ABSTRACT

Mechatronic systems such as the anti-lock braking system, the powertrain control system, and even domestic items such as a microwave oven are designed such that there is a seamless interaction between the electronic components and the mechanical components that make up these systems. Use of software tools in simulating the behavior of such systems is an important step in their product development cycle. Since the success of a mechatronic product depends heavily on the synergistic interaction of all its components, it is imperative that the software tool used in its design be capable of addressing its multi-domain needs.

There are several commercial software tools that are currently available for modeling and simulation of mechatronic systems. Adequacy of all these tools is not the same. Some are adept as a mechanical simulator but do a poorer job of electronic simulation. Others are just the opposite. In this paper a benchmarking study will be reported in which a couple of mechatronic systems will be modeled using several of these well known modeling tools. The capability of these software tools for modeling mechatronic systems will be analyzed through this effort.

INTRODUCTION

One of the important steps during the early stages of a design process is conceptual design of the system. In cases of complex systems, this is not only critical for the success of the product but is also the most difficult stage of the whole product development process. System simulation tools are used a lot at this stage to simulate system behavior. “What if” type questions are asked and answered by trying out different alternative solutions.

In many modern day products the traditional barriers between disciplines have broken down. For example, in automobiles, efficient computers and electronic control that work seamlessly with mechanical components have replaced several mechanical subsystems. The response time required and the accuracy and efficiency required in modern automobiles are not always achievable by purely mechanical subsystems. With more and more seamless integration of multi-disciplinary subsystems, modeling and simulation at the conceptual stage of the design has become even more important. The questions that need to be answered at this stage are not related to any specific discipline alone but are quite interdisciplinary.

Mechatronics is a term used to refer to an interdisciplinary area consisting of mechanical, electrical, electronics and computer technology. The term was originally coined by the Japanese in the mid-sixties and has been used since then to indicate simultaneous use of multidisciplinary technology for designing a better product. Mechatronics is not necessarily a new discipline but a synergistic combination of many disciplines in the context of design. To ensure true synergy in the mechatronics approach to design it is important that multidisciplinary solutions are considered at the early stages of the process. This is easier said than done because our traditional training is in more compartmentalized disciplines such as Mechanical, Electrical or Computer engineering.
In each of the traditional disciplines of engineering simulation techniques are well developed. But they have evolved along different paths determined by the unique needs of the discipline. For example, in the mechanical world, energy is transmitted through physical interaction between objects. As a result, Mechanical simulation tools rely a lot on visualization of physical interaction between objects. In the electrical world, the simulation techniques need to capture flow of charge or signals, none of which can be physically seen but their effects are quite perceptible. As a result, for the simulation of electrical components it is more important to look at signals and system behavior in the form of data plots. Therefore, tool such as ADAMS [1] simulates dynamic behavior of mechanical systems, and visualization of moving objects is an integral part of this tool. Conversely, a circuit simulator such as pSPICE [2] relies on graphical output of data such as current flow or voltage drop.

In modeling Mechatronic systems one is thus faced with the difficulty of choosing a suitable tool for use. One can always develop customized tools and many researchers have done that. However, there are some commercial tools that can be suitably adapted for use. In all of these cases there are some challenges in using the modeling tool chosen because of the way the tool has been developed. In the next few sections we have considered three such modeling tools and analyzed two mechatronic examples using these three tools of choice. Subsequently, we have summarized our findings in about the important issues associated with using each of these tools.

MODELING TOOLS EVALUATED

MATLAB/SIMULINK [3]

Matlab is an interpreted language for numerical computation. It allows one to perform numerical calculations, and visualize the results without the need for complicated and time consuming programming. Matlab allows its users to accurately solve problems, produce graphics easily and produce code efficiently. Widely used by engineers and scientists, it has over 500 mathematical, statistical and engineering functions. It's an integrated technical computing environment that combines numeric computation, advanced graphics and visualization and a high-level programming language. MATLAB includes functions for data analysis and visualization; numeric and symbolic computation; engineering and scientific graphics; modeling, simulation and prototyping; programming, application development; and GUI (graphical user interface) design.

Simulink is an interactive tool for modeling, simulating, and analyzing dynamic systems. It enables the user to build graphical block diagrams, evaluate system performance, and refine designs. Simulink integrates with MATLAB. Simulink is the tool of choice for control system design, DSP design, communications system design, and other simulation applications.

WORKING MODEL 2D [4]

Working Model 2D, is a popular 2D computer aided engineering tool for dynamic simulation. This product also has a 3D version that can be integrated with a finite element solver. The basic features of working model includes physics-based 2D kinematics and dynamic motion simulation and analysis, accurate solutions to complex engineering motion simulation problems, ability to apply field forces such as gravity, electromagnetic forces, etc, a variety of basic shapes to build body types along with property assignment, and automatic collision detection and response.

20SIM AND BG METHOD [5]

Bondgraph [6] is an explicit graphical tool for capturing the common energy structure of systems. It increases one's insight into systems behavior and gives concise description of complex systems. The notations of causality provides a tool not only for formulation of system equations, but also for intuition based discussion of system behavior, viz. controllability, observability, fault diagnosis, etc.

In 1959, Prof. H.M.Paynter gave the revolutionary idea of portraying systems in terms of power bonds, connecting the elements of the physical system to the so called junction structures which were manifestations of the constraints. This power exchange representation of a system is called BondGraph (some refer it as BG). In this approach, a physical system can be represented by symbols
and lines, identifying the power flow paths. The lumped parameter elements of resistance, capacitance and inductance are interconnected in an energy conserving way by bonds and junctions resulting in a network structure. This modeling technique has been extended to power hydraulics, mechatronics, thermodynamic systems, and recently to electronics and non-energetic systems like economics and queuing theory. The beauty of this technique is that it is discipline independent and mechatronic systems can be as easily modeled as purely mechanical or purely electrical systems. From the pictorial representation of the bond graph, the derivation of system equations is so systematic that it can be algorithmized. The whole procedure of modeling and simulation of the system may be performed by some of the existing commercial software tools e.g., 20-SIM [5], Camp-G [7], SYMBOLS 2000 [8], etc.

20-sim can simulate the behavior of dynamic systems, such as electric, mechanical and hydraulic systems or any combination of these systems. 20-sim has been developed at the Control Laboratory of the University of Twente. 20-sim fully supports graphical modeling, which allows designing and analyzing dynamic systems in an intuitive and user friendly way, without compromising power. 20-Sim also has the capability of defining engineering objects with all the properties built into the definition and with the associated bond graph model built into the object definition.

All three of the above described software tools are capable of modeling mechatronic systems. In the next section the two examples chosen for this study are described. Subsequently, models for the two examples are developed on all three software tools and solved. Some simulation results are included for comparison. Finally, pros and cons of each tool are discussed in the last section of the paper.

EXAMPLES

Example 1: Permanent magnet DC Motor

A permanent magnet DC motor is a very commonly used actuator in mechatronic applications. This is used quite often where an electrical input (voltage) is converted into a rotational motion output. For permanent magnet DC motors the torque generated is linearly related to speed of rotation. This makes it an attractive device for use in many applications. PM DC motors can also be quite compact in size and forms the core of other actuators such as servo and stepper motors.

The permanent magnet DC motor can be usually modeled as a voltage source that drives a current through an inductor (of the motor armature) and a resistance (armature winding resistance). The output of the motor is a torque that drives a mechanical inertia load with some damping and elasticity.

The equations that govern motor motions are:

Electrical Equation is given by

\[
\frac{di}{dt} = \frac{1}{L} \{V_a - E_a - i \cdot R\}
\]

\[E_a = K_{e} \omega\]

Where \(L\), \(R\), and \(i\) are armature inductance, resistance and current, respectively. \(E_a\) is back Emf that is proportional to the speed of rotation of the armature. The mechanical equation is

\[
\frac{d\omega}{dt} = \frac{1}{J} \{T - T_L - B \cdot \omega\}
\]

\[T = K_t \cdot i\]

where \(J\) is the rotor inertia, \(T_L\) is the load torque and \(B\) is the damping from bearings. Torque generated is directly proportional to the armature current.

In our example following parameters were used to model the motor.

Motor Resistance \((R_1 = R = 1 \text{ } \Omega)\)

Motor Inductance \((L_1 = L = 0.05 \text{ } \text{H})\)

Motor Back-EMF Speed Constant \((K_t = 3 \text{ } \text{V-s})\)

Motor Force/Torque Constant \((K_t = 3 \text{ N-m/A})\)

Rotational damping coefficient \((B = 0.09 \text{ N-m-s/} \text{rad})\)

Motor Input Voltage \((V = 42.0 \text{ V})\)

In our example the rotor inertia is varied at four levels \(2.4/1.2/0.6/0.06 \text{ (kgm}^2\))\). The output plots shown in this paper indicate the system response for all these four levels of inertia. As the inertia decreases the velocity response changes from overdamped to underdamped (oscillatory). In all
cases the final velocity reaches the same steady state value.

Figure 1 shows the model and the output as developed in Working Model. Working model has a predefined object representation for motors with options for constant speed, constant torque, and DC motor. For the DC motor option the above listed properties are included in the model. The inertia of the mass attached to the output of the motor is altered to obtain the set of output responses.

Figure 2 shows the Bondgraph model for the DC motor and the output inertia as drawn in 20-SIM. It uses standard terminology such as Se for source of effort, I for inertia or inductance, and R for resistance (damping or electrical). GY is used to represent Gyrators, objects that transform effort input into flow output and flow inputs to effort output (such as voltage to speed and current to torque).

Figure 3 shows the output response of the motor model from 20-SIM. It is essentially the same as the output graphs obtained from Working Model.

Figure 4 shows the Simulink model for the permanent magnet DC motor. The differential equation is modeled in Simulink with specific blocks that handle mathematical operations such as addition, multiplication, integration, etc. The output of this model is shown in figure 5. Like in the other software tools the output shows the system response for different load inertia.

Results from the three different software tools indicate that it was possible to model this particular device using all the tools. However, in case of Simulink the differential equation of the model had to be known completely. Although Working Model is primarily a mechanical simulator it offers some flexibility in modeling motors because motors are used in this tool as actuators.
To model the motor in Working Model one did not need to know the specific differential equation representation. While modeling through the Bondgraph approach using 20-SIM no knowledge of the governing differential equation was necessary. The system had to be drawn as a bond graph whose components are basic mechanical and electrical elements. This is a clear advantage of the Bondgraph based approach over others. In practical situations, one may not always know what the governing differential equation is but will know what basic components make up a system. A bondgraph model will generate the differential equation automatically and then solve it for the system response.

**Example 2: Solenoid Actuator**

There is a growing need for Solenoid Actuators with the increase in the number of control applications. Electromechanical Solenoids are often used as the main actuator for all types of systems ranging from electrically-controlled transmissions to active and semi-active suspensions; they are often good contributors to the dynamic response of a complete system. This section describes the modeling of solenoid using the same simulation and modeling tools MATLAB, Working Model and 20-Sim.

**Derivation of the Electrostatic Field Force**

The model can be divided into electrical, electromagnetic, and magneto-mechanical components. The electrical model accounts for the change of current in the coil. If \( V \) is the voltage input from the driver, \( R_E \) is the resistance in the coil, \( i \) the current and \( N \) the number of turns of the coil in the housing core, then the rate of change of flux \( \dot{\Phi} \) is given by,

\[
\frac{d\Phi}{dt} = \frac{(V_{in} - iR_E)}{N}
\]

and the Magnetomotive force is given by \( M = N i \). The magnetomotive force is divided into two components, a portion for the air gap and another one for the iron core. The portion for the air gap can be written as \( M_{air} = \Phi / C_{air} \) where \( C_{air} \) is a constant that is related to the airgap distance, the area of the gap, and the permeability of free space. The portion that goes through the iron core is in reality a non-linear function of magnetic flux. This is usually obtained from the B-H diagram of the core material and can change based on whether the current is increasing or decreasing because of hysteresis in the B-H diagram. We have made a simplifying assumption for our work and assumed that like \( M_{air}, M_{core} \) is also represented as \( M_{core} = \Phi / C_{st} \), where \( C_{st} \) is a constant as well. Using these assumptions the flux rate equation can be integrated in the following fashion:

\[
\Phi(t) = \int_{0}^{t} \frac{d\Phi}{V_{in} - S\Phi} = \int_{0}^{t} \frac{d\Phi}{V_{in} - S\Phi} = \frac{1}{N} \left[ \ln \left( \frac{V_{in} - S\Phi}{V_{in}} \right) \right] = -\frac{t}{N}
\]

\[
1 - S\Phi / V_{in} = e^{-st/N}
\]
\[ \Phi = \frac{V_{in}}{S} \left[ 1 - e^{-stN} \right], \]

where \( S = \frac{R_E}{N[1/C_{st} + 1/C_{air}]} \)

The mechanical force due to the magnetic flux is given as \( F = \Phi^2/2 \) (area of the gap \( \times \) permeability). Thus the force can essentially be obtained by squaring the magnetic flux expression.

To model the solenoid on the three software tools following constants are used as input parameters:
- Spring Stiffness \( 1/C_2 = K = 40 \text{ N/m} \)
- Damping Coefficient \( R_2 = 0.001 \text{ N-s/m} \)
- Input Voltage \( V_{in} = 12 \text{ V} \)
- Mass of the Armature \( m = 1/4 \text{ kg} \)
- Force Field Applied \( F_x = - ((24)*(1-exp (-t/40)))^2 \text{ N} \)

\[ R_1 = R_E = 5 \Omega \]
\[ GY = N = 10 \]
\[ C_1 = C_{st} = 2 \]
\[ C_{air} = 2 \]
\[ 1/C_3 = K_{stop} = 1600 \text{ N/m} \] (when displacement is \(-1\) or less), \( 0.001 \text{ (until the displacement reaches the maximum value)} \).

Figure 7 shows the solenoid model and the simulation result from Working Model. The mechanical components (the magnet mass, the armature mass, spring and damper) could be easily created but no provision is available for creating the electrical components. This software offers a provision for including field forces. The resulting mechanical force from the magnet on the solenoid as stated in the formula above is therefore included in this model as a magnetic field force. When the magnet is turned on the armature mass is drawn towards the magnet until it physically comes in contact with the magnet. At that point no further movement occurs. The output plot depicts the distance moved by the actuator. After a total movement of 1 unit (m) the armature comes in contact with the magnet and stops.

Figure 8 shows the bondgraph model for the solenoid [8]. In this case, once again, the model is built with certain basic standard elements such as resistance, source of effort, gyrator, and energy storage devices (capacitors/springs). The physical stoppage of the armature is modeled by adding an extra spring in the model (C3 in the figure) that has very negligible spring constant at the start but the spring becomes extremely stiff when the displacement of the armature reaches the value of 1 unit. The output obtained (figure 9) is exactly the same as the output from Working Model.

Figures 10 and 11 shows the Simulink model for the solenoid. Figure 10 shows the main model and figure 11 is a submodel used in figure 10. The submodel is used to create the logic for stopping the motion of the armature when it comes in physical contact with the electromagnet. The Simulink model calculates the displacement of the armature by integrating the governing differential equation. The submodel logic forces the model to stop the integration process when the displacement is reached a pre-defined value (1 unit in this case) and forces the displacement to remain unchanged at that point.

Figure 7: Solenoid model and results from Working Model

Figure 8: Bondgraph based model for solenoid
DISCUSSION AND CONCLUSIONS

We attempted to model two mechatronic systems using three different software tools. It is clear from the results that we were able to successfully model both systems. During this effort some of the advantages and disadvantages of using the different software tools was quite clear. In this section we discuss our findings.

Working model was developed as a mechanical simulator. It is an excellent tool for modeling mechanical systems. Physical objects can be easily defined by drawing the objects; specification inter-object relationships such as friction, gear ratio, applied force, etc can be done easily as well. The software develops and solves all the equations of multi-body dynamics. Collision between bodies is easily determined. Working Model is also capable of handling field forces such as gravity, magnetic forces, etc. User-defined inter-object relationships can be incorporated through the equation option. Even though it is such a useful tool for modeling mechanical systems, Working Model was not designed to handle electrical components. There is no way to model electrical components directly in this tool. For the two examples discussed here, the electrical aspects were handled in two slightly different ways. For the permanent magnet DC motor, Working Model provides a choice of different motors that could be used, such as constant torque, constant speed and DC motor. Once a DC motor is chosen the software requires parameters.
that describe a DC motor such as armature inductance, resistance, applied voltage, and motor torque and speed constants. The software uses these parameters to generate the differential equation for the motor. While we were successful in modeling the motor we will not be able to model other electrical circuit components explicitly since there are no such options available for standard electrical components. For the second example, the electrical components were not explicitly modeled but the magnetic force induced by the electromagnetic had to be derived and the derived equation had to be used as a field force equation in the model. This required through understanding of the problem details by the user. About Working Model one thing is clear; even though both the examples were successfully modeled using this software tool, there is no guarantee that we will be able to model another mechatronic system using this tool. Hence, although Working Model is an excellent software tool for modeling mechanical systems, it is not very suitable for modeling mechatronic systems.

Simulink was the second software tool used in this study. Using Simulink we were able to successfully model both the problems. Simulink uses block diagram based visual representation of the differential equations governing the system. Both mechanical and electrical systems (thus mechatronic systems as well) can be modeled with same level of ease/difficulty provided the governing differential equations are known completely. In both the cases reported here the Simulink model solves the governing differential equations to provide the system response. Simulink is thus a very powerful tool for mechatronic system modeling provided the governing equations are well understood.

20-Sim, a software tool that uses Bondgraph based system modeling approach, is the third software used for our study. Bondgraph technique is based on energy flow representation and works with some basic elements that are domain independent. These basic elements include energy storage devices such as capacitor (or spring) and inductor (or mass), energy dissipative devices such as resistance (or damper), energy sources such as source of effort (e.g. battery) or source of flow (e.g. current source) and a few other basic components. Since these basic elements are domain independent, this approach suits well for mechatronic systems, which are multi-domain in nature. An additional advantage of the Bondgraph based approach is that once the Bondgraph representation of the system is developed, the differential equations can be derived from it in a very algorithmic way. Within 20-Sim the user provides only the Bondgraph representation of the problem as input. 20-Sim automatically generates the differential equations and solves them. Thus, using a tool such as 20-Sim does not require the user to derive the governing differential equation prior to simulation.

Clearly, 20-Sim, the Bondgraph based modeling tool has some distinct advantages over some of the other tools available for modeling mechatronic systems. However, both Working Model and Simulink are versatile in their own rights and knowing how to use them offers the user opportunities to choose the right tool for appropriate applications.

REFERENCES


CONTACT

Shuvra Das, Professor, Mechanical Engineering Department, University of Detroit Mercy, 4001 W. McNichols Road, Detroit, MI 48219-0900, 313-993-3380, dass@udmercy.edu.

Arif Baig, Goldspark Holdings Limited, 3025 Boardwalk, Suite 150, Ann Arbor, MI 48108, 734-995-3910. arif_baig@yahoo.com.

Chandramohan D. Naidu, Goldspark Holdings Limited, 3025 Boardwalk, Suite 150, Ann Arbor, MI 48108, 734-995-3910. mohan@goldspark.com.