Ctrl+Shift+S)
AircraftSystem9
Aircraft6DOFflapSS
Fuel tank
Hydraulic
PI-lead

Figure 12. System model for Whole system simulation.



Hopsan (development version)
File Edit Tools Simulation View Help
0
0.01
1230
AircraftSystem9
Aircraft6DOFflapSS
Aircraft6DOFflapSS

Figure 13. Mission controller based on state machines.

There is also a subsystem to introduce time acceleration. I sections where attitudes are close to constant, time is accelerated. This has a similar effect to variable time step, but the difference is that the internal time step is constant but that the rate of states related to distance traveled and consumed flow are multiplied by a a large factor (here it is 100) which means that what determines the simulation time is essentially the transient part of the mission.

4.3 Simulation results

Using the simulation model a simulation of a mission can be performed several times faster than real time PC. This means that is possible to use the model for design analysis such as sensitivity analysis. Furthermore, it is even possible to use it for optimization of some design parameters.

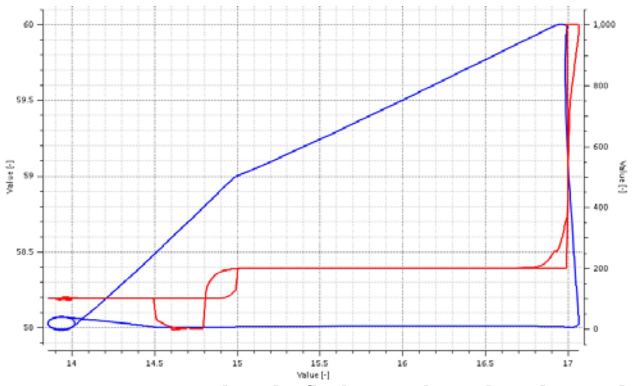


Figure 14. Simulated flight path. Altitude and lattitude as a function of longitude.

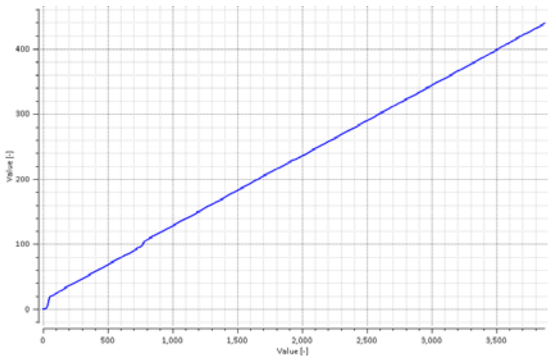


Figure 15. Consumed fuel as a funciton of time.

4.4 Influence of uncertainty in coefficient values

One way to manage the required accuracy of the coefficients involved in the simulation is to study the influence of removing the uncertainty of one coefficient (uncertainty variable), as in [16] and [17]. This is done through sensitivity analysis and is presented in a matrix where the relative influence on the functional characteristics, such as fuel consumption and mission time, is indicated.

5 Conclusions

In this paper the use of flight simulation, including subsystems, for design evaluation of the whole aircraft in the preliminary design phase, is demonstrated using the Hopsan NG simulation package. In this way system performance and subsystem interaction can be studied very early. Since efficient, hi-speed simulation of a complex system is desirable, robust simulation is a requirement. A key element is the method of using bi-lateral delay lines for partitioning of complex system models. In this way it is possible to use highly robust distributed solvers on subsystems that are then connected to each other using the bi-lateral delay lines, for system simulation. This means that highly robust models are achieved where very large time steps can be used. As a consequence fairly detailed simulation models can be used for whole mission simulation already at the preliminary design stage, for system optimization and design analysis.

References

[1] Auslander D M , 'Distributed System Simulation with Bilateral Delay-Line Models' Journal of Basic Engineering, Trans. ASME p195-p200, June 1968.

[2] Amies G., R Levek, D Struessel, 'Aircraft Hydraulic Systems Dynamic Analysis. Volume II. Transient Analysis (HYTRAN) Computer Program Technical Description'. A730930. 1977.

[3] Axin, M., R Braun, A Dell'Amico, B Eriksson, P Nordin, K Pettersson, I Staack, P Krus, 'Next Generation Simulation Software using Transmission Line Elements link', Fluid Power and Motion Control (FPMC), Bath, UK, 2010

[4] Braun B., P Nordin, B Eriksson, and P Krus, 'High Performance System Simulation Using Multiple Processor Cores', The Twelfth Scandinavian International Conference on Fluid Power (SICFP), Tampere, Finland, 2011.

[5] Burton, J.D., Edge, K.A. and Burrows, C.R., 'Modelling Requirements for the Parallel Simulation of Hydraulic Systems', ASME Journal of Dynamic Systems and Control, March 1994

[6] Duquette. ‘Effects-Level Models for UAV Simulation’, AIAA Modeling and Simulation Technologies Conference, Chicago, Illinois, USA, 2009

[7] Eriksson B., P Nordin, P Krus ‘Hopsan NG, A C++ Implementation using the TLM Simulation Technique link’, The 51st Conference on Simulation and Modelling (SIMS), Oulu, Finland. 2010

[8] Fritzson P., Principles of Object Oriented modelling and Simulation with Modelica 2.1, Wiley-IEEE Press. 2003.

[9] Jouannet C., Krus P. ‘Unsteady aerodynamic modelling: a simple state-space approach’. AIAAA Aerospace sciences meeting and exhibit,2005 Reno, USA.

[10] Johansson (now Lundén) B., Andersson J., and Krus P., “Thermal Modelling of an Electro-Hydrostatic Actuation System,” in Proceedings of the International Conference on Recent Advantages in Aerospace Actuation Systems and Components, Toulouse, France, June 13-15, 2001.

[11] Johns P., B. and M.O'Brien. 'Use of the transmission line modelling (t.l.m) method to solve nonlinear lumped networks.' The Radio Electron and Engineer. 1980.

[12] Krus P., ‘An Automated Approach for Creating Components and Subsystems for Simulation of Distributed Systems’ Presented at the ‘Ninth Bath International Fluid Power Workshop’, Bath, UK 1996.

[13] Krus P, ‘Complete Aircraft Simulation for Distributed System Design”, in Proceedings of the International Conference on Recent Advantages in Aerospace Actuation Systems and Components’, Toulouse, France, June 13-15, 2001.

[14] Krus P., A Jansson, J-O Palmberg, ‘Optimization Using Simulation for Aircraft Hydraulic System Design’, Proceedings of IMECH International Conference on Aircraft Hydraulics and Systems, London, UK, 1993

[15] Krus P., A Jansson, J-O Palmberg, K Weddfelt. ‘Distributed Simulation of Hydromechanical Systems’. Presented at 'Third Bath International Fluid Power Workshop', Bath, UK 1990.

[16] Krus P., ‘Computational tools for aircraft system analysis and optimization’. The 26th International congress of the aeronautical sciences. ICAS, Anchorage, USA, 2008.

[17] Steinkellner S. Licentiate thesis. ‘Aircraft Vehicle Systems Modeling and Simulation under Uncertainty’, LIU-TEK-LIC-2011:36, Linköping University.ACSL (1995),

[18] *Advance Continuous Simulation Language Version 11*, Reference Manual, Mitchell & Gauthier Associates, Concord MA, 1995

Copyright Statement