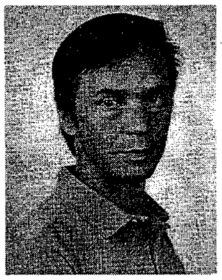
SIMULATION 52:3, 111-121
©1989 by Simulation Councils, Inc.
ISSN 0037-5497/89 $2.00 + .10
Printed in the United States of America

Key words: qualitative simulation, signal processing, decision-making and forecasting, management aids

Building intelligence into an autopilot using qualitative simulation to support global decision making



Pentti J. Vesanterä and François E. Cellier
Dept. of Electrical Engineering
University of Arizona, Tucson, AZ 85721

PENTTI J. VESANTERA recently received his M.S. degree in Electrical Engineering from the University of Arizona. He received his B.S. degree from Escola de Engenharia de Lins, Lins—SP, Brazil, in 1978. Before he came to the University of Arizona, Mr. Vesanterä worked as a project engineer for Elevadores KONE Ltda, the Brazilian subsidiary of a Finnish multinational elevator company where he designed control and logic systems for elevators for five years. Currently his main research interests are in simulation applied to AI and General Systems Theory.

FRANCOIS E. CELLIER received his B.S. degree in Electrical Engineering from the Swiss Federal Institute of Technology (ETH) Zurich in 1972, his M.S. degree in Automatic Control in 1973, and his Ph.D. degree in Technical Sciences in 1979, all from the same university. Following his Ph.D., Dr. Cellier worked as a Lecturer at ETH Zurich. He joined the University of Arizona in 1984 as an Associate Professor. Dr. Cellier's main scientific interests concern modeling and simulation, computer-aided modelling, and computer-aided design. He has designed and implemented the GASP-V simulation package, and he was the designer of the COSY simulation language, a modified version of which under the name of SYSMOD has meanwhile become a standard by the British Ministry of Defence. Dr. Cellier has authored or co-authored more than forty technical publications, and he has edited two books. He served as a chairman of the National Organizing Committee (NOC) of the Simulation '75 conference, and as a chairman of the International Program Committee (IPC) of the Simulation '77 and Simulation '80 conferences, and he has also participated in several other NOC's and IPC's. He is associate editor of several simulation related journals, and he served as vice-chairman of two committees on standardization of simulation and modelling software. Memberships include SCS and IMACS.

ABSTRACT

In this paper, qualitative simulation is applied to reason inductively about the behavior of a quantitatively simulated aircraft model, to determine on-line when a malfunction occurs in the quantitative model, to hypothesize about the nature of this malfunction, and to decide upon a global strategy that allows to operate (control) the quantitative aircraft model under the modified flying conditions.

Such an algorithm could be utilized as an addition to a conventional autopilot which would allow the autopilot to remain operational after a malfunction has taken place.

INTRODUCTION

With the continuous advances of technology in the area of automatic control, automation has become increasingly popular allowing us today to build highly complex control systems e.g. for modern jet cruisers, nuclear power plants, space stations, etc. Modern control technology allows systems analysis and control through a variety of techniques in the time- and/or frequency-domains. Lead-lag compensators, and state- and output-feedback designed by means of techniques such as pole-placement or parameter optimization by solving a matrix Riccati equation represent some of the techniques presently available. These are all well understood and widely in use. Decision-making in *clearly predefined situations* has also been successfully implemented with decentralized control systems, expert systems, and rule-based control systems.

Even though all these techniques work perfectly well under normal, well defined conditions, they drastically fail when facing a new, unforeseen (possibly emergency) situation such as a sudden, unexpected structural change in the system. Handling such a situation is a task that still has to be performed by a human operator with his/her inventiveness and capability of reasoning. The human mind has the skill of learning from the system behavior in the emergency, and the inventiveness of finding a new control strategy for the new situation. No automatic control technique presently available is able to adapt to unpredicted structural changes in the system.

Man-in-the-loop systems have been the answer to this problem until now. However when the degree of system complexity increases even further, human operators are being overloaded by the amount of information they are provided with. The human mind is incapable of taking many decisions instantaneously and simultaneously when these decisions are to be based on information arriving in huge bundles all at once. The amount of data to be processed is much too large for a human operator to act reliably and efficiently. Among other reasons, this is due to the fact that he loses his notion of temporal precedence of the arriving data items in the fast flowing sea of information that is formed.

This disables the operator from distinguishing between causes and effects as the emergency progresses, which is critical for his decision-making process, and an erroneous decision becomes likely.

We propose to address the stated problem by building an on-line monitoring system that mimics the human global assessment process, that identifies the event, learns the system behavior, and comes up inductively with a new control strategy for the structurally modified system. This on-line monitoring system will be an automatic device that merges the best of both worlds: the human inventiveness and the automatic controller's speed and systematism.

General Systems Problem Solving (GSPS) [10] is a methodological framework arising from General Systems Theory that allows the user to define and analyze types of systems problems. In this methodology, systems are defined through a hierarchically arranged set of epistemological subsystems. Forecasting and reconstruction analysis capabilities are two examples of the capabilities of the GSPS methodological tools. An on-line monitoring system can be implemented in the GSPS framework by using its inductive inference capability to imitate the human learning process. SAPS-II [1] is a software coded at the University of Arizona that implements the basic concepts of the GSPS framework. SAPS-II has been implemented as an application function library to the control systems design software CTRL-C [13]. In terms of common A.I. terminology, we can say that SAPS-II employs CTRL-C as an A.I. shell.

Our stratagem was successfully implemented in SAPS-II for a longitudinal model of an aircraft in cruise flight. An intelligent autopilot was implemented using the GSPS framework. It is able to identify structural changes in the monitored system, and to learn enough about the new (unknown, modified, broken) system to come up with a forecasting model that satisfactorily predicts future behavior of this system.

The approach taken is depicted in Figure 1.

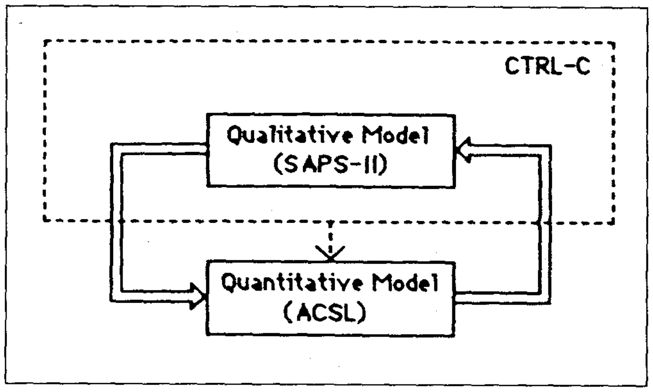


Figure 1. Block diagram of the experimental setup

A *quantitative* model of the airplane was coded in the continuous system simulation language ACSL [12]. "Accidents" are built into the aircraft model which alter the behavior of the aircraft. The data extracted from the quantitative ACSL simulation is used as our "measurement data." The ACSL simulation is executed under control of CTRL-C (the CTRL-C/ACSL interface is made available together with the CTRL-C software to customers who hold a valid license for the ACSL software). SAPS-II functions are used to *qualitatively* and *inductively* reason about the measurement data taken from the ACSL simulation run, determine that an accident has happened, find out when it occurred, hypothesize about the nature of the accident, and decide upon an appropriate corrective action to be taken.

THE QUANTITATIVE MODEL

Flight stability can basically be studied through two independent models: longitudinal and lateral. Longitudinal motions can be modeled independently from the lateral ones if the following simplifying assumptions are valid:

1. The airplane is perfectly symmetrical with respect to its median longitudinal plane.
2. There are no gyroscopic effects of spinning masses (engine rotors, airscrews, etc.) acting on the aircraft.

In this paper, we have adopted the above assumptions, and consider the longitudinal model of a B747 aircraft in cruise flight at high altitude.

A longitudinal flight is characterized by the absence of forces and moments that would cause its lateral motion. Furthermore, the aeroelastic nature of the airplane's structure is neglected as well, so that the *rigid body* equations of motion apply to the model.

The mathematical model described in this paper reflects an essentially longitudinal flight restricted to longitudinal deviations from a trimmed reference flight condition. This reference flight is characterized by the requirement that the resultant force and moment acting on the aircraft's center of mass are zero.

It is not the scope of this paper to provide full details about the stability study of flight. To follow the reasoning process here presented, one may consult specialized literature from that area [3,4,7,9]. The model described in this paper is structurally similar to the aircraft models that were given in [12,13]. The parameter values were taken from [8].

The Reference Flight Condition

We will define a reference flight condition as being characterized by a steady longitudinal and horizontal flight where the resultant force and moment acting on the plane are zero. The headwind is assumed to be constant and horizontal.

The Stability Axes

The theory presented in this paper is developed with respect to a set of body-fixed axes named the stability axes. The origin of this coordinate system is the center of gravity of the airplane: the x-axis points in the direction of the motion of the airplane in the reference flight condition, the z-axis points 'downward', and the y-axis runs spanwise and points to the right.

The Reference Angles

Three angles are defined to describe the relative position of the velocity vector of the center of gravity of the airplane with respect to an earth-fixed reference frame and a fuselage-fixed reference frame.

α is the angle of attack (or incidence) of the airplane which describes the inclination of the resultant velocity vector v to the x axis of the body-fixed coordinate system (stability axes). The usual notation of the velocity components in the stability axes is u for the x-axis component and w for the z-axis component. Hence

$$\alpha = tan^{-1}\left(\frac{w}{u}\right) \tag{1}$$

γ is the flight path angle of the aircraft, representing the inclination of the velocity vector to the horizontal, i.e., to the x-component of the earth-fixed reference frame.

θ is the pitch angle, being the one that is best sensed by a human pilot, for it represents the relative position between the two reference frames. It is defined in terms of the previous angles as follows:

$$\theta = \gamma + \alpha \tag{2}$$

The relationship between the different variables is graphically depicted in Figure 2.

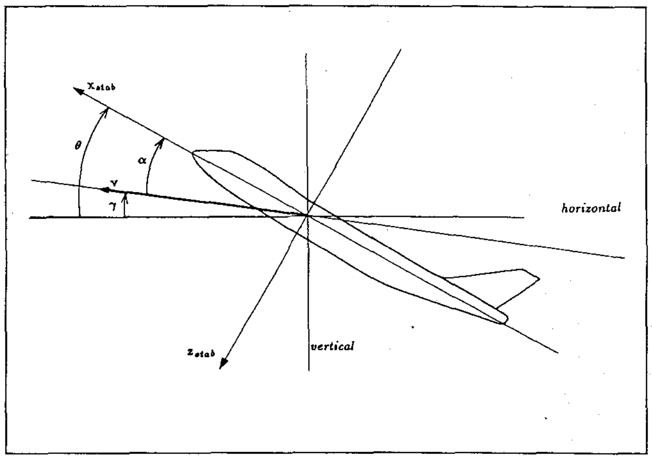


Figure 2. The reference angles

Forces and Moments

The tangential and normal components of the resultant force, and the moment about the center of gravity of the airplane considered as a rigid body, whose mass is constant over time, can be written in terms of the reference angles γ and θ as:

$$F_t = m\frac{dv}{dt} \quad (3a)$$

$$F_n = mv\frac{d\gamma}{dt} \quad (3b)$$

$$M_y = I_y\frac{d^2\theta}{dt^2} \quad (3c)$$

The quantities affecting the airplane in flight are its weight W, the thrust T developed by the engines, the aerodynamic forces Lift L and Drag D, and the aerodynamic pitching moment M.

The weight of the aircraft will be considered constant (thus the weight of the fuel consumed during the flight is neglected).

The thrust developed by the propulsive system will be considered as being a function of the flight velocity and of its own control variable δ_T, the throttle opening. For reasons of simplicity, the thrust line will be assumed to coincide with the x-axis of the stability axes. The center of gravity, by definition, is in this axis, and therefore, the thrust does not affect the moment directly.

The aerodynamic forces L and D compose the force response of the aircraft to the motion. They act in the mean aerodynamic center of the wing, causing the aerodynamic moment M about the center of gravity which is defined to be positive for a nose down effect. The Lift is defined as being the normal component of the aerodynamic force with respect to the flight path, and the Drag is its tangential component.

The forces and moments acting on the aircraft are graphically depicted in Figure 3.

The Aerodynamic Reactions L, D, and M

The standard way of expressing the aerodynamic forces D and L and the longitudinal aerodynamic moment M is through their nondimensional aerodynamic coefficients C_D, C_L, *and* C_M:

$$D = \frac{1}{2}\rho v^2 S C_D \quad (4a)$$

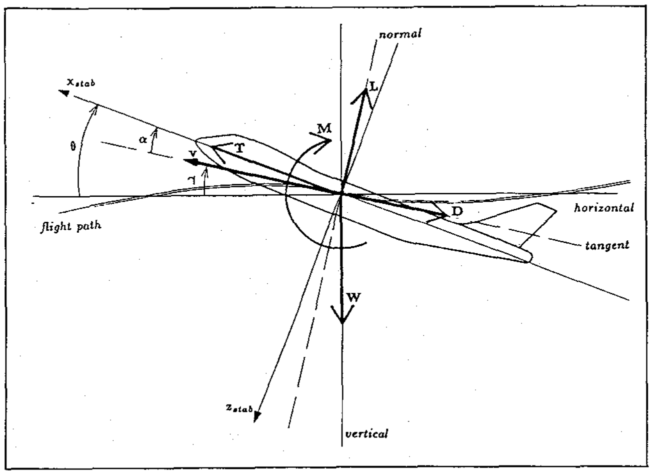


Figure 3. Forces and moments acting on the airplane.

$$L = \frac{1}{2}\rho v^2 S C_L \quad (4b)$$

$$M = \frac{1}{2}\rho v^2 S \frac{\bar{c}}{2} C_M \quad (4c)$$

which shows their direct dependence on the local air density ρ, the square of the cruising speed v, and the size of the aerodynamic surface S of the airplane.

Parameter $\frac{\bar{c}}{2}$ in the expression for the moment stands for the characteristic length (for the nondimensional coefficients), taken as half of the mean aerodynamic chord $\bar{c}$ of the wing.

The Nondimensional Coefficients C_D, C_L, and C_M

These three nondimensional coefficients express the aerodynamic response of the airplane to variations in the following aerodynamic variables:

1. α , the angle of attack
2. δ_e, the elevator deflexion
3. $\dot{\alpha}$, the angle of attack rate
4. q, the pitch rate

Equations (5*a*), (5*b*), and (5*c*) below describe the nondimensional aerodynamic coefficients expressed by a Taylor series expansion around an initial value (subscript 0) for which α, δ_e, $\dot{\alpha}$ and q are zero:

$$C_L = C_{L_0} + \frac{\partial C_L}{\partial \alpha}\alpha + \frac{\partial C_L}{\partial \delta_e}\delta_e + \frac{\partial C_L}{\partial \dot{\alpha}}\dot{\alpha} + \frac{\partial C_L}{\partial q}q \quad (5a)$$

$$C_D = C_{D_0} + \frac{\partial C_D}{\partial \alpha}\alpha \quad (5b)$$

$$C_M = C_{M_0} + \frac{\partial C_M}{\partial \alpha}\alpha + \frac{\partial C_M}{\partial \delta_e}\delta_e + \frac{\partial C_M}{\partial \dot{\alpha}}\dot{\alpha} + \frac{\partial C_M}{\partial q}q \quad (5c)$$

The Stability Derivatives

The aerodynamic reactions of the airplane can be represented approximately by means of stability derivatives, i.e., the coefficients of the Taylor series expansion above.

Note that as C_D is strongly influenced by the angle of attack, all other influences can be neglected.

The α derivatives C_{L_α}, C_{D_α}, and C_{M_α}, describe how changes in the angle of attack affect the aerodynamic forces and moments. An increase in the angle of attack generally induces an increase in the Lift, an increase in the Drag, and a decrease in the pitching moment.

The δ_e derivatives $C_{L\delta_e}$ and $C_{M\delta_e}$ describe the effect that a deflexion of the elevator has on the Lift and on the pitching moment. A positive elevator deflexion is defined as being *elevator down,* which causes an increase in the Lift and a negative pitching moment increment.

The $\dot{\alpha}$ derivatives $C_{L\dot{\alpha}}$ and $C_{M\dot{\alpha}}$ represent the adjustment of the pressure distribution on the aerodynamic surfaces to sudden changes in the angle of attack, as, for example, when sudden changes in the incidence of the headwind occur.

The q derivatives C_{Lq} and C_{Mq} represent the aerodynamic effects induced by a rotation of the airplane about its spanwise axis when the angle of attack is kept constant, for example, keeping the fuselage horizontal in an arbitrarily varying flight path.

These two rotational effects can be visualized considering a flight along an arbitrary flight path: first with the fuselage of the plane always tangential to the flight path (angle of attack kept zero), and second with the fuselage always horizontal (pitch angle kept zero).

Finally, the nondimensional aerodynamic coefficients can be expressed by the set of equations:

$$C_L = C_{L_0} + C_{L_\alpha}\alpha + C_{L\delta_e}\delta_e + \frac{\bar{c}/2}{v}[C_{L\dot{\alpha}}\dot{\alpha} + C_{L_q}q] \quad (6a)$$

$$C_D = C_{D_0} + C_{D_\alpha}\alpha \quad (6b)$$

$$C_M = C_{M_0} + C_{M_\alpha}\alpha + C_{M\delta_e}\delta_e + \frac{\bar{c}/2}{v}[C_{M\dot{\alpha}}\dot{\alpha} + C_{M_q}q] \quad (6c)$$

Note that the rotational derivatives $C_{L\dot{\alpha}}$, C_{Lq}, $C_{M\dot{\alpha}}$, and C_{Mq} are multiplied by $\frac{\bar{c}/2}{v}$. This comes from the fact that these derivatives are, in fact, taken with respect to the quantity $\frac{\dot{\alpha}\bar{c}/2}{v}$ (or $\frac{q\bar{c}/2}{v}$) where $\bar{c}$ is the mean aerodynamic chord of the wing and v is the cruising velocity.
For example, consider $C_{L\dot{\alpha}}$:

$$C_{L\dot{\alpha}} = \frac{\partial C_L}{\partial\frac{\dot{\alpha}\bar{c}/2}{v}} \quad (7a)$$

$$= \frac{\partial C_L}{\frac{\bar{c}/2}{v}\partial\dot{\alpha}} \quad (7b)$$

therefore

$$\frac{\partial C_L}{\partial\dot{\alpha}} = \frac{\bar{c}/2}{v}C_{L\dot{\alpha}} \quad (8)$$

Longitudinal Flight Control

Longitudinal flight control means control of the velocity vector v acting on the center of gravity of the airplane. The two available control elements are δ_e for the elevator deflexion and δ_T for the throttle control of the thrust.

The immediate response of the aircraft to a $\Delta\delta_e$ at constant throttle is a brisk rotation in pitch and a consequent change in both angle of attack and Lift, followed by a curvature $\dot{\gamma}$ of the flight path. After this first fast transient, the new steady state flight is characterized by the new values of γ_{ss} and u_{ss}. The steady-state speed u_{ss} is fixed by the value of $C_{L_{ss}}$, which, in turn, is determined by δ_e (cf. [4], Sections 2.5 and 9.1).

The immediate effect of a positive $\Delta\delta_T$ with fixed δ_e is essentially a change in the velocity followed by a change in the flight path angle γ. However as a given δ_e fixes a constant steady state velocity, the final effect of opening the throttle will be a change in the flight path angle without changing the speed.

The Overall Mathematical Model

Equations of Motion

We are now able to write down the equations of motion. This will not be done in the stability axes, but in the tangential and normal axes with respect to the flight path, because this simplifies the equations:

$$m\dot{v} = T\cos\alpha - D - W\sin\gamma \quad (9a)$$

$$mv\dot{\gamma} = T\sin\alpha + L - W\cos\gamma \quad (9b)$$

$$I_y\dot{q} = M \quad (9c)$$

$$\dot{\theta} = q \quad (9d)$$

Equation (2) reflects the relationship between the reference angles, and the position of the airplane with respect to the ground is given by the equations (10):

$$\dot{h} = v\sin\gamma \quad (10a)$$

$$\dot{x} = v\cos\gamma \quad (10b)$$

Aerodynamic Equations

Equations (4a), (4b), and (4c) specify the aerodynamic quantities L, D, and M, and the nondimensional coefficients C_L, C_D, and C_M are given by the equations (6a), (6b), and (6c).

Closed Loop Equations

The two control laws implemented in the model are standard procedure in the control of flight stability [4,5]. Feedback of the pitch angle deviation from its trimmed value (for which the airplane is in steady horizontal reference flight) into the elevator deflexion suppresses effectively the phugoid mode of the airplane which is slow and very lightly damped. The second control loop was similarly implemented feeding back the velocity into the thrust.

$$\delta_e = \delta_{etrim} + K_\theta(\theta - \theta_{trim}) \quad (11a)$$

$$T = T_{trim} + K_u(u - u_{trim}) \quad (11b)$$

The subscript $_{trim}$ refers to the trimmed value of the variable, and u is the x-component of the velocity in the stability axes, or

$$u = v\cos\alpha \quad (12)$$

Model Parameters

Listed below are all the values used for the flight related constants. The airplane related physical data was chosen for a large commercial/cargo Boeing 747 jet plane in cruise flight at an altitude of 20,000 ft and a speed of Mach .5 ($\approx$ 500 ft/s).

The aerodynamic coefficients were adapted for a trimmed reference flight with a given set of initial conditions which is characterized by a horizontal steady flight at 500 ft/s, an altitude of 20,000 ft, a zero angle of attack, an elevator deflexion of 1.6 degrees (.0279 rad), and a constant thrust of 33,000 slugs. The initial conditions were then set such that the flight would start perfectly trimmed, since approximation errors in the aerodynamic constants had still to be corrected.

Airplane Constants

I_y	= 27,000,000.0	$[lb\ ft^2]$
W	= 500,000.0	$[slug]$
$\bar{c}$	= 27.3	$[ft]$
S	= 6,000.0	$[ft^2]$

Physical Constants

g	= 32.2	$[ft/s^2]$
ρ	= 0.0012	$[lb/s^3]$, at 20,000 ft

Aerodynamic Constants

C_{D_0}	= 0.036667	[ ]
C_{D_α}	= 0.26	$[1/rad]$
C_{L_0}	= 0.5455	[ ]
C_{L_α}	= 5.2	$[1/rad]$
$C_{L_{\delta_e}}$	= 0.36	$[1/rad]$
$C_{L_{\dot\alpha}}$	= 2.0	$[1/rad]$
C_{L_q}	= 5.5	$[1/rad]$
C_{M_0}	= 0.039	[ ]
C_{M_α}	= −0.74	$[1/rad]$
$C_{M_{\delta_e}}$	= −1.4	$[1/rad]$
$C_{M_{\dot\alpha}}$	= −8.0	$[1/rad]$
C_{M_q}	= −22.0	$[1/rad]$

Feedback Gains

K_θ	= 0.25	[ ]
K_u	= 40.0	$[lb/s]$

Initial Conditions

v_0	= 500.1375	$[ft/s]$
h_0	= 20,000.0	$[ft]$
x_0	= 0.0	$[ft]$
q_0	= 0.0	$[rad/s]$
α_0	= −0.000055	$[rad]$
θ_0	= −0.000055	$[rad]$
γ_0	= 0.0	$[rad]$
δ_{e0}	= 0.027886	$[rad]$
T_0	= 33,005.5	$[slug]$

INPUTS (CONTROLS) AND OUTPUTS

The "external" connections to our quantitative aircraft model are shown in Figure 4.

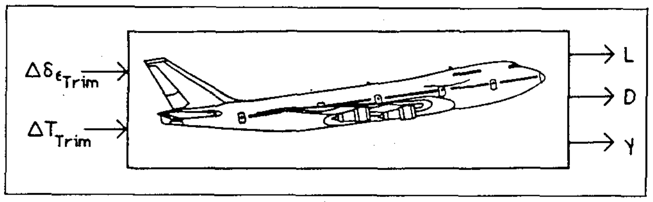


Figure 4. External connections to quantitative aircraft model

There are two control inputs, namely the differential elevator deflexion $\Delta\delta_{e_{trim}}$ (i.e., the deviation of the elevator deflexion from its trimmed value), and the differential thrust ΔT_{trim}.

Three quantities are measured as outputs, namely the Lift L, the Drag D, and the flight path angle γ. It is postulated that these three quantities together with the two control inputs suffice to capture the main characteristics of the aircraft during high altitude longitudinal cruise flight.

From a systems theoretic point of view, this means that our "source system" consists of the quintuple:

$$S_s = \{\Delta\delta_{e_{trim}}\ \Delta T_{trim}\ L\ D\ \gamma\} \tag{13}$$

QUANTITATIVE SIMULATION

In the following experiment, the quantitative model was trimmed around its operating point. We then evaluated the Jacobian of the model at the working point to determine the eigenmodi of the linearized system. These will be used later to determine an adequate sampling rate for the qualitative simulation. Thereafter, we applied differential step disturbances to the two control inputs of the nonlinear continuous-time model one at a time, performed two simulation runs in which the values of the two control inputs and the three system outputs (under closed loop control conditions) were recorded, and finally, the results of the two simulation runs were plotted on top of each other.

The following CTRL-C code shows how the CTRL-C/ACSL interface allows us to compute the eigenvalues and the time constants of the linearized model.

```
[> ACSL('SET TMX=0')
[> FREEZE('X,H')
[> START
[> a = JACOBIAN

A        =

  -0.6034    0.0236    0.6137    0.0003
   0.5538   -0.7408   -0.8705    0.0001
   0.0000    1.0000    0.0000    0.0000
 -17.1213    0.0000  -15.0787   -0.0059


[> lambda = EIG(a)

LAMBDA   =

  -0.0850 + 0.0056i
  -0.0850 - 0.0056i
  -0.5901 + 0.8722i
  -0.5901 - 0.8722i


[> tau = [ -1/REAL(lambda(1)) ; -1/REAL(lambda(3)) ]

TAU      =

  11.7648
   1.6948


[> tsample = ROUND(tau/2)

TSAMPLE  =

   6.
   1.
```

In the first statement ("[>" is CTRL-C's prompt for interactive user input), a value of 0 is passed from CTRL-C to the ACSL variable TMX (denoting the final time of the simulation). The next statement ("freeze('x,h')") tells ACSL to eliminate the two state variables x (the horizontal displacement) and h (the vertical displacement) from the linearized model. This is necessary since both these state variables are outputs of open-loop integrators that cannot be "trimmed." ACSL's "TRIM" command could have been used next to trim the model to stationary initial conditions (where all state derivatives are zero). However, this step was not necessary here since the initial values had been picked carefully so that the model would always start off from a trimmed initial state. The third statement ("start") performs a "simulation" from time 0 to time TMX (also 0) which is necessary to determine the linearized model. The last statement ("a=jacobian") tells ACSL to linearize the model around the current (trimmed) state, and to export the resulting Jacobian (the system matrix of the linearized model) back from ACSL to CTRL-C. The following statements are regular CTRL-C statements to determine the eigenvalues of the Jacobian, and to estimate the time constants which will be required later. The slowest time constant present in the model is $\tau_1 \cong 12$ seconds.

The following two sets of CTRL-C/ACSL code were used to retrieve the step input response of the system shown in Figures 5a and 5b:

```
[>  ACSL('SET inpt=1, tmx=200, cint=.1')
[>  ACSL('SET dde1=-.001, tde1=10')
[>  ACSL('SET dde2=0, dde3=0, dtr1=0, dtr2=0, dtr3=0')
[>  [t,de,detrim,l,d,ga] = START;
[>  SAVE >temp.dat
```

Here, the simulation is started from trimmed conditions at time zero. *It is being perturbed with a negative step of magnitude* –.001 radians in the reference value of the elevator deflexion scheduled at time t = 10 seconds. The time history of the perturbation step $\delta_{e_{trim}}$, and its effect on the elevator deflexion, Lift, Drag, and flight path angle are temporarily saved.

The next code describes the simulation of the model with a perturbation step of magnitude δ_T = 3,000 slug in the thrust, again scheduled to happen at time t = 10 seconds.

```
[>  ACSL('SET inpt=1, tmx=200, cint=.1')
[>  ACSL('SET dtr1=3000, ttr1=10')
[>  ACSL('SET dde1=0, dde2=0, dde3=0, dtr2=0, dtr3=0')
[>  [tr,trtrim,l,d,ga] = START;
[>  ltr = l;  dtr = d;  gatr = ga;
[>  CLEAR l d ga
[>  LOAD l d ga <temp.dat
[>  term = '4100';
[>  WINDOW('211'),  PLOT(t,[de,detrim]),  XLABEL('TIME'),  YLABEL('DE,DETRIM')
[>  WINDOW('212'),  PLOT(t,[tr,trtrim]),  XLABEL('TIME'),  YLABEL('TR,TRTRIM')
[>  // SCREENCOPY => Figure 5a.
[>  ERASE
[>  WINDOW('211'),  PLOT(t,[l,ltr]),  XLABEL('TIME'),  YLABEL('LIFT')
[>  WINDOW('212'),  PLOT(t,[d,dtr]),  XLABEL('TIME'),  YLABEL('DRAG')
[>  // SCREENCOPY => Figure 5b. (first part)
[>  ERASE
[>  WINDOW('211'),  PLOT(t,[ga,gatr]),  XLABEL('TIME'),  YLABEL('g','g')
[>  // SCREENCOPY => Figure 5b. (second part)
```

The outputs of the second simulation run are the perturbation step input, and its effects on the thrust, Lift, Drag, and flight path angle. The variables are renamed so that we can reload the time histories of these same variables from the first simulation run.

The results of the two ACSL simulations are plotted using CTRL-C's (rather than ACSL's) graphical output processor. The results are presented in Figure 5a (showing the two inputs for both simulation runs), and Figure 5b (displaying the three outputs for both simulation runs).

The solid-line plots represent the response of the system to a step change in the elevator deflexion at fixed throttle opening, and the dashed-line plots represent the response at fixed elevator deflexion and a step change in the thrust. Both perturbations start from the trimmed reference flight characterized by the initial conditions.

The Experimental Design

In order to take correct global decisions, we should learn as much as possible about the system under investigation. From classical identification techniques commonly used by control engineers, we know that we can learn the most about a system under investigation if we disturb (shake) it by exerting all frequencies equally. This can best be achieved by applying random input streams to all input variables.

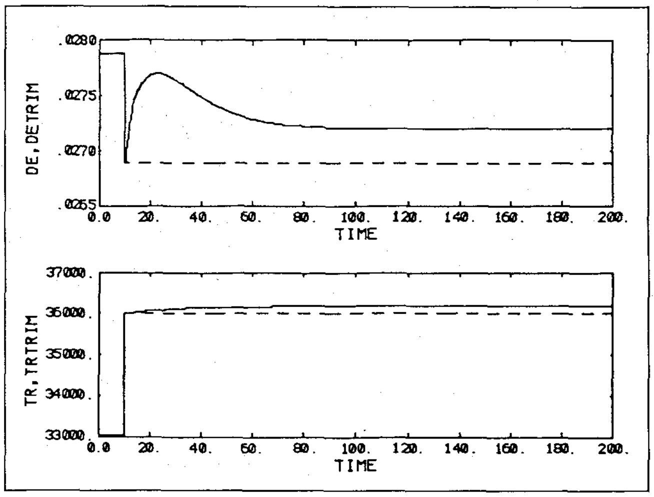


Figure 5a. Effect of the perturbations on the controls

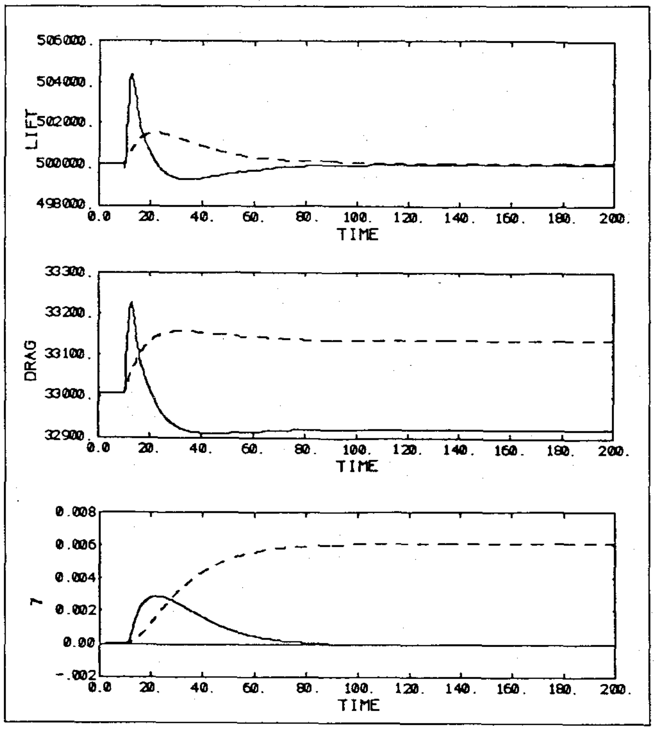


Figure 5b. System response to the perturbations

For our purpose, it proved most effective to apply *random ternary disturbances* to both control inputs, i.e., to assign a constant value to each of the two control inputs during each communication interval (CINT), a value which is either equal to the trimmed reference value itself or equal to the reference value plus or minus a "unit" step disturbance (.001 radians for the elevator deflexion, and 3,000 slug for the thrust). The sign of the disturbance is randomly selected between +1, 0, and –1 once per communication interval.

GSPS

General systems problem solving or analysis through General Systems Theory starts by defining a region in the universe where the system and the observer coexist and interact.

A system in this context can be interpreted as a set of relations between some objects that belong to that region of the universe and in which the observer is interested.

Therefore, the first step to problem solving, or analysis, is the definition of the system: what is it, that is of interest to us concerning the problem under study? A set of variables to represent the system has to be chosen, and this set is to be classified into *input variables* and *output variables*, which is a natural classification of the variables: input variables depend on the environment and control the output variables.

Epistemological Hierarchy

The GSPS framework is a hierarchically arranged set of epistemological subsystems. Starting at level zero, the amount of knowledge in the systems increases as we climb up the epistemological ladder. The lower level subsystems are contained in the ones that are at higher epistemological levels.

The Source System

At the lowest epistemological level, we find the Source System which represents the system as it is recognized by the observer. The amount of information present at this level represents the basic description of the problem in which the observer is interested: which variables are relevant to the problem, what causal relationships are present among them (which are inputs and which are outputs to the system), and which are the states these variables can possibly assume along their time-history.

To illustrate the definition, let us consider the first flying lesson of a future pilot in a B747 flight simulator. Let us assume the simulation starts off from a stabilized (trimmed) longitudinal flight at high altitude. The job of the would-be pilot is to bring the aircraft safely down to the ground. So, he starts by playing around with all the controls that he has available, very cautiously in the beginning. He suddenly detects a control that makes the nose of the aircraft go up and down with respect to the horizon (he "senses" the flight path angle). Then he gets curious about the velocity and checks out the speedometer: the speed is around "500." The number does not mean much to him, since he does not even know the unit it is coded in. But it is a reference value because the plane is flying alright; so that must be a good value for the air speed. He tries the controls again and observes the variation of the speed until he is able to code the reading of the instrument to something like: "somewhere near 500," "above 500" and "below 500." Then he moves to the other variables that he can find, and within his capabilities of observation, he analyses them and codes them in the way he understands and feels them. He builds a mental model of the aircraft. When he gets enough confidence in his understanding of the way things work in the cockpit, he starts to experiment a little more aggressively. Things like "What happens if..." start crossing his mind, and he starts restricting his attention to certain aspects of the flight, defining in this way an area of interest in that sea of instruments and different sensations he is experiencing for the first time. The knowledge that our aspiring pilot has now acquired-a set of variables of interest and a set of states these variables can potentially assume-is defined as a Source System in GSPS.

The number of states, or levels, that each variable can potentially assume is essentially problem dependent. It should be kept as low as possible without unacceptable loss of information. Consider again the speedometer with its needle at 500. Let us assume that that is the standard cruise velocity for the airplane, and therefore, in that region, the scale has a higher resolution with more subdivisions. In his first experiments, our pilot was doing fine interpreting the velocity as "about 500," "high," and "low," but now he may wish to become a little more accurate because he wants to fly faster and slower, and he decides that five levels are more appropriate for his new task.

The Data System

The next epistemological level in the hierarchy is represented by the Data System. It includes the Source System and, additionally, the time-history of all the variables of interest.

The Behavior System

One epistemological level higher we find the Behavior System which holds, besides the knowledge inherent to both, Source and Data Systems, a set of time-invariant relationships among the chosen variables for a given set of initial or boundary conditions. Behavior systems can be considered basic cells for yet higher epistemological levels, the so called Structure Systems.

The time-invariant relationships among the variables are *translation rules* mapping these variables into their common spaces. They can be used to generate new states of the variables within the time span defined in the Data Model, allowing in this way an *inductive system modelling* feature in the methodology. it is based on this feature that a monitoring device for the system can be built to detect structural changes in it. Due to this characteristic, Behavior Systems are also called Generative Systems.

The Concept of a Mask

A Data Model in the GSPS framework is an $n_{rec} \times n_{var}$ matrix where n_{rec} is the number of recordings (data points) collected in the time span covered by the Data Model, and n_{var} is the number of variables present in the model. This is a matrix representation of the time-history of the system, and the convention is that time increases from the top to the bottom of the matrix.

A mask is a matrix representation of a translation rule for a given Data Model, hence, it is a matrix representation of a Behavior Model of the system. The dimensions of the mask are $(d + 1) \times n_{var}$, where d is the depth of the mask representing the number of sampling time units covered by the mask.

The active elements of a mask are called sampling variables, and represent the variables that are to be considered in the translation rule associated with the time instant they occur.

Generative masks include in their structure the notion of *causality* among the variables. Elements of a generative mask are zero, negative, or positive, meaning "neutral element," "generating element," and "generated element," respectively. For example, a generative mask like

$$\begin{array}{c} \\ t-2\Delta t \\ t-\Delta t \\ t \end{array} \left(\begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \\ 0 & 0 & -1 & 0 & 0 \\ -2 & 0 & 0 & -3 & 0 \\ 0 & -4 & 0 & 0 & +1 \end{array} \right),$$

corresponds to the translation rule:

$$v_5(t) = f\left(v_3(t-2\Delta t),\ v_1(t-\Delta t),\ v_4(t-\Delta t),\ v_2(t)\right)$$

where $v_i(\tau)$ is the state assumed by the variable v_i at time $t = \tau$.

The Sampling Interval

Note that the Behavior Model above takes samples of the Data Model at every Δt_{th} data point to predict the state of v_5. Hence, Δt is the sampling interval t_s of the collected data set. There is not a precise way of determining the most effective sampling interval to be used, but a good rule of thumb is that the mask should cover the dynamics of the slowest mode in the model [2]. In the case of the given example, the mask has depth 2 and the sampling interval Δt should then be about half of the slowest time constant of the model. In our case, the slowest time constant was found to be $\tau_1 \cong 12$ seconds. Accordingly, we selected the sampling period (CINT in the ACSL program) to be 6 seconds. Experimentation with different sampling periods verified this to be a good choice.

Notice however that, as outlined in [2], selection of an appropriate sampling rate is absolutely crucial to the success of our endeavor, thus careful experimentation with this parameter is indicated under all circumstances.

Converting Quantitative Data into Qualitative Data

In order to be able to qualitatively reason about the behavior of our system, we need to convert the "measured" *quantitative data* (i.e. continuous variables) into *qualitative data* (i.e. variables of an enumerated type). GSPS calls this process the *recoding* of the measurement data. SAPS-II provides for various algorithms to recode *real* variables into sets of *integers.*

The control inputs of the source model are the perturbations $\Delta\delta_{e_{trim}}$ and ΔT_{trim} affecting the model through step changes in the trimmed reference values of the elevator deflexion and of the thrust developed by the engines. The shaken flight phase has been implemented in the model such that these step changes occur at every communication interval being randomly negative, zero, or positive, and therefore, each of these variables can assume three different states. These can naturally be recoded into the set of integers $\{1\ 2\ 3\}$.

The output variables $\{L\ D\gamma\}$ are truly continuous variables, and an appropriate selection of the recoding procedure will decide over success or failure of our endeavor. The number of recoding levels to be used for each variable has, intuitively, to be odd if we want to have a "normal" range of operation, and variations about it. Let us try first to allow five different ranges for each output variable denoting "very low," "low," "average," "high," and "very high" represented in SAPS through the set of integers $\{1\ 2\ 3\ 4\ 5\}$. But where should we draw the line between neighboring levels? Which values are "low" as compared to "very low"? We could ask an expert's opinion (which may frequently be the best solution). However in many cases (such as the case of our likely crash pilot in the flight simulator), there is no expert around. In this situation, it seems intuitively most appealing to request each "class" (range) to contain the same number of "members" (samples). This can best be achieved by sorting each output variable separately (using the standard CTRL-C *sort*-function), thereafter split the resulting vector into five subvectors of equal size, and determine appropriate elements for the *from*-matrix used in the recoding by looking at the first and last elements of each subvector. The following SAPS/CTRL-C code shows the recoding process.

```
[>  [nr,nv] = SIZE(data);
[>  from = ZROW(1,3);
[>  raw = ZROW(nr,1);
[>  FOR I=1:nv, ...
[>    [tag,d1] = SORT(data(:,I)); ...
[>    fr(1,1) = d1(1); ...
[>    fr(2,1) = 0.5*(d1(ROUND(nr/3)) + d1(ROUND(nr/3)+1)); ...
[>    fr(1,2) = fr(2,1); ...
[>    fr(2,2) = 0.5*(d1(ROUND(2*nr/3)) + d1(ROUND(2*nr/3)+1)); ...
[>    fr(1,3) = fr(2,2); ...
[>    fr(2,3) = d1(nr); ...
[>    r = RECODE(data(:,I),'DOMAIN',fr,1:3); ...
[>    raw = [raw,r]; ...
[>    from = [from;fr]; ...
[>    END
[>  raw = raw(:,2:nv+1);
[>  from = from(2:2*nv+1,:);
```

One last parameter still needs to be decided upon, namely the number of recordings that we need for our GSPS analysis. From classical statistical techniques, we know that each "class" (that is, each possible state) should contain at least five "members" (i.e., should be recorded at least five times) [11]. Therefore, if n_{var} denotes the number of variables, and if n_{lev_i} denotes the number of levels assigned to the variable v_i after recoding, we can write down the following (optimistic) equation for the minimum necessary number of recordings (n_{rec}):

$$n_{rec} = 5 \prod_{i=1}^{n_{var}} n_{lev_i} \tag{13}$$

i.e., in our case:

$$n_{rec} = 5 * 3 * 3 * 5 * 5 * 5 = 5625$$

Unfortunately, SAPS-II is currently limited to 2,000 recordings. For this reason, it was decided to recode all five variables into three levels only. However, it would have been perfectly feasible to recode two of the three output variables into five levels, and only one into three levels. There is always a conflict between the demands of simplicity for the purpose of a strong forecasting power, and an improved resolution for the purpose of a strong expressiveness of the model.

Recoding each variable into one level only results in an infinitely "valid" model with no expressiveness whatsoever. On the other hand, recoding each variable into a high number of levels (ignoring for the moment the given limitations of SAPS-II) will result in a highly "expressive" model with little to no forecasting power.

Notice that we just came across a *serious* problem with our modeling methodology which drastically limits the applicability of the proposed technique. Originally, we had believed that we could detect a structural change in the model, and *on-line* identify a new "model" (that is, generate a new set of masks) for the damaged aircraft. Unfortunately, this is not so. Since the sampling rate cannot be chosen freely but depends on the (structurally determined) eigenvalues of the system, and since the number of recordings cannot be chosen freely but depend on the complexity of the given source system (i.e. the number of variables and the number of levels), we can compute the minimum required total simulation time (TMX) for the generation of a set of new masks as:

$$TMX = n_{rec} * \Delta t \tag{14}$$

i.e., in our case:

$$TMX = 2500 * 6 \textit{ seconds} = 15000 \textit{ seconds} = 4 \textit{ hours and } 10 \textit{ minutes}$$

That is, we must fly the already damaged aircraft for more than four hours before we can identify a new "model" (set of masks) which would enable us to take a global decision as to how to control the future flight of the aircraft.

For the above reasons, our current study limits us to distinguishing between different types of *prerecorded* "accidents." Such a decision can be taken much more rapidly, namely after approximately 5 minutes only.

The conclusion that *something* went wrong can be drawn even faster, namely 30 seconds after the accident happened. (It is still questionable whether our "intelligent" autopilot would be granted his "flying license" with the exhibited response time though!)

In order to be able to react more quickly, we would have to feed more a *priori* knowledge into our qualitative simulator which would make the simulator less generally applicable, but more effective for the task at hand. E.g., rather than letting the simulator experiment with the controls at random, we could ask an expert pilot what type of experiments he would perform under such circumstances, namely e.g. to check which is the smallest radius of a circle that the aircraft is still able to fly.

The Optimal Mask Analysis

Given a Data Model, any topologically compatible mask associated with it is "valid" since it denotes a representation of a relationship among the sampling variables it contains. The question now is "How *good* is the mask?," "How valid is the translation rule it represents?" There are numerous possible masks that can be written for one set of variables, and it is desirable to determine among all possible masks the one that shows the least uncertainty in its generating capability, i.e., that maximizes the forecasting power. This is exactly what the *optmask*-function of SAPS-II evaluates. The measure of uncertainty that is currently employed by this function is the Shannon Entropy. SAPS-II requests the user to specify a mask candidate matrix which contains the element –1 for potential generating elements of the resulting optimal mask, 0 for neutral elements, and +1 for generated elements of the optimal mask.

We assume that the present states of the outputs do not affect each other, whereas current states of the input variables may affect any of the outputs instantaneously. The following set of mask candidate matrices was used for the optimal mask evaluation:

```
         -1 -1 -1 -1 -1              -1 -1 -1 -1 -1              -1 -1 -1 -1 -1
mcon1 = -1 -1 -1 -1 -1     mcon2 = -1 -1 -1 -1 -1     mcon3 = -1 -1 -1 -1 -1
         -1 -1  1  0  0              -1 -1  0  1  0              -1 -1  0  0  1
```

The following CTRL-C/SAPS code evaluates the optimal masks:

```
[> mcon1 = -1*ONES(3,5); mcon1(3,3:5) = [1,0,0];
[> mcon2 = -1*ONES(3,5); mcon2(3,3:5) = [0,1,0];
[> mcon3 = -1*ONES(3,5); mcon3(3,3:5) = [0,0,1];
[> mask1 = OPTMASK(rawrec,mcon1,5);
[> mask2 = OPTMASK(rawrec,mcon2,5);
[> mask3 = OPTMASK(rawrec,mcon3,5);
```

The following masks were found to best represent the aircraft under normal (i.e. undisturbed) flying conditions:

```
         -1  0  0  0 -2              -1  0  0  0  0               0  0  0  0 -1
mask1 = -3 -4  0  0  0     mask2 = -2 -3  0 -4  0     mask3 = -2  0  0 -3  0
          0  0  1  0  0               0  0  0  1  0               0  0  0  0  1
```

Forecasting

Now let us apply the generative systems represented by the optimal masks just calculated, to see how well they are able to forecast future behavior of the same system.

To do so, we will use a less shaken data model to represent a normal flight in a more natural way. A more realistic flight, but still a dynamic one, can be represented by the model being driven by harmonic functions of fairly long periods, which was implemented by sinusoidal functions affecting the same trim values $\delta_{e_{trim}}$ and T_{trim}. The amplitude of the functions is the same as the magnitude of the respective pulses in the shaken flight: $\Delta\,\delta_e$ = .001 rad and $\Delta T = 3000$ slug. In this flight phase, the frequency of oscillation can be set by the parameters WDE for the elevator deflexion and WTR for the thrust, but their default values: WDE = 0.1 and WTR = 0.05, yielding periods of about 63 and 126 sec. respectively, were used in the following run. The length of the flight will be 15600 seconds, generating, at a sampling interval of 6 sec., a data model with 2601 recordings, out of which the first 2500 are used for the determination of the optimal masks while the final 101 values are used later on for comparison with the forecast.

Recoding of the data model was done into three levels for all variables. The input variables now need recoding as well, since they are no longer discrete values. However, as both inputs are harmonic, their recoding can be done evenly in their range of angular variation:

$$\text{level 1:} \quad \left[0, \frac{2\pi}{3}\right) \quad \text{rad}$$

$$\text{level 2:} \quad \left[\frac{2\pi}{3}, \frac{4\pi}{3}\right) \quad \text{rad}$$

$$\text{level 3:} \quad \left[\frac{4\pi}{3}, 2\pi \equiv 0\right) \quad \text{rad}$$

Recoding of the output variables requires a little more insight. The inner limits of the recoding levels that were used to recode the shaken output data in the optimal mask analysis should be used here as well, since we are going to use those masks. The outer limits should be adjusted so that every measured data point belongs to one level. This is important, since the recoding function will not recognize values outside the limits specified in the *from*-matrices.

Now with the data model composed by the recoded data matrices and the system's optimal masks, we can use GSPS's inductive reasoning feature (the *forecast*-function of SAPS-II) to see how well it guesses which states the output variables will assume in the ten following sampling intervals, given a certain set of inputs for those data points.

We will use the user defined CTRL-C/SAPS function FRC (its code has been listed in [2]) which is a forecasting function coded specially for this type of problem. It uses the standard SAPS-II *forecast*-function three times for every time instant, once with each optimal mask, to forecast the state assumed by each output variable at that time instant. Forecasting will only be terminated when states for all three outputs have been forecast for the requested 100 steps; the minimum acceptable probability associated with the forecasting is set to zero.

Comparing an excerpt of the measured (and recoded) data (FUTURE) with the same excerpt of the forecast data (FRCST) e.g. for the data points 2551:2560, the following results are obtained:

```
future =  2.  2.  1.     frcst =  2.  2.  1.     error =  0.  0.  0.
          2.  2.  1.              2.  2.  1.              0.  0.  0.
          2.  2.  2.              2.  2.  2.              0.  0.  0.
          2.  1.  2.              2.  1.  2.              0.  0.  0.
          2.  1.  2.              2.  1.  1.              0.  0.  1.
          2.  1.  1.              2.  1.  1.              0.  0.  0.
          2.  1.  1.              2.  1.  1.              0.  0.  0.
          2.  1.  1.              2.  1.  1.              0.  0.  0.
          2.  2.  1.              2.  2.  1.              0.  0.  0.
          3.  3.  1.              3.  3.  1.              0.  0.  0.
```

Both the first and the second output variable were forecast *without a single error* (within the resolution of the discretized values). Only the third output variable (γ) contains one incorrect prediction.

More details about the process of determining the best optimal masks, and about the forecasting process are presented in [14].

Forecasting Behavior of Open Integrator Variables

One of the major weaknesses and, paradoxically, also one of the major strengths of the GSPS methodology is the fact that the forecasting process will come to a stop as soon as a system state has been reached that has never been seen before. In such a case, GSPS has no way of predicting what the next state might look like.

This creates considerable difficulties if one of the variables to be predicted is the output of an open integrator (such as the altitude of the aircraft in our example). The smallest disturbance will either increase or decrease the cruise altitude of the aircraft, and thereby jeopardize the forecasting capability of our tool.

This is a weakness since, all too often, the forecasting process indeed comes to an end, and we have to be quite inventive to convince SAPS-II to forecast any behavior beyond this point. This is also a strength since the model validation process is an intrinsic part of the simulation.

For comparison, consider Forrester's World Model [6]. System Dynamics, an alternative modeling methodology, has no scruples whatsoever in "predicting" the behavior of Forrester's "World" at a pollution level which is 100 times higher than any value that has ever been recorded on this planet. SAPS-II will strictly reject succumbing to such a temptation.

There are several ways in which we can cope with this difficulty. We may be able to circumvent the problem by selecting another measurement variable into our source model as a replacement for the open integrator variable. A good candidate might be the derivative of the open integrator. Most engineering systems don't exhibit two open integrators in a row. Alternatively, the open integrator variable can be filtered prior to its further use. SAPS-II allows to compute a pseudo-derivative of any variable through the use of its *diff*-function which shifts through the measurement vector, and successively computes the difference between neighboring elements. Another filter might be to compute a moving average of several elements in a measurement vector, and subtract this moving average from the measurement data. In SAPS-II, this can be accomplished by use of the *average*-function. It was the latter approach that has been adopted in our example to get rid of problems caused by altered steady-state levels.

THE BROKEN MODEL

The optimal mask analysis was repeated for a number of situations where one or the other type of "damage" had occurred. For each type of damage, we received one set of three optimal masks representing the behavior of the aircraft under the influence of this damage.

The ultimate idea was to simulate the occurrence of a damage in full flight. Figure 6 shows an excerpt of the data recorded from such an ACSL run for the flight path angle shortly before and shortly after the accident occurred. Notice that the momentary mean value has been subtracted already before this curve was drawn. The accident is marked by a vehement shock from which the aircraft recovers quickly. The recovery takes roughly 20 steps or 2 minutes. The shape of the curve as shown for the next 5 minutes of flight following the high frequency oscillation is an artifact resulting from the way in which the moving average was constructed. In our program, we used 50 steps (5 minutes) for the computation of the moving average, i.e., the data are still distorted for the next 5 minutes after the aircraft has already recovered from the accident.

For a human eye, the occurrence of the damage is quite visible from the graph. However, this would not have been true, had we just looked at the tabular values instead of the graph. Since that is really what the computer algorithm does, it is not so obvious how we can get our inductive reasoning mechanism to pick up and correctly interpret the damage as soon as possible once it occurred.

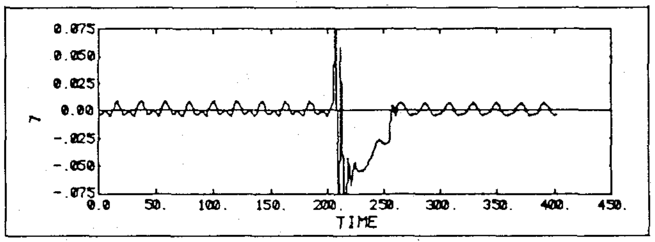


Figure 6. Excerpt of a trajectory in the neighborhood of the accident

The following approach was taken. By constantly comparing the observed (and recoded) values with their forecast counterparts, we should be able to register the accident since, following the accident, the mask no longer optimally represents the aircraft, and therefore, we expect to see a mismatch between the registered and the forecast values. The following CTRL-C code is used as a "data filter" applied to the error matrix between the "measured" and the predicted behavior of the aircraft.

```
[> error    = ABS(future - frcst);
[> error    = error*ONES(3,1);
[> smoothed = AVERAGE(error,3);
[> normed   = 1.49*smoothed/6;
[> filter   = ROUND(normed);
```

Figure 7 shows graphically the output of this data filter for two separate time periods, namely for the neighborhood of the accident, and again a couple of minutes later after the aircraft has stabilized itself.

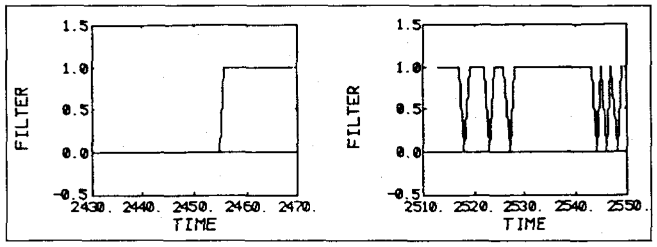


Figure 7. Output of the error signal data filter

The accident occurred at step 2452. As can be seen, the data filter operates very effectively on the error signal, allowing us to conclude after very few steps only (about 5 steps, i.e., 30 seconds at the latest) with certainty that *something* must have gone wrong with the aircraft. During the initial phase after the accident (i.e., during the oscillatory period), the forecasting process fails entirely (as expected). However, even after the aircraft has stabilized itself again (i.e., during a period where the output of the aircraft looks quite similar to the output from the period before the accident took place, the forecasting procedure still picks up the difference in the behavioral pattern that the aircraft now exhibits.

In the next step, we will, in parallel, apply all other previously established optimal masks to the measured data to see whether any of them produces a good match (i.e., a low signal at the output of the data filter) with the measured (and recoded) data. If this is the case, we reason that, with high probability, we have identified the type of accident that has occurred. If this is not the case, we had better wake up the human pilot and ask him to take over from there.

The set task has been accomplished with very good success. Applying the optimal mask analysis to the data *after* the aircraft has recovered from the accident leads to a different set of optimal masks than the one found for the original aircraft. Using this set of optimal masks in the forecasting process leads to a good and reliable prediction of the aircraft's behavior under the faulty flight conditions. Had this set of optimal masks been recorded beforehand, it could have been used to correctly identify the cause of the problem. It takes roughly 50 steps (i.e. 5 minutes) to come up with a decent hypothesis about the type of accident that has taken place. This period is related to the recovery of the aircraft. We must wait until the aircraft has stabilized itself again before we can successfully analyze the cause of the problem using our methodology.

Once a match has been found, we can do one of two things.

1. We can use this information to let the human pilot know what we found out about the cause of the problem, i.e., we can use this information for a "consultation system," or
2. we can use this information as an input to an "adaptive" control system which will exchange the previously used (conventional) autopilot for another modified (but still conventional) autopilot that has been optimized to fly the aircraft after an accident of the hypothesized type has occurred. From now on, SAPS-II will use the optimal masks for this type of damage to compare the measured (and recoded) data from the quantitative aircraft simulation with predicted qualitative values evaluated by shifting the optimal mask over the measured data matrix.

CONCLUSIONS

While the proposed methodology does not lend itself yet to adaptive on-line control of unforeseen types of accidents (because of the very long time delays that are imposed on the evaluation of a new behavior model), it was possible to aid a conventional autopilot in its global decision making process. This is quite feasible and practical, and could be adapted to a real autopilot of a real aircraft.

In order to truly solve the on-line control problem, it would however be necessary to feed the model search algorithm with more specific information about the type of process (the aircraft) that is to be controlled. In this way, the amount of time necessary to come up with a new "mental" model of the damaged aircraft could be drastically reduced, and could hopefully be made similar in length to the time needed by a human pilot to come up with an appropriate action plan.

REFERENCES

[1] Cellier, F. E. and D. W. Yandell (1987). "SAPS-II: A New Implementation of the Systems Approach Problem Solver," *International J. of General Systems,* 13(4), pp 307-322.

[2] Cellier, F. E. (1987). "Qualitative Simulation of Technical Systems Using the General System Problem Solving Framework," *International J. of General Systems,* 13(4), pp 333-344.

[3] Etkin, B. (1972). *Dynamics of Atmospheric Flight,* John Wiley & Sons, Inc.

[4] Etkin, B. (1982). *Dynamics of Flight,* John Wiley & Sons, Inc.

[5] Etkin, B. and S. Zhu (1982). *Control Logic for Landing-Abort Autopilot Mode,* UTIAS Report No. 258, Institute for Aerospace Studies, University of Toronto, Canada.

[6] Forrester, J. W. (1971). *World Dynamics,* Wright Allen Press.

[7] Hacker, T. (1970). *Flight Stability and Control,* American Elsevier Publishing Company, Inc.

[8] Heffley, R. K. *et alia* (1972). *Aircraft Handling Qualities Data,* Report NASA-CR-2144, Systems Technology, Inc.

[9] Irving, F. G. (1966). *An Introduction to the Longitudinal Static Flight of Low-Speed Aircraft,* Pergamon Press

[10] Klir, G. J. (1985). *Architecture of Systems Problem Solving,* Plenum Press, New York.

[11] Law, A. M. and W. D. Kelton (1982). *Simulation Modeling and Analysis,* McGraw Hill, New York, N.Y.

[12] Mitchell, E. E. L. and J. S. Gauthier (1986). *ACSL: Advanced Continuous Simulation Language-User/Guide Reference Manual,* Mitchell & Gauthier, Assoc., Concord, MA.

[13] Systems Control Technology (1984). *CTRL-C, A Language for the Computer-Aided Design of Multivariable Control Systems, User's Guide, Systems* Control Technology, Palo Alto, CA.

[14] Vesantera, P. (1988). *Qualitative Flight Simulation: A Tool for Global Decision Making,* MS Thesis, Dept. of Electrical & Computer Engineering, University of Arizona, Tucson, AZ.