Kinematics Support for Design and Simulation of Mechatronic Systems

Rajarishi Sinha[1], Christiaan J.J. Paredis[1,2] and Pradeep K. Khosla[1,2]
[1]*Institute for Complex Engineered Systems, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[2]*Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA*

Key words: CAD, Port-based modeling, VHDL-AMS, Simulation, Mechatronics, Multi-body mechanics, Design consistency

Abstract: We present a framework that verifies and maintains the consistency between the representations of the form, function and behavior of mechatronic devices. These three aspects of the device represent the geometry, the task, and the actions taken to realize the task, respectively (Pahl and Beitz, 1996). They evolve simultaneously through the design process. When the designer makes a change to one aspect of the representation, our framework automatically updates all other aspects impacted by this change and reports inconsistencies. Inconsistencies occur when the kinematic behavior of the device does not match the form, or the kinematic behavior does not match the currently specified functional description. Continuous feedback of this nature shortens the design-simulate cycle for product design. To represent the components in the device we use a port-based modeling paradigm. Components encapsulate both form and behavior and are interconnected to form the system model of the device. Simulation models for the components are defined in VHDL-AMS and are solved with a commercial solver.

1. INTRODUCTION AND MOTIVATION

The realization of new mechatronic devices is characterized by ever shortening times to market, along with increasing customer demand for improved quality. In this business environment, it is important for the designer to be able to simulate the behavior of the current state of the design. As the design evolves, its form, behavior and intended function should be consistent with each other (Figure 1). In addition, information about the behavior should be automatically obtained from the CAD model of the device. Simulation of the behavior will catch inconsistencies early in the design process, reducing the need for physical prototyping and decreasing the time to market. To accomplish this goal, we are developing a software environment for simulation-based design, in which modeling and design tools are tightly integrated.

Consider the following scenario. A designer begins the design process by defining the desired kinematic function of a device. She then converts the desired function into an intended behavior described by a simple ball-and-stick model. As the design evolves, she introduces information about local geometry at the joint contact, then the complete geometry, and finally the inertial properties. At each stage, the representation is enriched and a simulation can be generated with the available information. On demand, the

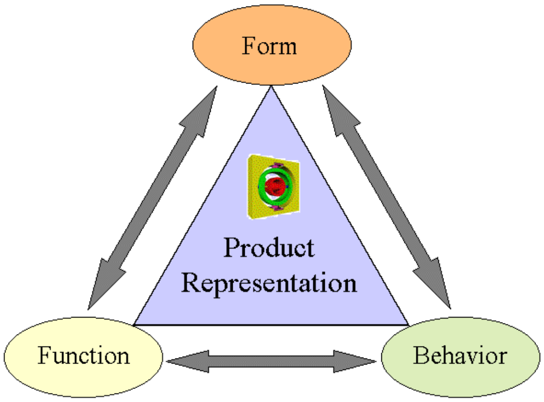

Figure 1. Relationship between the form, function and behavior of a product

kinematic representation is combined with inertial properties to generate automatically a dynamics behavioral simulation. If inertial properties are not available, then a pure form-based kinematic behavioral simulation is generated. If form (i.e. geometry and materials) is also not available, then a functional simulation is generated. The behavioral simulation results are compared with the desired function description. Inconsistencies are reported back to the designer.

This scenario illustrates the need for tools that support all the following aspects of the design process:

- Hierarchical representation of the form, function and behavior of the product incorporating kinematics.
- Automatic generation of such a representation from geometric information.
- Consistency checking between the three aspects of the representation.
- Automatic generation of a behavioral simulation from the representation.

We provide a framework that supports each of these aspects. We use the port-based modeling paradigm to describe the device. In this approach, each body or joint is represented as a component that interacts with other components in the system through interfaces known as ports. Each block represents a behavioral model that relates the port variables of the component with each other.

This framework is integrated with CAD, by providing algorithms that automatically derive the behavioral models from the geometry of the device. The parameters for the rigid body models are derived from the geometry of the parts of the device. The type and the parameters of the joint models are derived from the geometry of the part-part contacts.

VHDL-AMS (IEEE, 1999) is used to define the behavioral models of each of the components in the device. A commercial VHDL-AMS simulator is used to evaluate the models.

In section 5, we apply the framework to the mechanical design process of a 2-DOF missile seeker. During the scenario, we indicate situations where behavioral simulation of the state of the design provides feedback to the designer.

2. RELATED WORK

The related literature can be classified into the following categories: product representation for design, algorithmic modeling of multi-body systems, modeling of conserved-energy systems, and multi-domain modeling in VHDL-AMS.

Pahl and Beitz (1996) describe the geometry, the task and the actions taken to realize the task as three aspects of the representation of an artifact. Lyons et al. (1999) and Shooter et al. (2000) describe design as a process of transformation and exchange of information. They propose a framework to formalize the semantics of design information and to standardize the exchange of such information. Our framework maintains consistency between these three aspects during the transformation of design information.

Baraff (1989, 1990) and Baraff and Witkin (1992) used algorithmic methods to simulate the mechanical dynamics of multi-body systems with constraints. Such an approach involves setting up the ordinary differential equations (ODEs) that govern the dynamics of the multi-body system, and solving them using variable step numerical methods. Our framework extends this approach by allowing for the composition of models, or hierarchical systems, and for the easy definition of joint constraints. Orlandea et al. (1997a, 1997b) showed that springs and dampers could be modeled using sparse systems of linear equations. This work was subsequently incorporated in the ADAMS system (ADAMS, 1999). However, unlike ADAMS, our framework can automatically derive the behavioral models of the components from the geometry.

There exist several different modeling paradigms for describing multi-domain systems. Conserved energy-flow systems were modeled using bond graphs by Karnopp et al. (1990). Grimm and Waldschmidt (1997) describe a graph-based model to describe mixed signal systems. Linear graph techniques have been used to model rigid body dynamics (McPhee et al., 1996). More recently, Diaz-Calderon et al. (1999) have extended this linear graph theory to include *n*-terminal elements and software components. They have created a software architecture that allows for the composition of simulation models by connecting components through interfaces. Our framework is based on this approach.

VHDL-AMS (IEEE, 1999) is the IEEE standard that extends the VHDL language by adding the ability to handle continuous time signals, including non-electrical domains. Since this is a recent standard, little work has been done in realizing the potential of VHDL-AMS to implement multi-energy domain simulation. Most of the mechanics-related results come from the MEMS area (Romanowicz, 1998; Bielefeld et al., 1995). Pelz et al. (1996) describe a method of HW/SW cosimulation that uses VHDL and a proprietary analog simulation language to simulate the behavior of a wheel suspension. Our implementation uses VHDL-AMS for the behavioral modeling of mechanical systems.

3. FRAMEWORK FOR MECHANICAL COMPONENT MODELING

3.1 The Port-Based Modeling Paradigm

We view systems as structures of inter-connected elements interacting with the environment. Elements in the system interact with each other through *ports* (Diaz-Calderon et al., 1999). Ports are points on the boundary of the system where energy is exchanged between the system and the environment. Each interaction point has a port, and each port belongs to a particular energy domain.

Energy flow through a port is described by an *across* variable and a *through* variable. An across variable represents a value measured between a global reference and the port, such as a velocity, while a through variable represents a value measured through the element, such as a force. An across and through variable pair is usually chosen such that their product has units of power ($[M]^1[L]^2[T]^{-3}$). However, across variables may be replaced by their derivatives or integrals. For instance, position can be used instead of velocity.

A connection between ports results in algebraic constraints between the port variables. The constraints are described by the Kirchoffian network laws:

$$\textit{across}\ \text{variable}_A = \textit{across}\ \text{variable}_B \tag{1}$$

and

$$\textit{through}\ \text{variable}_A + \textit{through}\ \text{variable}_B = 0 \tag{2}$$

where *A* and *B* are the two components being connected. These interactions have no predefined direction, and are therefore non-causal.

Because each interaction point requires a separate port, our modeling paradigm is limited to interactions that can be modeled as being localized at a finite number of points on the boundary of the system. The paradigm further supports hierarchical model structure with any number of levels in the hierarchy (Diaz-Calderon et al., 1999). The hierarchy must be terminated by *primitive* components that are described by *declarative equations*. These equations establish differential-algebraic relationships between the variables of the ports of the component.

3.2 Port-Based Modeling of Mechanical Systems

Rigid bodies in contact with each other are constrained in their motion by the nature of the contact (Figure 2). The mechanical behavior of each rigid body is completely described by the position and orientation of the body (across variables), and the forces and torques acting on the body (through variables).

Since a rigid mass has only one set of across and through variables, it has a single port. The constraint between a pair of rigid masses is captured in a joint component that has two ports.

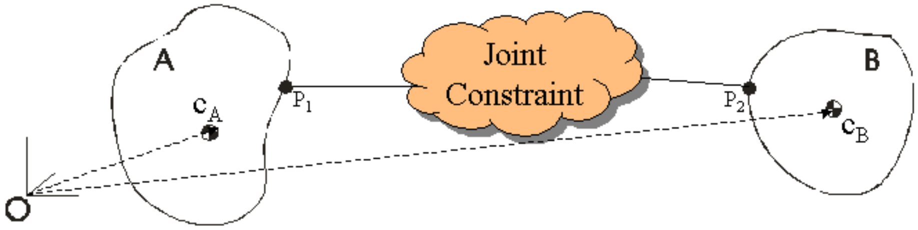

Figure 2. A joint constraint captures the contact interaction between two rigid bodies A and B. C_A and C_B are the positions of the centers of gravity and P_1 and P_2 are the contact points

Two rigid body models are never connected directly to each other; they are connected through a joint component. When the port on a mass component is connected to a port on a joint component, a *node* is implicitly created, and the two ports in question are connected to this node (Figure 3). Applied to the mechanical domain, node Equations (1) and (2) become:

$$p_A = p_B, \quad R_A = R_B \tag{3}$$

and

$$F_A + F_B = 0, \quad t_A + t_B = 0 \tag{4}$$

where $\left\langle \underbrace{p_i, R_i}_{across}, \underbrace{F_i, t_i}_{through} \right\rangle$ are the port variables for mass $i = A, B$.

In general, the internals of a component can be a behavioral model, or a sub-system consisting of interconnected components, allowing for composable and hierarchical models. The behavioral model of a component establishes relationships among the port variables in the form of ODEs or algebraic equations (AEs).

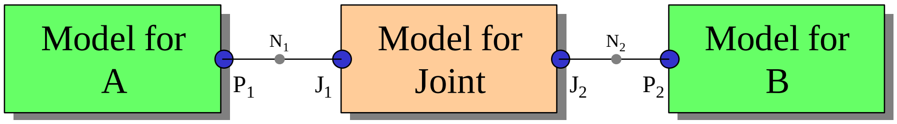

Figure 3. The schematic shown in Figure 2 can be mapped into a port-based block diagram that captures the system. Interaction of each block with other blocks is via ports, where energy flow takes place. Each block encapsulates a behavior model for that entity. A and B are rigid body components, each with a single port P_1 and P_2 respectively; Joint is the joint component with ports J_1 and J_2; N_1 and N_2 are nodes to which the ports are implicitly connected.

3.3 Rigid Body Component Model

A rigid body component is described by a point mass at the center of gravity and an inertia tensor that captures the mass distribution. All positions and orientations are expressed relative to a global frame of reference. The behavioral model for a mass component consists of the equations that relate the port variables $\langle p, R, F, t \rangle$ amongst themselves. These equations are:

$$v = \dot{p}, a = \dot{v} \tag{5}$$

and

$$\dot{R} = \frac{R \circ w}{2}, a = \dot{w} \tag{6}$$

where v and a are the linear velocity and acceleration. R is rotation represented as a quaternion. w and a are the angular velocity and acceleration. The $\circ$ operator is the quaternion multiplication operator (Dam et al., 1995; McCarthy, 1990). These port quantities are related to the other port quantities by the Newton-Euler equations:

$$F = ma, \quad t = Ia + w \times Iw \tag{7}$$

Where m is the mass, I is the inertia tensor, F is the force and t is the torque. The architecture of the model is shown in Figure 4.

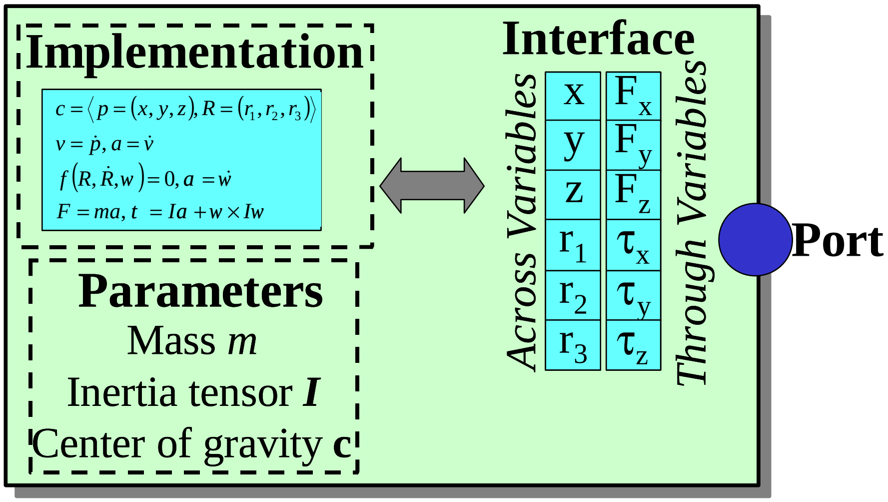

Figure 4. Representation of a mass component, with an interface, an implementation and a parameter set.

3.4 Joint Component Model

Joint component models relate the two ports of the joint through time. A joint is a constraint between a pair of masses. The behavioral model for the joint component relates the port variables of each port of the joint component via an algebraic equation or a differential-algebraic equation. A completely rigid joint, for instance, would equate the across variables at the two ports, causing the two mass components to be positioned in the same location relative to each other at all times. The architecture of the model is shown in Figure 6.

A constraint between two parts in an artifact results from the mating of the parts, i.e. by the nature of the contact between the parts. When the contact is a surface to surface contact, a lower kinematic pair is created (Figure 5).

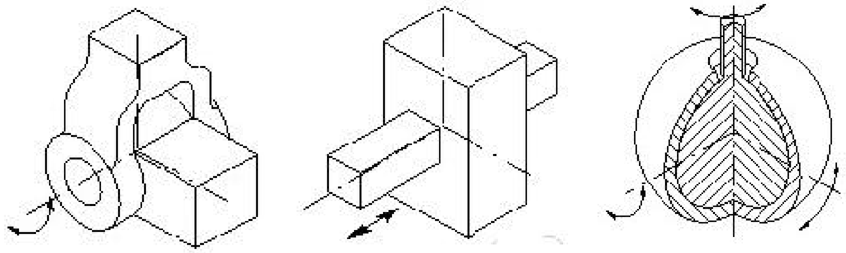

Figure 5. Revolute, prismatic and spherical joints

In our current system, we support the lower pairs of joints, with extensions to the other types of contact planned for future versions.

A revolute joint has a single degree of freedom—a pure rotation about an axis. Written in the homogeneous transform notation, we have:

$$^{F_1}T_{F_2} = R_K(q) \tag{8}$$

where $R_K(q)$ is the rotation transform about the rotation axis $\hat{A} = [k_x, k_y, k_z]$. The transform $^{F_1}T_{F_2}$ relates the transforms F_1 and F_2 of the two masses respectively by

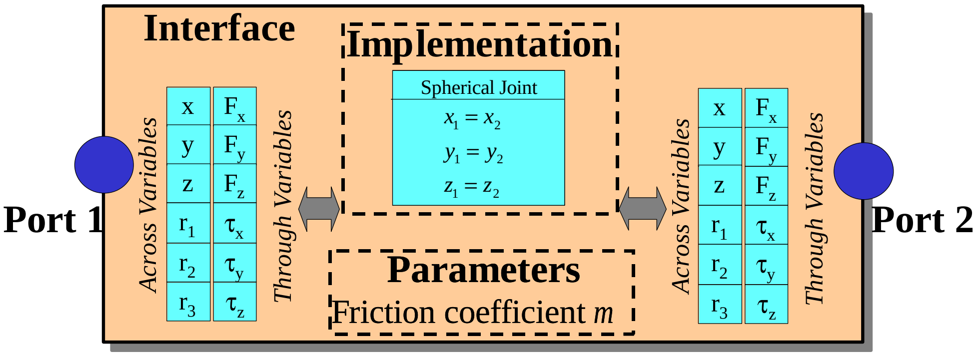

Figure 6. Representation of a joint component, with an interface, an implementation and a parameter set.

constraining them to rotate about each other about a specified axis, *A*, and by a specified angle, *q*.

A Prismatic Joint has a single degree of freedom, namely pure translation along an axis:

$$^{F_1}T_{F_2} = T_K(d) \tag{9}$$

where $T_K(d)$ is the translation transform along a general translation axis $\hat{A} = [k_x, k_y, k_z]$.

A spherical joint has three rotational degrees of freedom about a point, called the center. This is expressed by the constraint:

$$p(F_1) = p(F_2) \tag{10}$$

where $p(F)$ is the origin of the frame F. The orientation is unconstrained, realizing a spherical joint constraint.

4. DERIVATION OF THE BEHAVIORAL MODEL FROM GEOMETRY

4.1 Contact Analysis for Kinematics

To verify whether a complex device is a correct spatial realization of the intended functional design concept, we need to extract its behavior from its geometric representation. Techniques have been developed to predict the instantaneous degrees of freedom from the CAD models of parts composed of polygonal planar faces (Mattikalli et al., 1994). However, these techniques handle only parts with planar faces while most engineering devices have curved parts. When curved parts are approximated as piecewise planar parts, it is possible to overlook degrees of freedom in the device, due to erroneous collisions.

In our previous work (Sinha et al. 1998, 1999), we have shown that when rigid bodies are in contact, the kinematic degrees of freedom can be automatically derived from the nature of the contact. When two rigid parts share a surface-surface contact, every contact point is subject to a non-penetration condition. This condition requires that the instantaneous velocity of separation of the two bodies does not have a component in the direction opposite to that of the surface normal at that point. We write this condition as a linear algebraic constraint of the form:

$$(\vec{v} + \vec{w} \times \vec{r}) \bullet \vec{n} \geq 0 \tag{11}$$

where $\vec{v}$ and $\vec{w}$ are the relative translational and angular velocities between the two bodies, $\vec{r}$ is the position of the point and $\vec{n}$ is the normal at a point of contact on the surface of contact. Imposing the constraint Equation (11) at every point on the contact surface is equivalent to imposing the constraint at a finite number of points on the convex hull of the surface. Therefore, a finite number of linear constraints are imposed simultaneously for every contact surface in the device. This analysis results in a linear relationship of the form:

$$J_{assembly}\vec{v} \geq 0 \tag{12}$$

where $\boldsymbol{J}_{assembly}$ is the description of the surface-surface contacts in the system and $\vec{v}$ is the generalized velocity vector for the system. In this work, the $\boldsymbol{J}_{assembly}$ matrix is used to verify that the behavioral model is consistent with the CAD model of the system. Additionally, this matrix can be used to verify that a desired degree of freedom as specified in the functional description actually exists in the behavioral model. The properties of the $\boldsymbol{J}_{assembly}$ matrix determine the choice of kinematic joint for this pair of rigid bodies, and the parameters for this joint. For example, the basis vectors of the nullspace of the $\boldsymbol{J}_{assembly}$ matrix are the contact-preserving degrees of freedom.

Our method can infer behavior from devices with (incomplete) curved geometry, while at the same time resolving global (i.e. multi-part) constraint interactions. Linear algebra-based constraint models are derived directly from CAD models, and then converted into articulation representations suitable for assembly planning and motion simulation. Our underlying algorithms support automatic extraction of the kinematic behavioral model from the geometry. The model can be queried about candidate degrees of freedom to verify whether the actual and desired degrees of freedom match. The algorithms propagate global interactions throughout the model, and support a wide variety of geometric features that are encountered in a CAD environment. They are implemented in C++ using the ACIS solid modeler and use MATLAB for numerical computations.

4.2 Geometric Compilation for Dynamic Parameters

The CAD Model of a primitive component is a distributed geometric description of the associated rigid body. Deterministic algorithms can *compile* the CAD model to obtain lumped parameters that are used in the declarative equations of the component. These parameters include the mass of the rigid body, the inertia tensor of the rigid body and the center of gravity of the rigid body. They are obtained from the geometry and material properties by using standard methods in mechanism dynamics (Shames, 1993).

4.3 VHDL-AMS Behavioral Models from Kinematics and Dynamics Parameters

We view mechanical system design as an iterative process of configuration of components. Components include both the geometry and the behavior. A component is completely and correctly instantiated when both the geometry and behavior are specified and synchronized with each other. The previous two sections dealt with obtaining the type of joint as well as the parameters for the joint component (behavior) by reasoning on the geometry. Parameters for the declarative equations of the behavioral model are obtained from algorithmic compilation of the geometry. These are incorporated into the VHDL-AMS description of the system. A change in any of the form, function or behavioral aspects of the product representation results in a regeneration of the VHDL-AMS description, if necessary.

The environment is an agent-based Java implementation, with a product model based on our product representation. This model is queried and updated by agents and by the designer through GUIs, while iterating towards a final device design. A simulation of the current state of the design is created by automatically converting the product representation (stored as XML) into a VHDL-AMS specification (IEEE, 1999) of the system. A commercial VHDL-AMS solver is used to evaluate the models over time.

5. DESIGN SCENARIO

We now examine the design process for a missile seeker. We assume that some portions of the design will be reused, at least conceptually, from a previously designed missile seeker.

The seeker is a device with 2 rotational degrees of freedom. It carries a camera as a payload that scans a 2-dimensional area for a particular target. The seeker incorporates the articulated mechanism that realizes these degrees of freedom, as well as DC motors and controllers. The design process of the complete mechatronic device involves refining the design in all the energy domains. For this work, we will only consider the kinematics and mechanical dynamics of the design.

5.1 Design Initiation

5.1.1 Review of legacy design and simulation results

A previously designed missile seeker is retrieved from the database. It is a device that can point the camera payload to any location along a line. This seeker has one rotational degree of freedom, realized by a revolute joint mounted in the housing (Figure 7).

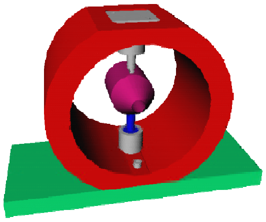

Figure 7. Legacy design search returns a 1-DOF seeker with the camera payload mounted in the center of the device.

The new seeker must have 2 degrees of freedom to scan a 2-dimensional workspace. Therefore, the designer decides that the legacy design can be modified to add another rotational degree of freedom that is coupled to the existing degree of freedom.

5.1.2 Ball-and-stick Kinematic Model

The designer constructs a ball and stick kinematic model for the new seeker by decomposing the function into two rotational DOFs. The new design includes a second rotational DOF coupled with the existing DOF (Figure 8).

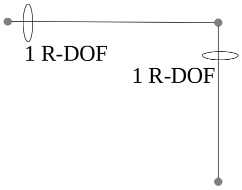

Figure 8. Kinematics model for the new 2-DOF seeker.

At this point, the ball and stick model has no geometry associated with it. It only describes the intended behavior or function of the mechanism. Nevertheless, the designer can still use our simulator to verify whether these intended kinematics satisfy the design requirements.

5.2 Iterative Design Refinement

5.2.1 Positioning of Joints

At this point in the design process, the kinematic model is realized in geometry by adding mating surfaces. To realize the degrees of freedom, the joints are positioned in space with their axes aligned with the desired DOF. A structural model of the design is created with blocks representing components in the design (Figure 9).

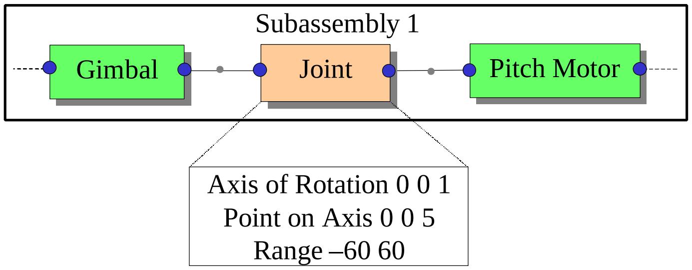

Figure 9. Structural model of a portion of the device.

The joints are grouped with geometry to form structural subassemblies as shown in Figure 9. Surface contacts are placed in space such that the specified rotational degrees of freedom meet the functional requirement.

5.2.2 Instantiation of geometry

When the designer creates a new geometrical entity in the CAD system, a corresponding mass component is instantiated in the behavioral system description. Our algorithms automatically extract the parameters for this component. Joint components are automatically computed and instantiated from the contacts between mass components. For this seeker, two motors are created, each with a revolute joint between the stator and the rotor (Figure 10). The revolute joint realizes the contact model shown in Figure 8. These motors and their potentiometers are placed at the spatial locations of the contact.

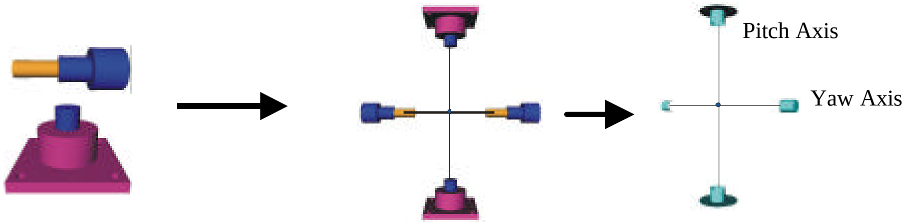

Figure 10. Instantiation of geometry at the spatial locations of the contacts realizes the degrees of freedom.

Some geometry can be reused from the design of the 1-DOF seeker. The yaw motor is similar to the motor used in the 1-DOF seeker in the legacy design database. The designer verifies that the instantiated geometry matches the desired kinematic behavior by performing a contact analysis on the fly. The analysis results in a behavioral model of the current state of the design. The behavioral model is incorporated converted into VHDL-AMS entities. A simulation is generated from the VHDL-AMS behavioral models of the mass and joint components. The designer runs the simulation to verify that the intended function is achieved.

5.3 Final Design and Dynamics Simulation

At this stage, the geometry of the payload is introduced into the design (Figure 11). In addition, a gimbal is introduced to couple the degrees of freedom while preserving symmetry in the geometry.

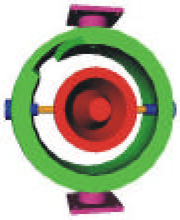

Figure 11. Partially instantiated geometry with a gimbal to couple the degrees of freedom.

Our framework compiles the CAD models to extract lumped parameters to refine the mass component models. Compilation generates values for the mass, the center of gravity and the inertia tensor. These values are used to instantiate the corresponding mass components in the VHDL-AMS system representation.

Contact analysis is performed to propagate global multi-part constraints. This verifies that the instantiated geometry still matches the desired behavioral description. Once a verified behavioral model is obtained, a kinematic simulation is performed to verify the kinematic behavior.

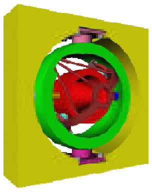

Figure 12. Complete geometry for the 2-DOF seeker.

Complete geometry is now available (Figure 12). Therefore, mass components and joint components are completely instantiated in the VHDL-AMS system description. The VHDL-AMS solver collects the declarative DAEs for the system and solves them through time. The designer uses our framework to perform a dynamics simulation of the mechanics behavior of the entire device (Figure 13).

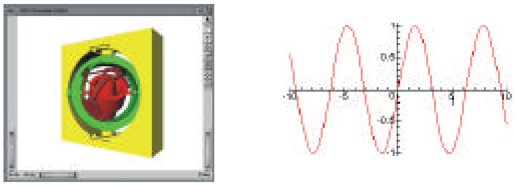

Figure 13. Dynamics simulation of the completely specified 2-DOF seeker.

The mechanics design process is now complete. The VHDL-AMS system description can be augmented with descriptions of the electrical behavior of the motors, models for the controllers, etc. Now more advanced analyses such as finite element analysis, thermal analysis and manufacturability analysis can be performed.

6. SUMMARY

We present a hierarchical framework to model mechanical components to support composable simulation. The framework verifies and maintains consistency between the form, function, and behavior of mechatronic devices, as all three aspects evolve during the design process. A change made to one aspect is automatically propagated to the other aspects, and inconsistencies are reported. As more design information becomes available during the design process, behavioral models for rigid bodies and joints are automatically created, and parameters are automatically derived. Mass and joint components are instantiated in VHDL-AMS, and a commercial solver is used to simulate the system model of the mechatronic device.

The design process for a 2-DOF missile seeker is used to demonstrate the applicability of simulation on demand. By observing the process of design, and examining the use of simulation during this process, we conclude the following: a tightly coupled iterative cycle between design and simulation can save time and money by catching errors early in the design process; design quality may improve because of incremental simulation and continuous synchronization of the form, function, and behavior of the device; automatic derivation of kinematic behavioral models from form allows for automatic instantiation and update of joint components; compilation of CAD models to extract parameters allows for automatic instantiation and updating of mass components; the port-based modeling paradigm permits hierarchical reusable models; and the use of VHDL-AMS allows for a single design/simulation framework to capture multi-domain behavioral and structural models.

ACKNOWLEDGMENTS

The authors would like to acknowledge Antonio Diaz-Calderon for his input and advice. This research was funded in part by DARPA under contract ONR # N00014-96-1-0854, by the National Institute of Standards and Technology, by the National Science Foundation under grant # CISE/115/KDI 98 73005, by the Pennsylvania Infrastructure Technology Alliance, and by the Institute for Complex Engineered Systems at Carnegie Mellon University.

REFERENCES

ADAMS, 1999, http://www.adams.com.

Baraff, D. and Witkin, A. (1992). Dynamic simulation of non-penetrating flexible bodies, *Computer Graphics*, Vol. 26, No. 2, pp. 303-308.

Baraff, D. (1990). Curved surfaces and coherence for non-penetrating rigid body simulation, *Computer Graphics*, Vol. 24, No. 4, pp. 19-28.

Baraff, D. (1989). Analytical methods for dynamic simulation of non-penetrating rigid bodies, *Computer Graphics*, Vol. 23, No. 3, pp. 223-232.

Bielefeld, J., Pelz, G., and Zimmer, G. (1995). Analog Hardware Description Languages for Modeling and Simulation of Microsystems and Mechatronics, *Proceedings of the Conference on Mechatronics and Robotics*, Stuttgart, 1995, pp. 85-92.

Dam, E.B., Koch, M., and Lillholm, M. (1998). Quaternions, Interpolation and Animation, *Technical Report DIKU-TR-98/5*, Department of Computer Science, University of Copenhagen, Denmark.

Diaz-Calderon, A., Paredis, C.J.J., and Khosla, P.K. (2000). Automatic generation of system-level dynamic equations for mechatronic systems, *Computer Aided Design*, Vol. 32, No. 5-6, pp. 339-354.

Diaz-Calderon, A., Paredis, C.J.J., and Khosla, P.K. (2000). Reconfigurable |models: A modeling paradigm to support simulation-based design, *ICES Technical Report*, Institute for Complex Engineered Systems, Carnegie Mellon University, Pittsburgh, PA, USA.

Diaz-Calderon, A., Paredis, C.J.J., and Khosla, P.K. (1999). A Composable Simulation Environment for Mechatronic Systems, *Proceedings of the 1999 SCS European Simulation Symposium*, October 26-28, 1999, Erlangen, Germany.

Grimm, C., and Waldschmidt, K. (1997). KIR - A Graph-Based Model for Description of Mixed Analog/Digital Systems, *Proceedings of the EURO-DAC '96 with EURO-VHDL '96*, September 16-20, 1996, Geneva, Switzerland.

IEEE. (1999). *1076.1-1999 IEEE Standard VHDL Analog and Mixed-Signal Extensions*.

Karnopp, D. C., Margolis, D. L., and Rosenberg, R.C. (1990). *System dynamics: A unified approach*. John Wiley & Sons, Inc. New York, NY.

Lyons, K., Hart, P., Shooter, S., and Keirouz, W. (1998). The Open Assembly Design Environment Project: An Architecture for Design Agent Interoperability, *Proceedings of the 1999 ASME Design Engineering Technical Conferences*, September 1999, Las Vegas, NV.

Mattikalli, R., Baraff, D., and Khosla, P.K. (1994). Finding All Gravitationally Stable Orientations of Assemblies, *Proceedings of the IEEE International Conference of Robotics and Automation*, May 1994, San Diego, CA, pp. 251-257.

McCarthy, J. M. (1990). *An Introduction to Theoretical Kinematics*, The MIT Press, Cambridge, MA, USA.

McPhee, J. J., Ishac, M. G., and G. C. Andrews. (1996). Wittenburg's formulation of multibody dynamics equations from a graph-theoretic perspective, *Mechanism and Machine Theory*, Vol. 31, pp. 202-213.

Orlandea, N., Calahan, D.A., and Chace, M.A. (1977a). A Sparsity-Oriented Approach to the Dynamic Analysis and Design of Mechanical Systems - Part 1, *Transactions of the ASME - Journal of Engineering for Industry*, pp. 773-779.

Orlandea, N., Chace, M.A., and Calahan, D.A. (1977b). A Sparsity-Oriented Approach to the Dynamic Analysis and Design of Mechanical Systems - Part 2, *Transactions of the ASME - Journal of Engineering for Industry*, pp. 780-784.

Pahl, G., and Beitz, W. (1996). *Engineering Design: A Systematic Approach, 2nd Edition*, Springer-Verlag, London, U.K., ISBN 3-540-19917-9.

Pelz, G., Bielefeld, J., Hess, G., and Zimmer, G. (1996). Hardware/Software-Cosimulation for Mechatronic System Design, *Proceedings of the EURO-DAC '96 with EURO-VHDL '96*, September 16-20, 1996, Geneva, Switzerland.

Romanowicz, B.F. (1998). *Methodology for the Modeling and Simulation of Microsystems*, Kluwer Academic Publishers.

Shames, I.H. (1993). *Engineering Mechanics*, Prentice-Hall, NJ, USA.

Shooter, S. B., Keirouz, W., Szykman, S., and Fenves, S.J. (2000). A Model for the Flow of Design Information, *Proceedings of the 2000 ASME Design Engineering Technical Conferences (12th International Conference on Design Theory and Methodology)*, September 2000, Baltimore, MD.

Sinha, R., Gupta, S.K., Paredis, C.J.J., and Khosla, P.K. (1999). Extracting Articulation Models from CAD Models of Parts with Curved Surfaces, *ICES Technical Report*, Institute for Complex Engineered Systems, Carnegie Mellon University, Pittsburgh, PA, USA.

Sinha, R., Paredis, C.J.J., Gupta, S.K., and Khosla, P.K. (1998). Capturing Articulation in Assemblies from Component Geometry, *Proceedings of the ASME Design Engineering Technical Conference*, September 1998, Atlanta, Georgia, USA.