1999 American Control Conference, San Diego California, June 1999.

On the Synthesis of the System Graph for 3D Mechanics

Antonio Diaz-Calderon[1], Christiaan J. J. Paredis, Pradeep K. Khosla
Institute for Complex Engineered Systems
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This paper presents a methodology for deriving the system graph of a 3D mechanism from CAD models. That is, a linear graph that captures the energy flow in a system. This work is part of a larger research effort in composable simulation*. In composable simulation, CAD models of system components are augmented with simulation models describing the component's dynamic behavior in different energy domains. By composable simulation we mean then the ability to automatically generate system-level simulations through composition of individual component models. From the system graph, the system-level dynamic equations can be derived independently of the underlying energy domains.*

1. Introduction

The work presented in this paper is part of a larger effort to develop a framework for *composable simulation*. By composable simulation we mean the ability to generate system-level simulations automatically by simply organizing the system components in a CAD system. A system component can be either a physical component (electrical motor, gearbox, etc.) or an information technology component (embedded controller or other software component). Physical components can have more than one simulation models associated with them describing their dynamics in multiple energy domains. Of particular importance is the dynamic model of the 3D mechanism. When physical components are combined into a complete system, the dynamic model for the 3D mechanism should reflect all the interactions between parts: joints, forces, etc. Although there exist commercially available systems for the derivation of the dynamic equations of mechanical systems [9], [11], [15] these systems cannot be readily applied to composable simulation or to automatic model refinement. Composing simulation models of system components requires that the models be given in explicit symbolic form, because there may exist dependencies that require the simultaneous solution of equations. Such dependencies make the simultaneous solution of the dynamic equations much more difficult when the dynamic model of a 3D mechanism is only available in numerical form. Furthermore, changing the fidelity of that model on the fly, may not be possible at all. A graph-theoretic approach, on the other hand, can be used to provide different levels of model accuracy.

In this paper, we will address the issue of generating a topological representation of the 3D mechanism that can be used by a graph-theoretic dynamic analysis system to generate the dynamic equations.

2. Related work

The relationship between physical systems and linear graphs was first recognized by Trent [14] and by Brannin [1]. Roe [10] and Koenig [6] apply the theory of linear graphs to the systems theory and provide important results that can be directly related to the two basic laws in circuit theory: Kirchhoff's voltage and current laws. Linear graph theory has been used in the analysis of rigid body dynamics [7], [8], [13].

Composition of simulation models can be accomplished by combining fundamental building blocks described in a high level object-oriented modeling language [2], [5]. The object-oriented approach facilitates model reuse and simplifies maintenance. Using these modeling languages, software executables can be generated automatically from individual sub-models and the interactions between them.

Synthesis of the system-level dynamic equations for mechatronic systems based on a linear graph representation of the system is presented in [4]. In this work, system components are represented as linear graphs (possibly more than one) that are combined into a single system graph. From the system graph the system-level dynamic equations can be derived independently of the underlying energy domains. To address the problem of software integration when composing information technology system components, we have defined a software architecture that provides the infrastructure to dynamically define an arrangement of software modules for simulation [3].

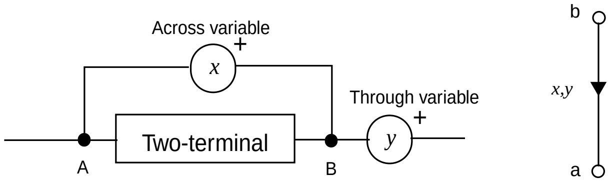
3. Graph-theoretic component modeling

Linear graph theory is a branch of mathematics that studies the algebraic and topological properties of graphs. Trent [14] showed that there is an isomorphism between a physical system and a linear graph based on two types of measurements: *across* and *through* measurements (Figure 1). The variables associated with this pair of measurements are

1. Corresponding author.

called *terminal variables*. The terminal variables are chosen such that the power of the corresponding component is characterized by their product. The mathematical relations between terminal variables define the component's physical characteristics and are called *terminal equations*. Based on the form of the terminal equations, one can distinguish three types of elements: passive elements that can be further divided into dissipative and non-dissipative (energy storage elements), sources and transducers.

Figure 1: Through and across measurements on a general two-terminal element and its terminal graph.



All derivatives of an across or through variable are across or through variables as well.

3.1. Constraint equations

The topological and algebraic properties of the system graph can be described by two matrices, namely the incidence and circuit matrix. The incidence matrix $\mathbf{A}$ of a connected directed graph $\mathbf{G}$ with v vertices and e edges is a $(v-1)\times e$ matrix where the entries can be -1, 1, or 0 if the edge is positively, negatively or not incident onto node v respectively. Furthermore, if $\mathbf{T}$ is a tree of the connected graph $\mathbf{G}$, the $v-1$ columns of $\mathbf{A}$ that correspond to the branches of the tree $\mathbf{T}$ constitute a nonsingular matrix. Thus if a tree is chosen and the columns of $\mathbf{A}$ are properly arranged, the matrix $\mathbf{A}$ can be partitioned into the $(v-1)\times(v-1)$ submatrix $\mathbf{A}_T$, referring to the branches of the tree only, and the $(v-1)\times(e-v+1)$ submatrix $\mathbf{A}_C$, referring to the chords of the cotree:

$$\mathbf{A} = \begin{bmatrix}\mathbf{A}_T & \mathbf{A}_C\end{bmatrix} \tag{1}$$

If a tree $\mathbf{T}$ is selected, we can identify $v-1$ pairs of terminal variables (x_T, y_T) associated with the *branches* of the tree and $e-v+1$ terminal variables (x_C, y_C) associated with the *chords* of the cotree. If the system graph is divided in two non-connected subgraphs by a cut including exactly one branch of $\mathbf{T}$ and some chords, the cut is unique. It is clear that for a tree $\mathbf{T}$ with $v-1$ branches, there will be as many unique cuts. The sum of the vertex equations for all nodes within the cut-set[1] contains only through variables corresponding to the cut-set elements. The set of all cut-set equations can be written in the form:

$$\begin{bmatrix}\mathbf{U}_T & \mathbf{A}_C\end{bmatrix}\begin{bmatrix}\mathbf{y}_T \\ \mathbf{y}_C\end{bmatrix} = \mathbf{0} \tag{2}$$

where $\mathbf{U}_T$ is a unit matrix of dimension $(v-1)$ and the cut-set matrix $\begin{bmatrix}\mathbf{U}_T & \mathbf{A}_C\end{bmatrix}$ can be derived by applying row operations on the reduced incidence matrix $\mathbf{A}$.

Each chord in the system graph is uniquely associated with a loop in the system. For a given loop, its orientation will be determined by the orientation of the defining chord. A new matrix, namely the circuit matrix $\mathbf{B}$ that captures the connectivity relations between circuits and edges can be defined. The circuit equations associated with each circuit can be written in the form:

$$\begin{bmatrix}\mathbf{B}_T & \mathbf{U}_C\end{bmatrix}\begin{bmatrix}\mathbf{x}_T \\ \mathbf{x}_C\end{bmatrix} = \mathbf{0} \tag{3}$$

Where $\mathbf{U}_C$ is a square unit matrix with dimensions equal to the number of chords in the system graph.

The circuit matrix $\mathbf{B}$ captures the connectivity relations between circuits and edges. If a tree $\mathbf{T}$ has been chosen, we can arrange the columns of $\mathbf{B}$ such that it can be partitioned into the $(e-v+1)\times(v-1)$ submatrix $\mathbf{B}_T$ referring to the branches of the tree and the $(e-v+1)\times(e-v+1)$ submatrix $\mathbf{B}_C$ referring to the chords of the cotree.

From the *principle of orthogonality* which states that the vector space spanned by the rows of matrix $\mathbf{A}$ and the vector space spanned by the rows of matrix $\mathbf{B}$ are orthogonal complements [6] (i.e., $\mathbf{A}\mathbf{B}^{\mathrm{T}} = \mathbf{0}$ and $\mathbf{B}\mathbf{A}^{\mathrm{T}} = \mathbf{0}$) we can obtain an expression for $\mathbf{B}_T$ as follows:

$$\mathbf{B}_T = -\mathbf{A}_C^{\mathrm{T}}(\mathbf{A}_T^{-1})^{\mathrm{T}} \tag{4}$$

The $e-v+1$ fundamental circuit equations and the $v-1$ cut-set equations of a system graph $\mathbf{G}$ with e edges and v vertices are referred to as the *constraint equations* of the system.

4. Synthesis of the System Graph for 3D Mechanics

The dynamic equations of the 3D mechanics of the system are derived using an analysis program (Dynaflex) which is specifically designed for the analysis of three dimensional constrained mechanical systems [13]. Dynaflex is a graph-

1. A cut-set is a set of edges that divide the graph into exactly two components.

theoretic approach in which the connectivity of the bodies in the mechanism and the forces acting on them are represented by a linear graph (mechanical system graph).

The system graph of the mechanical system captures the topology of the mechanism. However, to have a complete model, geometric and inertial information must be added to the topology. This information is derived from the Intelligent Assembly Modeling and Simulation (IAMS) tool kit developed in our lab [12]. Based on the geometry of the mechanism, the IAMS tool kit will automatically determine the instantaneous kinematic relationships between components in the mechanism. IAMS further provides information about the origin of the inertial frame, center of mass of each body, location of articulation points in each body, type of joint, and points of application of internal forces.

Similar to the basic modeling elements we defined in Section 3, Dynaflex provides a set of modeling elements for mechanical systems, including [13]: body elements, arm elements (position vectors), motion and force drivers, spring-damper-actuator elements, and joint elements.

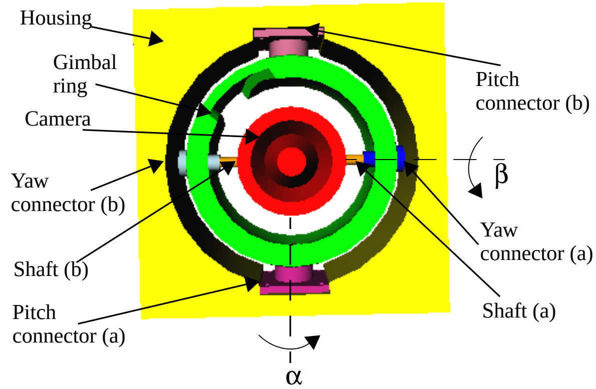
The process for obtaining the graph representation suitable for Dynaflex consists of three steps. First an *extended* system graph is generated. This step maps the geometry of the mechanism directly into a linear graph representing its topology. The second step identifies composite bodies consisting of rigidly connected subcomponents. In a final step, composite bodies are replaced by single bodies reducing the system graph to a minimal graph with the same topological properties.

The generation of the system graph involves a direct translation of the kinematic information into the linear graph representation. In general, the result of the first stage is an extended system graph that includes all kinematic information including fixed joints and redundant joints. However, to avoid structural singularities and indexing problems, and to improve the efficiency of the symbolic computations in Dynaflex, we simplify this initial system graph by lumping all rigidly connected bodies into a single composite body.

Composite bodies are identified by performing a depth-first traversal on the extended system graph starting from the node representing the center of mass of a body. The algorithm explores all paths created by rigid connections and collects all bodies along the path into a single composite body.

The last stage in the synthesis of the system graph is to perform the reduction process that will combine the identified bodies into single composite bodies and remove redundant joints. In this context, redundant joints are joints that duplicate already existing kinematic constraints; for instance, co-linear revolute joints. They need to be removed from the representation to keep Dynaflex from interpreting the result as an overconstrained system. Possibly overconstrained systems can be recognized in the system graph as kinematic loops. We use the IAMS tool kit to determine whether a kinematic loop contains a redundant joint or whether it results in an overconstrained system.

Figure 2: **Missile seeker.**



As an example of how these steps are followed consider the design of a missile seeker shown in Figure 2.

This design contains 9 bodies: housing, gimbal ring, camera, pitch connector (2) yaw connector (2), shaft (2). A kinematic description of the system reveals that there are a number of bodies that may be combined to form composites:

Table 1: **Kinematic description for the seeker system**

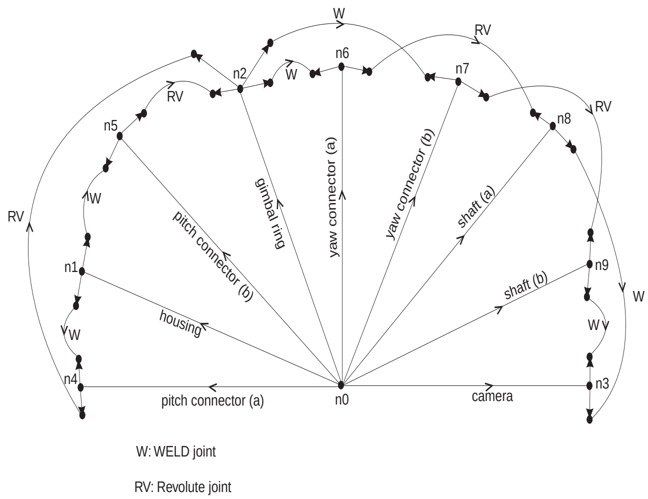
Type of joint	Reference body	Secondary body
FIXED	housing	pitch connector (a)
FIXED	housing	pitch connector (b)
REVOLUTE*	pitch connector (a)	gimbal ring
REVOLUTE	pitch connector (b)	gimbal ring
FIXED	gimbal ring	yaw connector (a)
REVOLUTE*	yaw connector (a)	shaft (a)
FIXED	gimbal ring	yaw connector (b)
REVOUTE	yaw connector (b)	shaft (b)
FIXED	shaft (a)	camera
FIXED	shaft (b)	camera

From the kinematic description shown in Table 1, the first stage of our derivation generates an extended system graph shown in Figure 3. Secondly, the stage that identifies the composites finds that the shaft (b), the camera and the shaft (a) can all be combined into one composite body. The composite bodies found are listed in Table 2:

Table 2: **Composite bodies**

Body	Components		
BODY_1	shaft (b)	camera	shaft (a)
BODY_2	housing	pitch connector (b)	pitch connector (a)
BODY_3	gimbal ring	yaw connector (b)	yaw connector (a)

Figure 3: **Extended system graph. Only joint and body elements are shown for clarity**



Finally, the reduction stage yields the following kinematic relations:

Table 3: **Kinematic description for the composite bodies in the seeker**

Type of Joint	Reference body	Secondary body
REVOLUTE	BODY_2	BODY_3
REVOLUTE*	BODY_2	BODY_3
REVOLUTE	BODY_3	BODY_1
REVOLUTE*	BODY_3	BODY_1

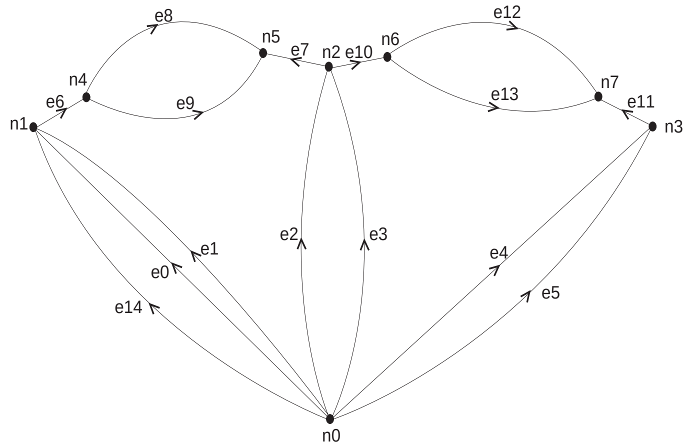
Notice that there are two revolute joints per pair of composite bodies. Kinematic analysis reveals that the rotation axes of each pair of joints coincide. For the Dynaflex analysis, one of the two points is removed to avoid concluding overconstrained kinematics; only the joints marked with an asterisk are considered: these joints correspond to actuated joints in the design. At the end of the reduction process, we obtain the reduced system graph shown in Figure 4.

To conclude the Dynaflex description, dynamic elements are introduced consisting of generalized forces provided by motors, external forces applied to the bodies, and forces acting between two bodies. For this example, only two force elements are introduced: e9 and e13, which are the result of the motors built into the corresponding joints. Furthermore, we introduce gravity forces acting on the bodies at their center of mass (e1, e3, e5) representing the weight of BODY_2, BODY_3, and BODY_1, respectively.

5. System equations

Once the mechanical system has been described as a system graph, Dynaflex can provide the dynamic equations for the mechanism. These equations are then combined with the equations derived for the mechatronic system once it has been described as a system graph. This can be done without the need to consider the underlying physics in each of the

Figure 4: **Reduced system graph.**



energy domains involved in the system. As mentioned in Section 3, the system equations can be derived by simultaneously considering the *e* terminal equations and the *e* independent topological constraints. To produce a set of differential equations that describe the system, we need to address two issues: 1) which topological constraints need to be considered, and 2) which of the two system variables (across or through) should be the independent variable in each of the *e* terminal equations? In [4], we present an algorithm based on the selection of a minimum cost spanning tree to select the constraint equations and to decide what variable should be independent.

The terminal equations plus any independent set of *e* constraint equations unambiguously define the dynamics of the system. However, before these equations can be numerically solved they must be expressed in state space form in which the derivatives of a state x are expressed as explicit functions of the states and time. To meet this requirement, we present in [4] an algorithm to derive the state-space form of the dynamic equations.

6. Conclusions

In this paper, we have presented a methodology to derive a linear graph representing the 3D mechanics of a mechatronic system. This approach is part of a framework for composable simulation in which component models are automatically combined to create system-level simulation models. Dynaflex is used to derive the dynamic equations of the 3D mechanical system. While the dynamic equations for the rest of the mechatronic system are derived from the system graph. An algorithm based a minimum cost spanning tree results in a set of equations in state-space form.

7. Acknowledgments

We would like to thank Dr. John J. McPhee and Dr. Pengfei Shi from the Motion Research Group in the Department of Systems Design Engineering at the University of Waterloo for their help with the use of Dynaflex.

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 Pennsylvania Infrastructure Technology Alliance, by the National Council of Science and Technology of Mexico (CONACyT), and by the Institute for Complex Engineered Systems at Carnegie Mellon University.

8. References

[1] F. H. Branin, "The algebraic-topological basis for network analogies and the vector calculus," presented at Symposium on Generalized Networks, Polytechnic Institute of Brooklyn, 1966.

[2] F. E. Cellier, Automated formula manipulation supports object-oriented continuous-system modeling. IEEE Control Systems, Vol. 2, No. 13, April 1993. pp. 28-38.

[3] A. Diaz-Calderon, C. J. J. Paredis, and P. K. Khosla, "A modular composable software architecture for the simulation of mechatronic systems," Proceedings of the ASME Design Engineering Technical Conference, 18th Computers in Engineering Conference, Atlanta, GA, 1998.

[4] A. Diaz-Calderon, C. J. J. Paredis, and P. K. Khosla, "Automatic generation of system-level dynamic equations for mechatronic systems," Tech. Report, Institute for Complex Engineered Systems, Carnegie Mellon University, Pittsburgh PA, 1999.

[5] H. Elmqvist, and D. Bruck. "Constructs for object-oriented modeling of hybrid systems." In Eurosim Simulation Congress, Vienna, Austria, September 11-15, 1995.

[6] H. E. Koenig, Tokad, Y., Kesavan, H. K., and H. G. Hedges, Analysis of discrete physical systems. New York: MacGraw-Hill, 1967.

[7] T. W. Li, and G. C. Andrews, "Application of the vector-network method to constrained mechanical systems," *Mechanisms, Transmissions, and Automation in Design*, vol. 108, pp. 471-480, 1986.

[8] J. J. McPhee, "On the use of linear graph theory in multibody system dynamics," *Nonlinear Dynamics*, vol. 9, pp. 73-90, 1996.

[9] P. E. Nikravesh and I. S. Chung, "Application of euler parameters to the dynamic analysis of three-dimensional constrained mechanical systems," *Journal of Mechanical Design*, vol. 104, pp. 785-791, 1982.

[10] P. H. O'n. Roe, *Networks and systems*. Reading, Massachusetts: Addison-Wesley, 1966.

[11] N. Orlandea. "Node-analogous, sparsity-oriented methods for simulation of mechanical dynamic systems," Ph.D. Thesis, Mechanical Engineering, University of Michigan, 1973.

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

[13] P. Shi. "Flexible multibody dynamics: A new approach using virtual work and graph theory," Ph. D. Thesis, Systems Design Engineering, University of Waterloo, Waterloo, Canada, 1998.

[14] H. M. Trent, "Isomorphisms between oriented linear graphs and lumped physical systems," *The Journal of the Acoustical Society of America*, vol. 27, pp. 500-527, 1955.

[15] J. Wittenburg and U. Wolz, "MESA VERDE: A symbolic program for nonlinear articulated-rigid-body dynamics," presented at ASME Design Engineering Conference, Cincinnati, OH, 1985.