Building Hybrid Automata of Complex Physical Systems for Real-time Applications

Pieter J. Mosterman[1]
Institute of Robotics and System Dynamics
DLR Oberpfaffenhofen
D-82230 Wessling
Pieter.J.Mosterman@dlr.de

Gautam Biswas[2]
Dept. of Electrical Engineering and Computer Science
Vanderbilt University
Nashville, TN 37235
biswas@vuse.vanderbilt.edu

Abstract

Dynamic behavior of complex physical systems is often nonlinear and includes multiple temporal scales. *Singular perturbation* methods decouple the fast and slow behavior and by assuming that the fast behavior is at a quasi steady state, the slow behavior of the system can be analyzed. The decoupling reduces the complex system of ordinary differential equations (ODEs) to simpler ODEs that allow fixed time step integration methods, and, therefore, are suitable for real-time applications. This model reduction may cause discontinuous jumps in the initial values of model variables. This paper applies and extends singular perturbation methods to compute discontinuous state changes for piecewise continuous, *hybrid*, models when the model configuration changes. Computing the explicit state change allows the use of hybrid automata as modeling framework when augmented with execution semantics for state vector updates around discontinuities.

1 Introduction

The pressure to achieve optimal performance and meet rigorous safety standards in industrial processes, aircraft, and nuclear plants, necessitates more detailed modeling and analysis of these systems. Complex systems exhibit nonlinearities attributed to small parameters that manifest as behaviors on very fast time scales that complicate this task. In case of numerical simulation, sophisticated algorithms vary their time step to accommodate multi time scale behaviors, but the variable step size makes it hard to bound their runtime computational complexity. This makes them unsuitable for real-time applications. As an alternative, *hybrid* modeling methodologies represent system behavior as piecewise continuous modes interspersed with discrete transitions. This facilitates the use of fixed time step integration methods to numerically simulate the continuous dynamics, whereas the fast dynamics are replaced by instantaneous changes, allowing real-time simulation.

The primary flight control system of aircraft (illustrated in Fig. 1) demonstrates the paradigm for hybrid, i.e., mixed continuous/discrete, modeling of embedded control systems. At the lowest level in the control hierarchy, continuous PID control moves the rudder, elevators, and ailerons to desired set positions. On a higher level, digital control may mandate *mode* changes at different stages of a flight plan (e.g., *take-off*, *cruise*, *go-around*) and detection of failures may lead to discrete changes in system configuration. Furthermore, abstracting fast, nonlinear transients may produce discontinuous variable changes.

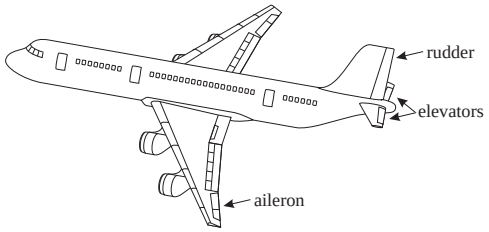

Figure 1: Primary aerodynamic control surfaces.

The *singular perturbation* approach [4] abstracts detailed continuous behavior represented as a system of complex nonlinear ordinary differential equations, cODE, into piecewise simpler sODEs. Our goal is to extend and generalize this approach to apply to hybrid systems that include one or a number of, possibly consecutive, discrete mode changes. We have shown [6, 8] that small time constant effects cannot always be removed (*parameter* abstractions) in analyzing dynamic system behavior. Abstracting fast transients may require explicit additional jump constraints for the system state vector variable values when configuration changes occur (*time scale* abstraction).

This paper presents the effects of abstracting fast continuous transients exhibited by complex systems into

[1] Pieter J. Mosterman is supported by a grant from the DFG Schwerpunktprogramm KONDISK.
[2] Gautam Biswas is supported by grants from Hewlett-Packard, Co.

discontinuous changes of the continuous state vector. It demonstrates a systematic methodology for generating the simpler ODE models from the more complex ODE models of system behavior. The simpler piecewise ODE models are then compiled into *hybrid automata* [1] with extended execution semantics to facilitate efficient real-time applications.

2 Hybrid Dynamic Systems

Hybrid dynamic systems combine discrete state changes with continuous behavior evolution [1, 5, 8]. Differential equations form a common representation of continuous system behavior. The system is described by a *state vector*, x. Behavior over time is specified by a field f. Interaction with the environment is specified by *input* and *output* signals, u and y. The dynamics of system behavior is expressed as a set of ODEs, $\dot{x} = f(x, u)$ and algebraic relations $y = h(x)$.

Discrete behavior can be modeled by a state machine representation, consisting of a set of discrete modes, α. Mode changes caused by events, σ, are specified by the *state transition function* ϕ, i.e., $\alpha_{i+1} = \phi(\alpha_i)$. A transition may produce additional discrete events, causing further transitions.

A mode change from α_i to α_{i+1}, may result in a field definition change from f_{α_i} to $f_{\alpha_{i+1}}$, and a discontinuous change in the state vector governed by an algebraic function g, $x^+ = g_{\alpha_i}^{\alpha_{i+1}}(x)$. Discrete mode changes are caused by an *event generation function*, γ, associated with the current active mode, α_i, $\gamma_{\alpha_i}(x) \leq 0 \rightarrow \sigma_j$.

3 Abstracting Fast Transients

Continuous behavior in physical systems can occur on a hierarchy of temporal and spatial scales. To simplify system models, small[1] parameters are abstracted away but this may cause discontinuous changes in system behavior. These discontinuous changes are present as fast continuous transients in the more detailed models. In this section, we formalize the derivation of simplified models generated by parameter and time scale abstractions and derive explicit discontinuous state changes to facilitate the use of the simpler models in a hybrid automata framework.

3.1 Parameter Abstraction

Parameter abstractions eliminate small parameters in a system model. A singular perturbation representation formulates the system behavior model into a cODE with two time scales:

$$\begin{cases} \dot{x} = f(x, z, \epsilon, t), \ x(t_0) = x^0, \ x \in \Re^n, \\ \epsilon \dot{z} = g(x, z, \epsilon, t), \ z(t_0) = z^0, \ x \in \Re^m, \end{cases} \quad (1)$$

where ϵ is a small parameter that embodies small and large parameter values that cause fast transients. The function f models the slower dominant system behavior. Setting ϵ to 0 reduces the second equation to an algebraic form. Assuming that $g(x, z, 0, t) = 0$ has distinct real roots, the fast behaviors corresponding to z can be solved for algebraically, and substituted in f. This results in a reduced-order *quasi steady state* model that embodies an sODE.

We apply this approach to the collision between two bodies shown in Fig 2. A first order approximation of the collision process includes two parameters: (i) C, that models the elastic interaction between the bodies, and (ii) R, that models the dissipative effects. If the momentum, p_i, of the bodies, m_i, and the displacement, q, of the spring that models the elasticity parameter C, are chosen as state variables, the dynamic behavior of the system is described by the following ODE:

$$\begin{cases} \dot{p}_1 = -\frac{q}{C} - R(\frac{p_1}{m_1} - \frac{p_2}{m_2}) \\ \dot{p}_2 = \frac{q}{C} + R(\frac{p_1}{m_1} - \frac{p_2}{m_2}) \\ \dot{q} = \frac{p_1}{m_1} - \frac{p_2}{m_2}. \end{cases} \quad (2)$$

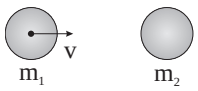

Figure 2: Collision of a body m_1 and a body m_2.

This singular system of equations can be reduced by applying the transformation $v = \frac{p_1}{m_1} - \frac{p_2}{m_2}$, resulting in a second order ODE

$$\begin{bmatrix} \dot{v} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} -R(\frac{1}{m_1} + \frac{1}{m_2}) & -\frac{1}{C}(\frac{1}{m_1} + \frac{1}{m_2}) \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ q \end{bmatrix}. \quad (3)$$

If the R and C parameters are removed from the model, a simpler system of equations results, but the state variables may exhibit explicit discontinuous jumps. The hybrid automata model requires explicit definition of these jumps, and necessitates their computation from the detailed continuous transients.

To apply singular perturbations, we assume C to be small and R to be large and take $\frac{1}{R}$ to be the small ϵ parameter. Therefore,

$$\begin{bmatrix} \frac{1}{R}\dot{v} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} -(\frac{1}{m_1} + \frac{1}{m_2}) & -\frac{1}{RC}(\frac{1}{m_1} + \frac{1}{m_2}) \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ q \end{bmatrix}, \quad (4)$$

where v contains the fast behavior. Substituting $\frac{1}{R} = 0$ results in $v = 0$. Transforming this back to the original state variables yields $\frac{p_1}{m_1} - \frac{p_2}{m_2} = 0$, i.e., $v_1 - v_2 = 0$. This is the equivalent of a perfect non elastic collision [6].

[1] Also, large, because the reciprocal of a large parameter value is small.

3.2 Time Scale Abstraction

Instead of eliminating the fast transient due to dissipative effects, if we were to reduce the effect of elasticity to occur at a point in time, we get a time scale abstraction. Consider the system of colliding bodies again (Fig. 2) with detailed behavior given by Eq. (3). If C is taken to be the small ϵ parameter, this gives

$$\begin{bmatrix} C\dot{v} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} -RC(\frac{1}{m_1}+\frac{1}{m_2}) & (\frac{1}{m_1}+\frac{1}{m_2}) \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ q \end{bmatrix}, \tag{5}$$

For $\epsilon = 0$, this yields $q = 0$, and, therefore, $\dot{q} = 0$ which requires $v = 0$. When C becomes small but not 0, the solution of the system in Eq. (3) has eigenvalues with imaginary components and the resultant dynamic behavior for the transient is:

$$v(t) = v(0)e^{-\frac{R}{2}(\frac{1}{m_1}+\frac{1}{m_2})t} cos((\sqrt{\frac{4}{C} - R^2(\frac{1}{m_1}+\frac{1}{m_2})^2})t). \tag{6}$$

This shows that $v = 0$ is the steady state solution. However, in case of colliding bodies this behavior transient is aborted long before steady state is attained, because the v and q values generated by the transient cause the two bodies to disconnect.

This is illustrated by the case of two point masses that collide when $x_1 \geq x_2$, where x_1 and x_2 are the positions of body m_1 and m_2, respectively. The bodies disconnect when the force between them becomes negative, i.e., $F_{12} < 0$. At this point, the state variable values (i.e., the two body velocities) constitute the final, *a posteriori*, values around the discontinuous jump corresponding to the collision. Since $F_{12} = \frac{q}{C} < 0$ at the disconnect point, this implies $q < 0$ since $C > 0$. The time point at which the disconnect occurs is computed to be

$$t_d = \frac{\pi}{\sqrt{\frac{4}{C} - R^2(\frac{1}{m_1}+\frac{1}{m_2})}}. \tag{7}$$

At t_d, v has changed from $v(0)$ to $v(t_d) = \lambda v(0)$ with $(cos(\pi) = -1)$, therefore,

$$\lambda = -e^{-\frac{R}{2}(\frac{1}{m_1}+\frac{1}{m_2})t_d} \tag{8}$$

As the C parameter becomes very small, t_d does too, and in the limit, $v(t_d) \rightarrow v(0)^+$. The discontinuous change in v can then be represented by an algebraic equation $v(0)^+ = \lambda v(0)$. Transforming this back to the original state variables yields $\frac{p_1^+}{m_1} - \frac{p_2^+}{m_2} = \lambda(\frac{p_1}{m_1} - \frac{p_2}{m_2})$. Written in terms of the body velocities,

$$v_1^+ - v_2^+ = \lambda(v_1 - v_2), \tag{9}$$

this is the well known Newton's collision rule [2], where λ is called the coefficient of restitution that describes the amount of kinetic energy loss in the collision. If $R = 0$ in Eq. (8), $\lambda = -1$ and this describes a perfect elastic collision with no loss of energy. Note that C cannot be taken to equal 0, as this would remove all elasticity and the corresponding ideal rigid body collision has no mechanism for storing kinetic energy as potential energy and returning it as kinetic energy. Therefore, this immediately causes $v = 0$. Consequently, behavior does not converge uniformly as $C \rightarrow 0$.

4 The Elevator System

The elevator control subsystem in Fig. 1 consists of two mechanical elevators that are positioned by two electro-hydraulic actuators each [3, 9]. When a failure occurs, redundancy management may switch actuators to ensure maximum control.

Figure 3 shows the operation of one actuator. The continuous PID control mechanism for elevator positioning is implemented by a servo valve. The output of the servo valve controls the direction and speed of travel of the piston in the cylinder. A spool valve mechanism controls whether the actuator is *active* or *passive*. The piston in the positioning cylinder and connected elevator flap constitute the load and excessive pressures are prevented by a pressure relief valve.

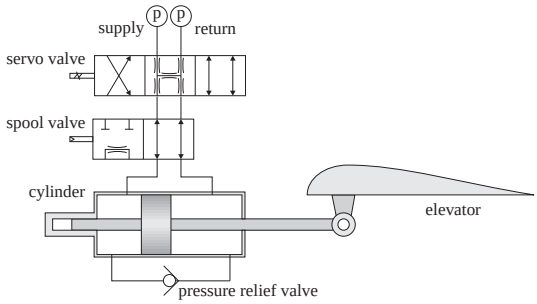

Figure 3: Hydraulics of one actuator.

4.1 The Components

The servo valve consists of a cylinder that connects its supply and return side with its loading side. A piston inside the cylinder controls the amount of oil flow from supply to loading. The amount of oil flowing in, q_s, has to equal the amount of oil flowing out q_l. This oil flow is determined by the pressure drop, $p_s - p_l$, across the orifice that is opened by an amount x, the PID control variable,

$$\begin{cases} q_s = (p_s - p_l)x \\ q_s = q_l \end{cases} \tag{10}$$

Like a servo valve, a typical spool valve (Fig. 4) consists of a piston that moves in a cylinder. A number of cylinder ports connect the supply and return part of the hydraulic system with the load. When the actuator is *active*, the spool valve is in its *supply* mode,

α_2, shown in Fig. 4(a), and the control signal generated by the servo valve is transferred to the cylinder that positions the elevator. In this mode, the pressure on the supply side of the valve, p_s, equals the pressure on the load side, p_l. Also, the oil flow from the supply, q_s, equals the oil flow to the load, q_l. When the actuator is *passive*, the spool valve is in its *loading* mode, α_0, shown in Fig. 4(c), and control signals cannot be transferred to the cylinder. However, oil flow between the chambers is possible through a loading passageway with fluid flow resistance R_l. When moving between *supply* and *loading*, the spool valve passes through the *closed* configuration, α_1, where oil flow is blocked, as shown in Fig. 4(b). This is captured by the following equations:

$$\alpha_2 : \begin{cases} p_s = p_l \\ q_s = q_l \end{cases} \quad \alpha_1 : \begin{cases} q_l = 0 \\ q_s = 0 \end{cases} \quad \alpha_0 : \begin{cases} p_l = q_l R_l \\ q_s = 0 \end{cases} \tag{11}$$

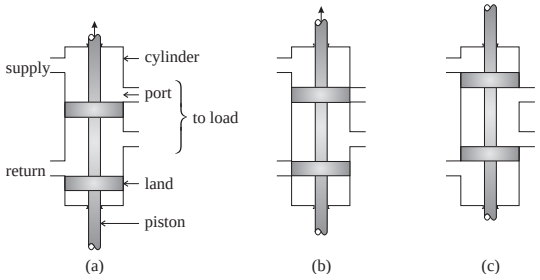

Figure 4: A typical spool valve.

A pressure relief valve connected to the positioning cylinder as a safety device is normally closed (α_0), but it may open (α_1) when the pressure in the elevator positioning cylinder, i.e., the input pressure to the relief valve, p_r, exceeds a threshold value, p_{th}. When open, it allows an oil flow, q_r, through a fluid path with resistance R_l:

$$\alpha_0 : \begin{cases} q_r = 0 \end{cases} \quad \alpha_1 : \begin{cases} p_r = q_r R_l \end{cases} \tag{12}$$

4.2 Modeling the Elevator Dynamics

The dynamics of the elevator are studied in terms of the movement of the piston in the positioning cylinder. The behavior can be derived by composing models of the servo valve, spool valve, relief valve, and the positioning cylinder. This results in a second order system with two state variables: (i) p_c, the pressure of the oil in the cylinder, and (ii) v_e, the elevator velocity.

$$\begin{cases} C_c \dot{p}_c = q_{in} + q_r - q_e \\ q_e = A_p v_e \\ A_p F_e = p_c + R_c(q_{in} + q_r - q_e) \\ m_e \dot{v}_e = F_e \end{cases} \tag{13}$$

C_c models the elasticity effects and R_c models the dissipative effects of the oil in the positioning cylinder. The variables q_{in} and q_r represent the inflow of oil into the cylinder from the servo and relief valves, respectively, and q_e represents the oil flow due to movement of the piston. The value of q_e is a function of A_p, the area of the piston and v_e, the elevator velocity. The force exerted on the piston is a function of p_c, and the product of internal dissipation of the oil, R_c, and the overall flow rate. Newton's Second Law relates the elevator velocity to the force exerted on the piston. In state equation form Eq. (13) is:

$$\begin{bmatrix} \dot{p_c} \\ \dot{v_e} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{A_p}{C_c} \\ \frac{1}{m_e A_p} & -\frac{R_c}{m_e} \end{bmatrix} \begin{bmatrix} p_c \\ v_e \end{bmatrix} + \begin{bmatrix} \frac{1}{C_c} & \frac{1}{C_c} \\ \frac{R_c}{m_e A_p} & \frac{R_c}{m_e A_p} \end{bmatrix} \begin{bmatrix} q_{in} \\ q_r \end{bmatrix}. \tag{14}$$

When a sudden pressure drop is detected in the hydraulics supply system of an elevator actuator, redundancy control moves the spool valve of this actuator from *supply* to *loading* and the spool valve of another actuator from *loading* to *supply* to take over control. When the spool valve of an actuator moves to its *closed* mode, oil flow into and out of the positioning cylinder is blocked. This implies that the cylinder piston that controls the elevator position cannot move, and the elevator stops moving as well. In more detail, the internal dissipation and small elasticity parameters of the oil cause the elevator velocity to change continuously during the transition. The continuous transient behavior between *supply* and *closed* is shown in Fig. 5(a). How quickly the system reaches 0 velocity in the *closed* mode depends on the elasticity and internal dissipation parameters of the oil. Typically, soon after the *closed* mode, the spool valve starts opening and goes into the *loading* mode. The effect on the elevator velocity for the detailed continuous behavior when switching from *supply* to *loading* is shown in Fig. 5(b). A detailed analysis is presented in [7].

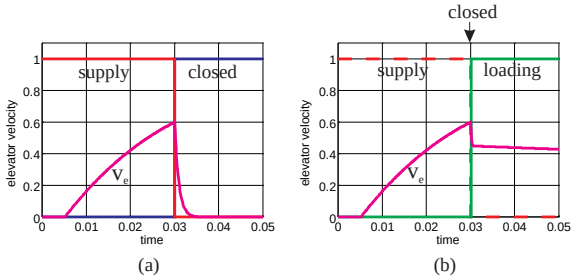

Figure 5: Continuous transients.

The details of the continuous transients are not of much interest for analysis of the control behavior. Model simplification by parameter and time scale abstractions results in removal of small elasticity and large dissipative effects. At the same time, the fast continuous transients that affect the elevator velocity, v_e, need to be preserved and captured by discontinuous changes. These

discontinuities indicate that state variable dependencies arise between the model component ODEs. This requires model manipulations to generate a global ODE or methods to handle the resulting high index differential and algebraic equations (DAE).

We systematically derive the simpler models for the hybrid automata and the transition conditions using the methods based on singular perturbation described in Section 3 and replace the detailed continuous transients defined by Eq. (14) by an equation that captures the fast continuous change as an instantaneous discontinuous jump. We analyze the transient about the point where the spool valve closes, and the relief valve is also closed, i.e., $q_{in} = q_r = 0$.

In case of real eigenvalues, the elevator dynamics can be computed to be ($D = \frac{{R_c}^2}{{m_e}^2} - \frac{4}{m_e C_c}$)

$$v_e(t) = e^{-\frac{R_c}{2m_e}t}(k_1 e^{\frac{1}{2}(\sqrt{D})t} + k_2 e^{-\frac{1}{2}(\sqrt{D})t}), \qquad (15)$$

where k_1 and k_2 are constants that depend on $v_e(0)$ and $p_c(0)$. Like before, the restitution coefficient for the oil, affected by the spool valve closing, i.e., λ_s, can be computed by determining the value of t_d at the point when the ports are opened again. If x is the displacement of the piston in the spool valve, the piston may first block the ports when $x = 0$ and open them again when $x > x_{th}$, where x_{th} is a parameter depending on the particular type of spool valve. The value of t_d is then determined by x_{th} and the speed with which the piston is moved by an external control signal. The corresponding time interval during which the oil flow into the cylinder is 0 results in an elevator velocity change as a function of $v_e(0)$ and $p_c(0)$.

In case of complex eigenvalues, the elevator dynamic behavior is governed by

$$v_e(t) = e^{-\frac{R_c}{2m_e}t}(k_1 cos(\frac{1}{2}(\sqrt{-D})t) + k_2 sin(\frac{1}{2}(\sqrt{-D})t)) \qquad (16)$$

where k_1 and k_2 are constants depending on $v_e(0)$ and $p_c(0)$. Again, the change of elevator velocity at t_d can be computed as a function of $v_e(0)$ and $p_c(0)$. In this case, the elevator velocity may reverse much like the velocity of a bouncing ball reverses.

4.3 A Scenario

Figure 6 defines the individual hybrid automata for the spool valve, the positioning cylinder, and the relief valve. When the spool valve goes from *supply* mode (α_2) to *closed* mode (α_1), causing q_s, and, therefore, q_{in}, in the positioning cylinder to change discontinuously, the fast transient that affects v_e can be simplified by parameter and time scale abstraction, and v_e goes through an instantaneous change in velocity given by ${v_e}^+ = \lambda_s v_e$.

To compute the reduced continuous behavior model, the oil is assumed to be incompressible, the corresponding simplified ODE for elevator velocity in the positioning cylinder is calculated by setting $C_c = 0$:

$$\begin{cases} 0 = q_{in} + q_r - q_e \\ q_e = A_p v_e \\ A_p F_e = p_c \\ m_e \dot{v}_e = F_e \end{cases} \qquad (17)$$

This sODE is first order, whereas the cODE was second order.

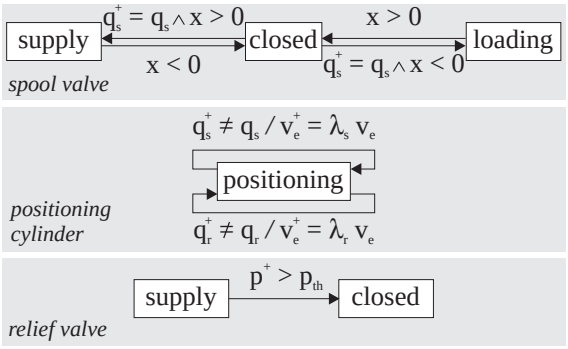

Figure 6: Individual hybrid automata.

In this model, because the behavior of the spool valve around $x = 0$ is abstracted away, the spool valve switches into its *closed* mode when the piston in the valve reaches 0 from the right, $x < 0$, or from the left, $x > 0$. Now, $q_s^+ = 0$ and $q_s^+ \neq q_s$, causing a discontinuous change in v_e^+. Immediately after the discontinuous changes are effected, $q_s^+ = q_s$, and the spool valve switches out of the *closed* mode. Note that the abrupt change in velocity from v_e to ${v_e}^+$ will cause a fast pressure buildup. In the reduced order model, this buildup is governed by a discontinuous change of v_e, and, therefore, $v_e^+ \neq v_e$. The $m_e \dot{v}_e = F_e$ equation causes an impulse force, F_e, and corresponding pressure p_e.

This pressure impulse will always cause the relief valve to open because of its infinite magnitude, no matter how small the $v_e^+ - v_e$ difference. The more detailed model of the cylinder includes small elasticity and dissipation parameters, and they are employed to compute a more realistic value of the maximum pressure generated. This can be included in the reduced order model by replacing the $m_e \dot{v}_e = F_e$ equation with the algebraic constraint $K_c(v_e^+ - v_e)$ providing the value for F_e. K_c is a damping coefficient that captures the $R_c C_c$ effect. Using this first order approximation, the pressure buildup can be described as $p_c^+ = A_p K_c({v_e}^+ - v_e)$,

If the value of p_c^+ exceeds the critical value, p_{th}, this causes a further discontinuous mode change in the relief valve, that goes from closed (α_0) to open (α_1). In this case, the abrupt change in elevator velocity is governed

by a restitution coefficient, λ_r, defined by the complex ODE model of the relief valve that can be derived in a manner similar to the derivation for the spool valve. The final elevator velocity after the mode transitions is now given by ${v_e}^+ = \lambda_r v_e$. The simplified ODE model for v_e in the supply mode with relief valve *open* can be derived similarly.

5 The Actuator Hybrid Automata

If the relief valve opens, the effect of λ_s on the elevator velocity is replaced by λ_r, and, therefore, the change of velocity as computed by λ_s has to be reversed. A hybrid automata that correctly includes this effect is shown in Fig. 7. The modes are α_{ij}, where the subscript i, represents the mode of the spool valve (2 - open, 1 - closed, and 0 - loading), and subscript j represents the mode of the relief valve (1 - open, and 0 - closed). The corresponding sODEs are also subscripted accordingly. Initially, the actuator is in mode α_{20}. In the simplified hybrid automata, the detailed continuous behavior around $x = 0$ is abstracted away, and the corresponding discrete events, $\{\sigma_{close}, \sigma_{spool}, \sigma_{load}, \sigma_{relief}\}$ are generated by monitoring physical variables. The stroked transitions in Fig. 7 represent transitions where the event generation function, γ, is applied after the state vector has been updated.

When in α_{20}, closing the spool valve moves the system into α_{10} and causes an instantaneous change in the oil flow rate to 0. Therefore, $q_s^+ \neq q_s$ and a rapid drop in the elevator velocity, ${v_e}^+ = \lambda_s v_e$, occurs before the valve opens again and goes into the loading mode, α_{00}. However, the change in velocity causes a pressure transient, $p^+ = A_p K_c({v_e}^+ - v_e)$, and if $p^+ > p_{th}$, σ_{relief} is generated causing the relief valve to open, and the system goes into mode α_{11}, with ${v_e}^+ = \lambda_r v_e$. Therefore, ${v_e}^+ = \lambda_s v_e$ is not executed and v_e^+ not affected by mode α_{10}. Once the state vector is updated, ${q_s}^+ = q_s$, and σ_{load} is generated causing the system to go into α_{01}. If σ_{relief} did not occur, ${v_e}^+ = \lambda_s v_e$ remains valid, and after the state vector is updated ${q_s}^+ = q_s$ and the mode transition to α_{00} occurs based on the event σ_{load}.

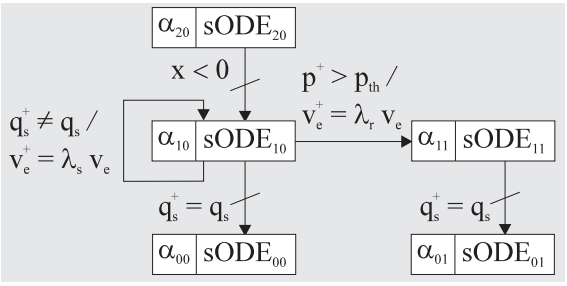

Figure 7: Hybrid automata of an actuator.

6 Conclusions

In this paper we have developed a systematic methodology derived from the singular perturbations approach for generating simpler ODE models for hybrid systems by applying time scale and parameter abstractions to complex nonlinear system models that exhibit fast transient behavior. Compiling the sODEs and transition conditions into hybrid automata with extended execution semantics generates runtime models that can be applied to real-time simulation and analysis of system behavior.

The apparent drawback of creating piecewise simpler hybrid models is that the compositionality property is lost when interactions between the component subsystems have to be analyzed in advance to build the hybrid automata.

References

[1] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Lecture Notes in Computer Science*, vol. 736, pp. 209–229. Springer-Verlag, 1993.

[2] R. M. Brach. *Mechanical Impact Dynamics*. John Wiley and Sons, New York, 1991.

[3] W. L. Green. *Aircraft Hydraulic Systems*. John Wiley, Chichester, UK, 1985.

[4] P. V. Kokotović, H. K. Khalil, and J. O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design*. Academic Press, London, 1986.

[5] B. Lennartson, M. Tittus, B. Egardt, and S. Pettersson. Hybrid systems in process control. *IEEE Control Systems*, pp. 45–56, Oct. 1996.

[6] P. J. Mosterman and G. Biswas. A theory of discontinuities in dynamic physical systems. *Journal of the Franklin Institute*, 335B(3):401–439, Jan. 1998.

[7] P. J. Mosterman and G. Biswas. Building Hybrid Observers for Complex Dynamic Systems using Model Abstractions. In *Hybrid Systems: Computation and Control*, vol. 1569, pp. 178–192, 1999.

[8] P. J. Mosterman, G. Biswas, and J. Sztipanovits. A hybrid modeling and verification paradigm for embedded control systems. *Control Engineering Practice*, (6):511–521, 1998.

[9] J. Seebeck. *Modellierung der Redundanzverwaltung von Flugzeugen am Beispiel des ATD durch Petrinetze und Umsetzung der Schaltlogik in C-Code zur Simulationssteuerung*. Diplomarbeit, Technische Universität Hamburg-Harburg, 1998.