2009-01-1041

Robust Design of Control Systems with Physical System Variances

Tom Egel
The MathWorks, Inc.

ABSTRACT

Today's automotive control system engineering requires precision and accuracy. The cost of a controller designed with conservative margins may increase significantly, causing the design, when produced and marketed, to be less competitive. On the other hand, a design with too little margin may lead to system malfunction under marginal environment conditions or due to component aging. A robust design is one that is immune to the effects of component variance due to tolerance, temperature, and aging, among other factors. Achieving a robust design involves careful analysis of the controller and plant operating together. This paper discusses how MATLAB and Simulink can be leveraged to ensure the robustness of a mechatronic system design. The merits of the network approach as a technique for modeling physical systems as an alternative to the signal flow (block diagram) approach are also discussed. Finally, the advantages of integrating these methods within Simulink as the environment for Model-Based Design for mechatronic systems are presented. An example involving Monte Carlo simulation on a simple multidomain mechatronic control system is used as a case study.

INTRODUCTION

Mechatronics[1] is a commonly used term for describing the combination of electromechanical physical systems with computer controls (see Figure 1). Designers of embedded controls for mechatronic systems face difficult challenges. Completion of a successful mechatronic system design requires the integration of multiple engineering domains and collaboration between the engineering teams. For example, exhaustively testing the software control algorithm for an antilock brake system requires accurately representing the physics of the electronics, hydraulics, and mechanics. In addition, as embedded controls continue to become more and more part of the core functionality of the modern automobile, time-to-market pressures, cost sensitivity, and quality expectations all contribute to the challenge. Traditional methods of designing, testing, and implementing mechatronic systems cause designers to wait until late in the design effort, when actual or prototype products and real-time embedded targets become available, to find out whether the system actually meets the performance requirements. Only then, as system integration occurs, can the designer uncover the errors that may have found their way into the product during the early design stages.

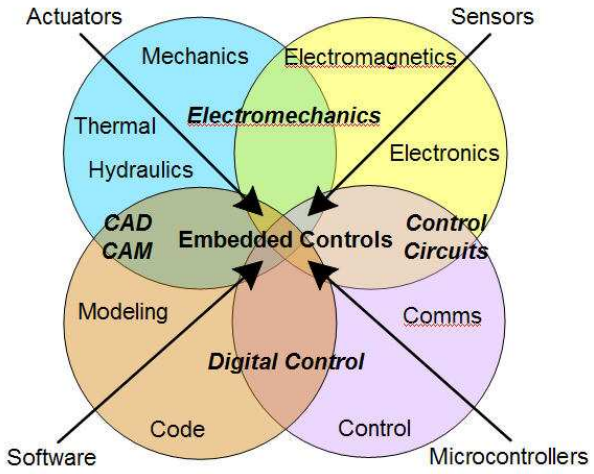

Figure 1. Mechatronics Venn diagram.

The principles of Model-Based Design as a proven technique for creating embedded control systems[2,3] apply equally as well when designing mechatronic systems. Using Model-Based Design, the various design teams can evaluate design alternatives without relying solely on expensive prototypes. Model-Based Design allows engineers to mathematically model the behavior of the physical system, design the software and model its behavior, and then simulate the entire system model to accurately predict and optimize the overall performance.

Once the base design has been established, further optimization can be easily performed by studying the effects of component variances on the overall system performance. A robust design is one that is immune to component variances due to temperature, manufacturing tolerances, and other factors. There are many sources of information on robust design, Six Sigma, and the Taguchi method; the focus of this paper is to show the impact of applying such methods to a mechatronic embedded control system.

MODELING THE PHYSICAL SYSTEM

Model-Based Design is widely used to develop software algorithms for deployment onto an embedded controller. For closed-loop testing of the control algorithm, the first thing that is needed is a representation of the plant. There is no shortage of techniques for modeling physical systems. Commonly used methods include signal flow diagrams[4], bond graphs[**Error! Reference source not found.**], and even manual coding of the system equations in C or Fortran. Since a mechatronic design relies on collaboration between engineering teams, the model must be easily shared and understood by the various stakeholders. While the methods above are perfectly valid for accurately modeling the physics, none of these are particularly well-suited for meeting the collaboration and integration needs of a multidomain mechatronic system design. As a simple illustration, consider the problem of modeling a DC motor with speed and current control.

SIGNAL FLOW APPROACH - Simulink[6] from The MathWorks is widely used to design control algorithms using the signal flow approach. Once implemented in Simulink, Model-Based Design methods are commonly used to verify the controller design and automatically generate the code for deployment onto the microcontroller for rapid prototyping and production. As a result, the signal flow method has also historically been used to model the plant in Simulink to test the controller in simulation and with a real-time hardware-in-the-loop system.

Figure 2 shows a common textbook representation of a DC motor.

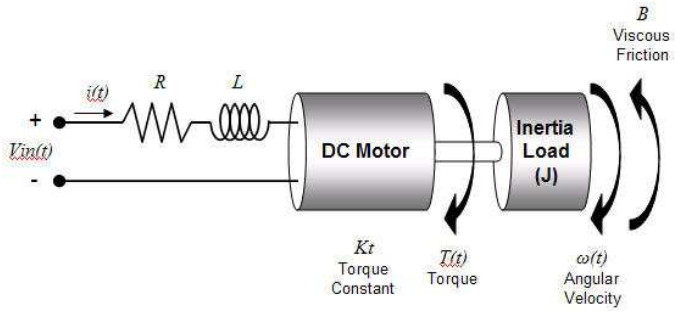

Figure 2. DC motor architecture.

The signal flow modeling approach is a multistep process that requires first deriving the motor equations:

$$T = K \cdot i - B \cdot \omega - J \cdot \frac{d\omega}{dt} \quad (1)$$

$$V = K \cdot \omega + i \cdot R + L \cdot \frac{di}{dt} \quad (2)$$

The next step is to graphically model these equations in a signal flow diagram, but this step often requires reformulating the equations to support this approach:

$$\frac{di}{dt} = \frac{1}{L}\left(V - i \cdot R - K \cdot \frac{d\theta}{dt}\right) \quad (3)$$

$$\frac{d\omega}{dt} = \frac{1}{J}\left(K \cdot i - B \cdot \frac{d\theta}{dt}\right) \quad (4)$$

Finally, a signal flow model of the equations can be created, as shown in Figure 3.

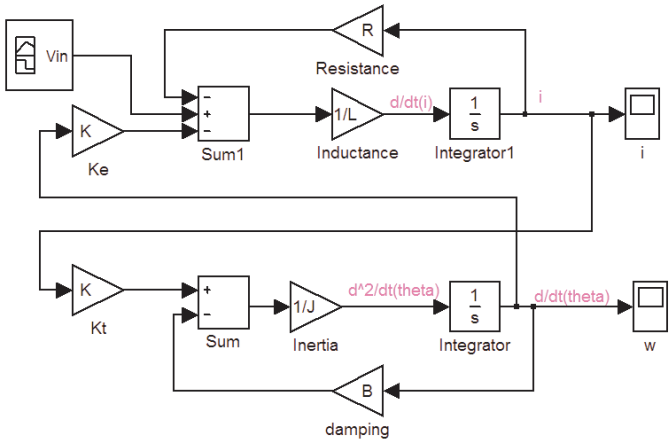

Figure 3. DC motor signal flow model.

Simulating this model yields the expected results (see Figure 4), but the multistep modeling process results in a model that is unrecognizable when compared with the original diagram in Figure 2. Sharing even this simple model with others would require significant explanation and documentation.

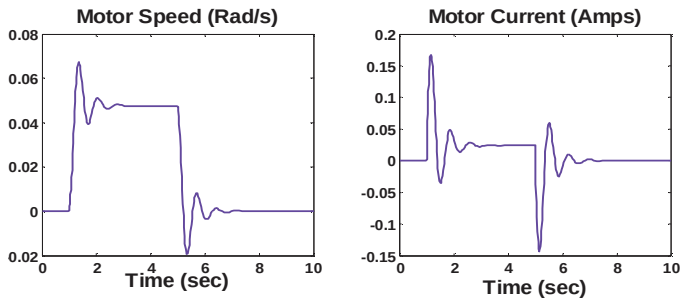

Figure 4. DC motor model simulation results.

NETWORK APPROACH - A more universal method for modeling multidomain physical systems is often referred to as the network modeling approach[7]. Its origins come from the method of network analysis for electrical systems and have been extended to also model systems consisting of mechanical, hydraulic, thermal, and magnetic components. The main advantage of a network model over a signal flow model is the acausal[8] nature of the connection ports.

In a signal flow diagram, the connections are causal. That is, every block is a transfer function with a signal on the input causing the output to behave according to the defined transfer function. The model in Figure 3 illustrates how data flows through the model. Any interaction between blocks must be explicitly modeled by creating feedback loops. As the interactions become more complex and commonplace, as with a mechatronic system, the signal flow method quickly becomes untenable for all but the most expert users. For example, if additional effects such as damping, friction, or hard stop limits are desired, the system equations (1) and (2) would need to be reformulated and the model recreated, resulting in an even more complicated model that is more difficult to interpret.

Figure 5 illustrates the same DC motor model using the network approach modeled using foundation blocks from the Simscape[9] multidomain physical modeling environment within Simulink.

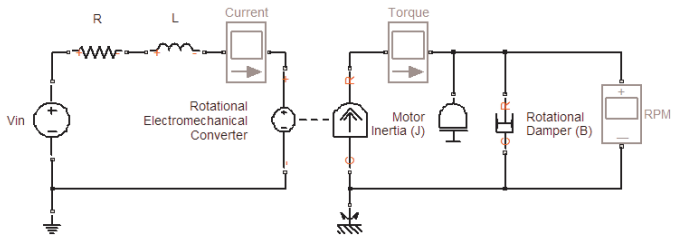

Figure 5. DC motor network model.

As you can see, the model bears a close resemblance to the original diagram. This is an important benefit of the network approach, making it much easier for others to understand and interpret, thus fostering the collaboration needed for designing mechatronic systems. The electrical side of the model solves for current and voltage while the mechanical side solves for torque and angular velocity, resulting in identical simulation results. In the network terminology, these are commonly referred to as "through" and "across" variables. Current and torque scopes are placed in the network to measure the through variables, and the RPM scope is placed to measure the across variable of motor shaft speed. These measured quantities can also be easily fed back to the control algorithm modeled in Simulink (more on this later).

A major advantage of the network approach is the ability to quickly modify the system model without the need to derive the system equations. The individual blocks contain the fundamental component equations defining the relationship between the through and across variables. The system equations are then automatically formulated by interconnecting the components into the desired topology. For example, the rotational damper component contains the equation:

$$T = B \cdot \omega \quad (5)$$

This equation defines the relationship between the through variable (torque) and the across variable (angular velocity) as a linear relationship with the damping coefficient (B) as a constant of proportionality. This method of embedding the first-principle equations into the component models allows additional physical effects to be easily added to the system model without needing to worry about their effect on the overall system equations. For example, let's say we want to add limits to the angle of rotation. Using the network approach, you can simply connect a rotational hard stop to the motor shaft as shown in Figure 6.

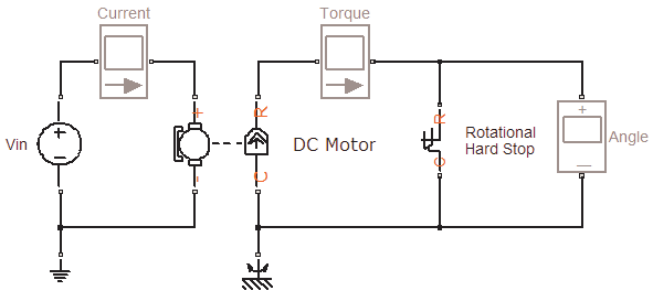

Figure 6. DC motor model with hard stop as load.

Here we also used hierarchy to group the previous motor model into a subsystem and added the rotational hard stop as an external load. An angular position scope is used to measure the angle of the motor shaft. As you can see, the network approach makes it easy to quickly add effects and immediately see them in simulation.

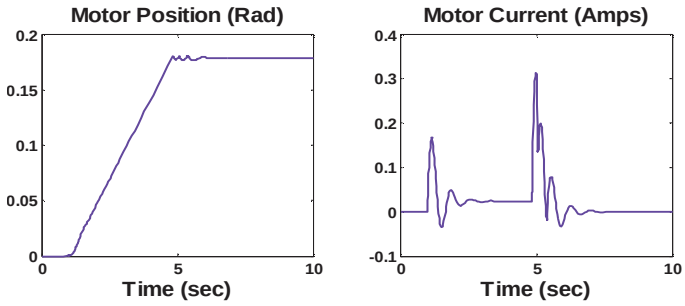

Figure 7. Motor position and current with hard stop.

Figure 7 clearly shows the results of introducing angular travel limits. The motor shaft reaches the hard stop at about 5 sec, resulting in an increase in the motor current as it works harder to overcome the obstacle. The motor

angular position, like angular velocity, is an across variable that can be fed back to the controller if desired.

MODELING LANGUAGE - The enabling technology for the network approach is a modeling language for formulating the component's characteristics equations relating the through and across variables in the various domains. The Simscape language, based on MATLAB[10], provides the necessary constructs for modeling the multidomain aspects of mechatronic systems. In the DC motor example, the motor equations can be directly modeled using the Simscape language, as shown in Figure 8.

```
component dc_pm
  nodes
    p = electrical; % p:left
    n = electrical; % n:left
    r = rotational; % r:right
    c = rotational; % c:right
  end
  parameters
    Kt = {10 'N*m/A'};      % Torque constant
    Ke = {10 'V/(rad/s)'}; % Back EMF Constant
    Rwind = {1 'Ohm'};      % Winding Res
    Lwind = {1e-3 'H'};     % Winding Ind
    J = {1 'kg*m^2'};       % Motor Inertia
    B = {1 'N*m/(rad/s)'}; % Motor Damping
  end
  variables
    theta = {0,'rad'}; % Angular Displacement
    tq = {0,'N*m'};    % Torque thru variable
    w = {0,'rad/s'};   % AngVel across var.
    i = {0,'A'};       % Current thru var.
    v = {0,'V'};       % Voltage across var.
  end
  function setup
    through(tq,r.t,c.t); % thru variable tq
    across(w,r.w,c.w);   % across variable w
    through(i,p.i,n.i);  % through variable i
    across(v,p.v,n.v );  % across variable v
  end
  equation
      w == theta.der;
      v == Ke*w+i*Rwind+Lwind*i.der; % Motor
      tq == -Kt*i+B*w+J*w.der;       % Eq'ns
  end
end
```

Figure 8. DC motor model using Simscape language.

This modeling method creates a new foundation component that can then be easily integrated into a larger system model by inserting it between the electrical controls and mechanical loads.

For example, consider the hydraulic actuation system in Figure 9. The DC motor is used to energize the hydraulic pump providing pressure to the system, which will actuate mechanical motion as the valve directs the fluid to the double-acting cylinder. Table 1 summarizes the multidomain through and across variables present in this system.

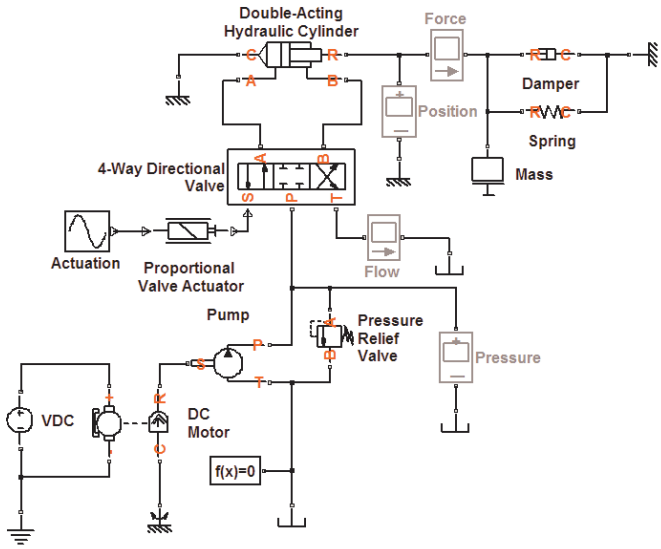

Figure 9. Hydraulic actuation system.

Domain	Through	Across
Electrical	Current	Voltage
Rotational	Torque	Angular velocity or position
Translational	Force	Velocity or position
Hydraulic	Flow rate	Pressure

Table 1. Domain variables.

With the network approach, a system model can be quickly constructed by interconnecting the individual component models. The resulting model representation is intuitive and easily interpreted due to the physical connection ports. The overall system equations are automatically formulated from the individual component equations based on the system topology. Figure 10 shows the overall system response (cylinder rod position).

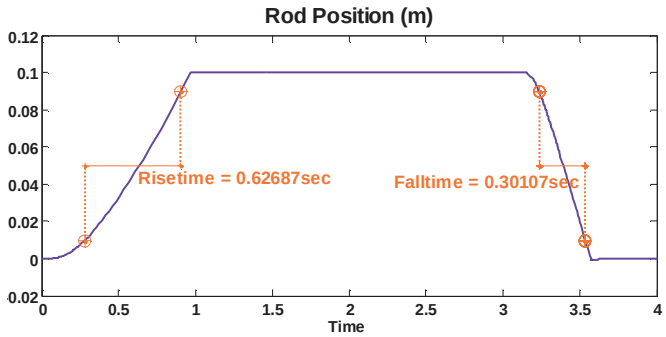

Figure 10. Cylinder rod position.

Additional instrumentation can be added to examine the internal through and across variables. Figure 11 shows the pressure on the P port of the hydraulic valve along with the flow rate though the valve and cylinder.

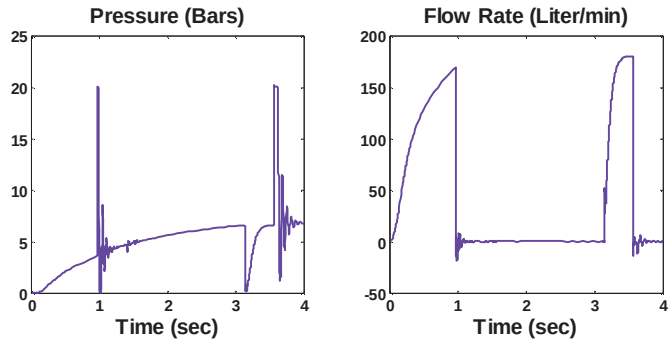

Figure 11. System pressure and flow rate.

The results show that the flow rate is positive during the movement of the cylinder. The spikes in pressure occur when flow rate suddenly goes to zero due to valve closure. The pressure relief valve is set to 20 bar to mitigate this situation. The overall pressure profile is a gradual increase except for a sharp drop at about 3.2 sec followed by a quick recovery when the valve opens the second time and reverses the cylinder direction.

TUNING PARAMETERS - One of the challenges of any physical model is validating that the simulation results are accurate and represent reality. With the network approach to modeling, the model parameters are the degrees of freedom for adjusting the model performance. In some cases, these parameters can be populated directly from the datasheet of a component manufacturer. For example, the stall torque and no-load speed curves on a motor datasheet could be used to parameterize the motor model. Many times, however, good data is not available and the model parameters must be manually adjusted. This is typically a tedious trial-and-error (adjust, simulate, repeat) process until a reasonable result is obtained.

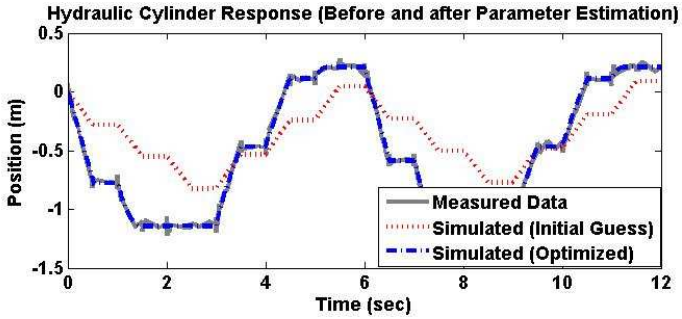

Figure 12. Hydraulic cylinder response (measured and simulated).

Optimization tools such as Simulink Parameter Estimation[11] can be used to automate this process by automatically tuning the model parameters by comparing the simulation results with measured lab data until a satisfactory parameter set is obtained. Figure 12 shows the results from using Simulink Parameter Estimation to automatically tune parameters in the hydraulic portion of the model in Figure 9.

PARAMETER VARIATIONS

The nominal simulation results with optimized parameters are useful for testing the controller and verifying the overall system performance. However, an optimized design does not necessarily ensure that the design is robust. A robust design[12] is one that is immune to component variances due to tolerances, temperature, aging, and other factors. Once the nominal performance has been validated, it is important to consider these variances and account for their effect on system performance when modeling a physical system. The MATLAB programming language can be used to automatically measure various aspects of the simulation result from the Simulink and Simscape physical model. For our hydraulic actuation system in Figure 9, the overall performance metric might be the time it takes for the cylinder to open and close. Measuring the risetime and falltime of the rod position as shown in Figure 10 quantifies this performance and provides the basis for further parametric analysis.

A common “brute force” technique for analyzing the effect of parametric variances is Monte Carlo[13] simulation. This method randomly varies component parameter values within a prespecified tolerance range and according to a probability distribution[14]. Figure 13 shows some common distributions.

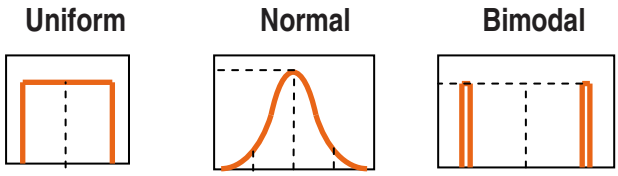

Figure 13. Common probability distributions.

With today’s computing power and tools such as Parallel Computing Toolbox[15] to distribute the simulation runs across multiple computers, a large number of runs can be simulated and the results automatically post-processed with little or no human intervention.

For this example, SystemTest[16] was used to assign tolerances to a set of physical parameters, perform 1000 simulation runs, and collect the performance measurement data. For simplicity a normal (Gaussian) distribution with 10% tolerance was assigned to the physical parameters (see Appendix A).

The simulation results in Figure 14 show the effect of the parametric variances on the rising edge of the cylinder rod position. During the Monte Carlo simulations, the MATLAB risetime measurement was applied after each run and the results were collected in a histogram plot (see Figure 15).

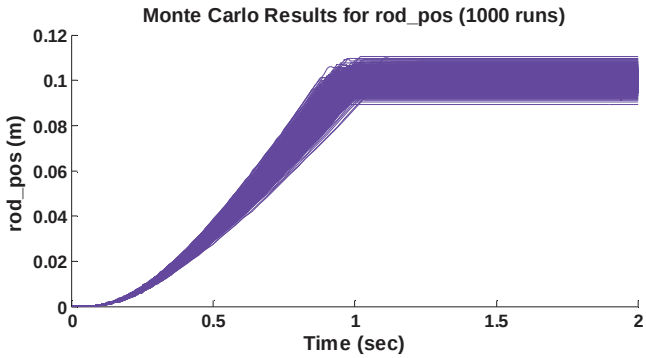

Figure 14. Monte Carlo results (1000 runs).

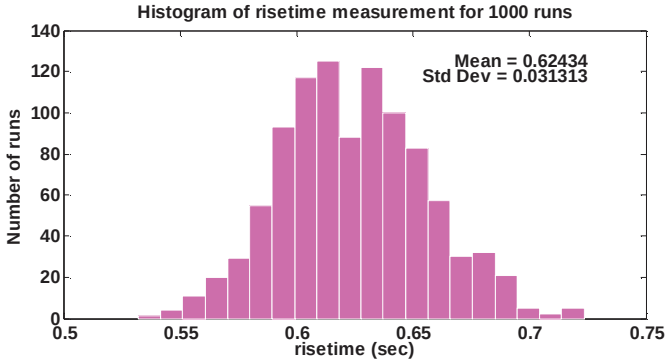

Figure 15. Histogram of risetime measurement.

The histogram provides us with a statistical view of the system performance that takes into account parameter variances. From this, we arrive at a quantifiable assessment of the robustness of our design and can use the results to directly determine if the observed performance variation falls with acceptable limits based on requirements. Requirements management tools such as Simulink Verification and Validation[17] can then use this information to automatically generate reports to communicate the results to other members of the design team.

If the performance variation is not acceptable, the next logical step is to determine the major contributors to this variation. Again, the MATLAB plotting and visualization capability can be used to further analyze the data. The scatterplot[18], shown in Figure 16, is one useful representation. Here, we have plotted the measured risetime for each simulation run against the value of one of the parameters (area_b of the directional hydraulic valve). The scatterplot data points reveal a visual trend of increasing risetime for increasing values of the parameter value. Additional measurements quantify this data into sensitivity (slope of best-fit line) and correlation coefficient (measure of deviation from the best-fit line). This information can be used to assess the allowable tolerances for a given parameter, but for a complex mechatronics design with many physical parameter variations and multiple performance measurements, a more efficient mechanism for visualizing the data is needed. Again we resort to MATLAB to manage the information using a common representation known as a Pareto chart[19], shown in Figure 17.

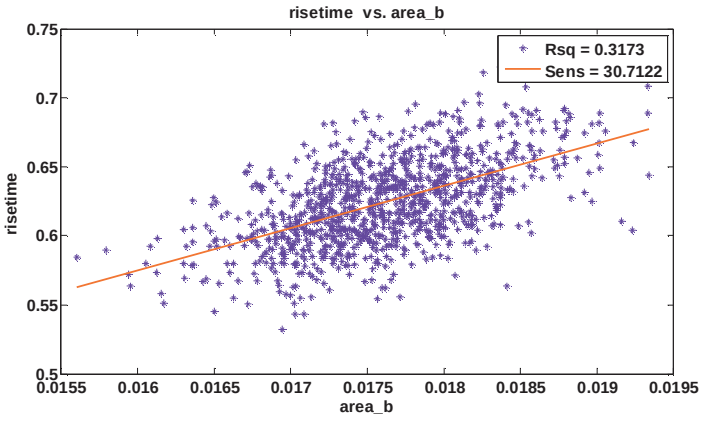

Figure 16. Correlation scatterplot.

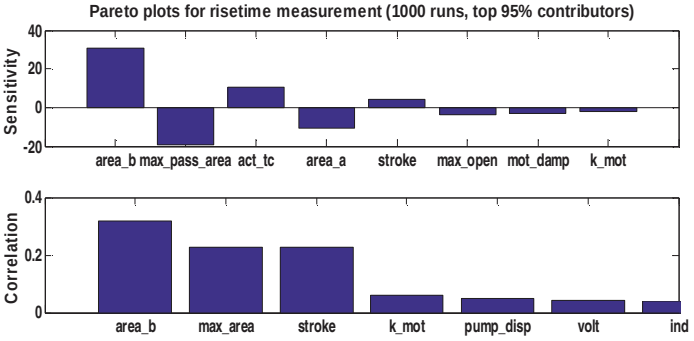

Figure 17. Pareto plot of risetime for 1000 runs.

This compact representation allows us to quickly see the contributors to the variation in terms of both sensitivity and correlation. For the top contributors, we will want to carefully control the variation by tightening the tolerances, and for the parameters that have little or no contribution, we can relax the tolerances where it saves cost during production. Once the Monte Carlo data has been generated, MATLAB can then be used to automate the data mining process by performing similar analysis on multiple performance measurements. This information can then be used by the design engineers to make engineering decisions and tradeoff assessments to optimize the performance and cost of the mechatronic design prior to building any hardware.

ROBUST CONTROL

Understanding the effects of parameter variation on the physical system provides valuable insight from an open-loop perspective. During controller design, it is important to understand the effects of tolerances on the closed-loop performance. The measure of controller robustness is how well it controls the desired output when parameter variances are present. In Figure 18, the physical plant model is encapsulated into a Simulink subsystem complete with sensors and actuator so that it can be

connected to a PID controller[20]. A command signal is introduced for the desired position, and the overall performance is measured by comparing the system output with the command.

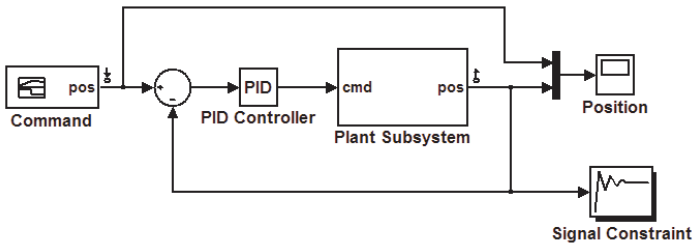

Figure 18. Closed-loop system model with controller and physical plant subsystem.

The system response can then be optimized by tuning the controller gains. This can be done manually using trial and error, or an optimization tool such as Simulink Response Optimization[21] can be used to automatically tune the gains to meet the desired performance. This is done by placing a signal constraint block that specifies the acceptable region of operation on the desired output. Figure 19 shows the result of the optimization.

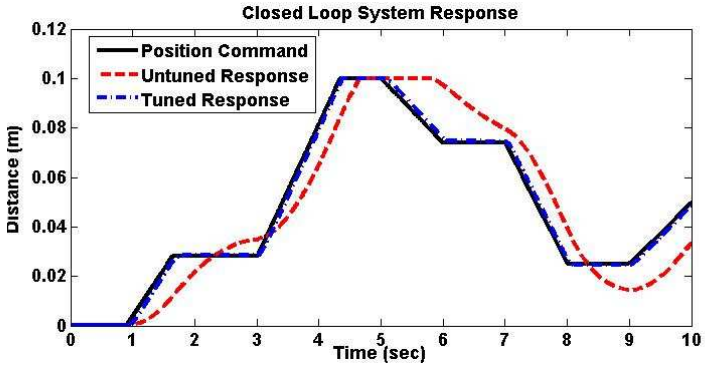

Figure 19. Closed-loop system response.

The initial choice of PID gains resulted in a poor response when connected to the plant model, but closed-loop response after tuning the gains is much more acceptable.

The Pareto chart from the closed-loop Monte Carlo results revealed some tolerance issues with the cylinder. After the suspect tolerances were tightened, the simulation results of the closed-loop system in Figure 20 show that the controller is indeed robust enough to handle the specified variation in parameters. As the system design continues and more complex controls are added, tools such as Robust Control Toolbox[22] can be utilized to analyze and design multi-input/multi-output (MIMO) control systems by providing methods for mode order reduction and consideration of stability margins and worst-case performance. These advanced techniques can be applied to any Simulink model, including Simscape physical models, but are beyond the scope of this paper.

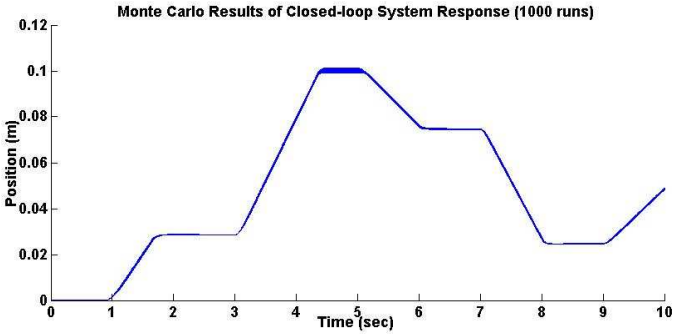

Figure 20. Closed-loop response (1000 runs).

CONCLUSION

To design a robust mechatronic embedded control system, you must account for the effect of parameter variances on system performance. Model-Based Design has been shown to be a proven method for control algorithm design, verification, and deployment. To test the controller using Model-Based Design, you must create a model of the physical system (or plant). The network approach offers many advantages over traditional signal flow or block diagram methods that are typically employed by controls engineers, especially for multidomain mechatronic systems.

Instead of relying on domain expertise to develop and model the system equations, the network approach with the aid of a modeling language applies first-principle equation modeling at the component level. The individual components are then combined to model the larger system. The result is a very readable system model that can be easily shared with others on the team and quickly modified to account for additional effects as needed. In constructing the system model in this way, the engineer can then apply optimization techniques to find a set of nominal parameters to meet the performance requirements.

The process of robust design includes methods for analyzing the effects of parameter variations on the overall system performance. Monte Carlo simulation is one method that can be applied to both open- and closed-loop systems and requires the ability to analyze the large amounts of data typically generated. Statistical techniques such as histograms, scatterplots, and Pareto charts can be used to uncover relationships between the performance measurements and the model parameters. This information can then be used to make design decisions such as cost/performance tradeoffs. Expanding use of Model-Based Design in this way to consider both controller and plant models can help the entire mechatronics design team build a more robust design.

ACKNOWLEDGMENTS

I would like to thank my colleagues at The MathWorks Steve Miller, Jeff Wendlandt, Val Tchkalov, Nathan Brewton, Rick Hyde, Craig Borghesani, and Chris Portal (among many others) for supporting me with Simulink models and MATLAB functions while listening to me preach about the importance of robust design. I would also like to thank some of my past colleagues Darrell Teegarden, Mike Donnelly, David Bedrosian, and Subba Sommanchi from my days at Analogy Inc. for providing the inspiration with their foundational work in the area of robust design.

REFERENCES

1. en.wikipedia.org/wiki/Mechatronics
2. Paul F. Smith, Sameer M. Prabhu, Jonathan H. Friedman, “Best Practices for Establishing a Model-Based Design Culture,” SAE Paper 2007-01-0777.
3. Jeff Thate, Larry Kendrick, Siva Nadarajah, “Caterpillar Automatic Code Generation,” SAE Paper 2004-01-0894.
4. *Feedback Control of Dynamic Systems,* Gene F. Franklin, J. David Powell, Abbas Emami-Naeini, Prentice Hall, ISBN 0-13-032393.
5. en.wikipedia.org/wiki/Bond_graph
6. www.mathworks.com/products/simulink
7. *Mechatronics: An Integrated Approach,* Clarence W. De Silva, CRC Press, 2005, ISBN 0849312744.
8. en.wikipedia.org/wiki/Acausal
9. www.mathworks.com/products/simscape
10. www.mathworks.com/products/matlab
11. www.mathworks.com/products/simparameter
12. en.wikipedia.org/wiki/Robust
13. en.wikipedia.org/wiki/Monte_Carlo_Simulation
14. en.wikipedia.org/wiki/Probability_distribution
15. www.mathworks.com/products/parallel-computing
16. www.mathworks.com/products/systemtest
17. www.mathworks.com/products/simverification
18. en.wikipedia.org/wiki/Scatterplot
19. en.wikipedia.org/wiki/Pareto_chart
20. en.wikipedia.org/wiki/PID_control
21. www.mathworks.com/products/simresponse
22. www.mathworks.com/products/robust

CONTACT

Tom Egel
Principal Application Engineer
The MathWorks
tom.egel@mathworks.com

APPENDIX A

Physical model parameters (nominal values and standard deviation) were used for Monte Carlo analysis. A normal (Gaussian) distribution was used for all the parameters.

MOTOR
res=0.15, sd=0.005
ind=200e-3, sd=0.0067
k_mot=0.1, sd=0.0033
mot_inertia=0.1, sd=0.0033
mot_damp=0.01, sd=0.00033
volt=100, sd=3.333

PUMP
pump_disp=0.5, sd=0.0167

DIRECTIONAL VALVE
max_area=200, sd=6.6667
max_open=0.015, sd=0.0005

RELIEF VALVE
max_pass_area=1e-3, sd=3.33e-5
set_pres=20, sd=0.6667

CYLINDER
area_a=0.0106, sd=3.53e-4
area_b=0.0176, sd=5.87e-4
stroke=0.1, sd=0.0033

ACTUATOR
act_gain=250, sd=8.333
act_tc=0.002, sd=6.667e-5
act_sat=0.3, sd=0.01

LOAD
stiff=1000, sd=33.33
damp=100, sd=3.33
mass=1, sd=0.0333