ELSEVIER

Advances in Engineering Software 35 (2004) 273–287

ADVANCES IN ENGINEERING SOFTWARE

www.elsevier.com/locate/advengsoft

An object-oriented design of a finite element code: application to multibody systems analysis

V. Kromer[a,*], F. Dufossé[b], M. Gueury[a]

[a]*ERIN, ESSTIN, Université Henri Poincaré, Nancy 1, 54519 Vandoeuvre-lès, Nancy, France*
[b]*SA VALUTEC, Université de Valenciennes, 59314 Valenciennes, France*

Received 30 October 2002; revised 22 March 2004; accepted 30 March 2004

Abstract

This paper will describe one approach to the design and implementation of a multibody systems analysis code using an object-oriented architecture. The principal objective is to show the adequacy between object-oriented programming and the finite element method used for the treatment of three-dimensional multibody flexible mechanisms. It will show that object-oriented programming greatly simplifies the choice and the implementation of other formalisms concerning polyarticulated systems, thus conferring high flexibility and adaptability to the developed software.
© 2004 Elsevier Ltd. All rights reserved.

Keywords: Object-oriented programming; Multibody systems; C++; Finite element analysis

1. Introduction

The finite element method (FEM) has become the most popular numerical method for solving a wide variety of complex engineering problems. Over the years, FEM codes have emphasized the use of the Fortran programming language, with a so-called 'procedural programming'. As a result, the codes contain numerous complex data structures, which can be called anywhere in the program. Because of this global access, the flexibility of the software is decreased: when a FEM program has many computational capabilities, it becomes very difficult to maintain the code and even more difficult to enlarge the program codes. The recoding of these finite element programs in a new language is not a solution to this inflexibility problem, thus a redesign is needed.

Object-oriented programming is currently seen as the most promising way of designing a new application. It leads to better structured codes and facilitates the development, the maintenance and the expansion of such codes.

The object-oriented philosophy was proposed as a general methodology for FEM implementation for the first time in Ref. [1]. Over the past decade, it has been successfully applied to various domains in finite element developments: constitutive law modeling [2,3], parallel finite element applications [4,5], rapid dynamics [6], multi-domain analysis for metal cutting, mould filling and composite material forming [7–9], coupled problems [10], non-linear analysis [11], symbolic computation [12,13], variational approach [14], finite element analysis program architecture [15–20], neural networks [21], impact simulation [22], among others.

However, little effort has been made to implement object-oriented programming in multibody systems analysis: like many other engineering applications, multibody systems analysis codes (ADAMS [23], DADS [24], MBOSS [25]) are written in Fortran. Therefore, the main objective of this work is to describe one approach to the design and implementation of a multibody systems analysis code using an object-oriented architecture.

The first part of the paper deals with the formalism used for the treatment and the resolution of multibody systems. It will be shown that the structure of multibody systems presents similarities with object-oriented concepts and lends itself very well to object-oriented programming techniques [26,27]. The principal features of object-oriented programming are summarized in the second part of this paper. The architecture of the computational engine of the software

* Corresponding author. Tel./fax: +33-3-83-68-50-91.
E-mail addresses: kromer@esstin.uhp-nancy.fr (V. Kromer), francois.dufosse@univ-valenceiennes.fr (F. Dufossé), gueury@esstin.uhp-nancy.fr (M. Gueury).

0965-9978/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2004.03.008

is then presented, with the description of the most important classes. Emphasis is placed on the adequacy between object-oriented programming and the FEM within a given formalism and given hypotheses. It will be shown that object-oriented programming greatly simplifies the choice and the implementation of other formalisms concerning polyarticulated systems, conferring high flexibility and adaptability to the developed software. The last section is dedicated to numerical examples.

2. Governing equations and numerical resolution for multibody systems

At each step of the formalism, choices have been made in order to favor concepts such as modularity, polyvalence and evolutivity, which fit particularly well to the object-oriented philosophy.

2.1. Flexible beam dynamic formulation

In order to describe the dynamics of a flexible beam, an inertial reference frame (**e**) (orthogonal basis vectors $\vec{e}_i$, $i = 1–3$) is used for the description of the translational motion, whereas a body-fixed frame (**b**) (orthogonal basis vectors $\vec{b}_i$, $i = 1–3$) is used for the rotary motion [28,29].

The motion due to rigid motion is not distinguished from that due to the deformations. Moreover, the translational inertia is completely decoupled from the rotaty inertia. The advantage to this is that the beam inertia is identical in form to that of rigid body dynamics.

Thus, the same formalism can be used for mechanisms containing rigid elements as well as deformable elements.

The location from the inertial origin of an arbitrary point P on the beam (Fig. 1) is represented by the following position vector:

$$\vec{r} = \vec{X}^e + \vec{u}^e + \vec{l}^b \tag{1}$$

where $\vec{X}$ is the position vector of a point of the original neutral axis, $\vec{u}$ is the total translational displacement vector of the neutral axis, $\vec{l}$ is a vector connecting the beam neutral axis to the material point P located on the deformed beam cross-section.

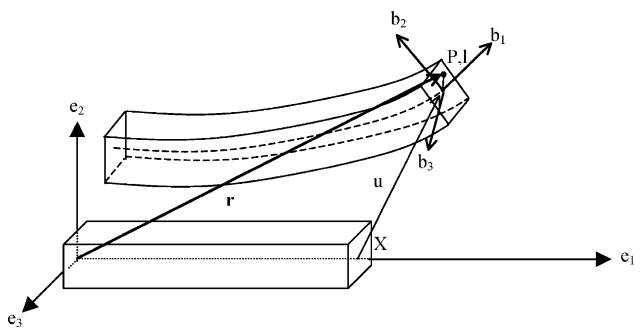

Fig. 1. Spatial beam kinematics.

The notation $\_^e$ or $\_^b$ in Eq. (1) indicates that the quantity is expressed with respect to the frame (**e**) or (**b**).

The orientation of the body-fixed reference frame is expressed with respect to the inertial reference frame through an orthogonal transformation matrix, as:

$$(\mathbf{b}) = [R](\mathbf{e}) \tag{2}$$

The body frame components of the angular velocity tensor are obtained by:

$$[\tilde{\omega}] = -[\dot{R}][R]^{\mathrm{T}} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{3}$$

where ω_i $(i = 1–3)$ are the components of the angular velocity vector $\vec{\omega}$, the notation $[\dot{x}]$ indicates the time derivative, the notation $[\tilde{x}]$ indicates a skew symetric tensor and the notation $[x]^{\mathrm{T}}$ indicates a transpose matrix.

The final discrete equations of motion of a flexible beam element are given as:

$$\begin{bmatrix} m & 0 \\ 0 & J \end{bmatrix} \begin{Bmatrix} \ddot{\vec{u}} \\ \dot{\vec{\omega}} \end{Bmatrix} + \begin{Bmatrix} \vec{0} \\ \vec{D} \end{Bmatrix} + \begin{Bmatrix} \vec{S}^e \\ \vec{S}^b \end{Bmatrix} = \begin{Bmatrix} \vec{F}^e \\ \vec{F}^b \end{Bmatrix} \tag{4}$$

where $[m]$ and $[J]$ represent the mass and inertia matrices; $\ddot{\vec{u}}$ and $\dot{\vec{\omega}}$ represent the nodal accelerations vectors; $\vec{D}$ represents the non-linear acceleration, $\vec{S}^{e,b}$ and $\vec{F}^{e,b}$ represent the internal and external force vectors partitioned into translational and rotational parts, respectively.

These equations can be specialized to the case of static equilibrium as:

$$\vec{S} = \vec{F} \tag{5}$$

The equations of motion (4) can also represent a rigid body by setting the internal force vector $\vec{S}$ to zero.

Therefore, the unconstrained equations of an arbitrary configuration of flexible beams and rigid bodies can be written in terms of one set of kinematical coordinates denoting both the nodal coordinates of the flexible members and the physical coordinates of the rigid bodies.

2.2. Equations of motion for multibody systems

We have chosen to use the Lagrange multiplier technique to couple the algebraic constraint equations with the differential equations of motion of the assembled mechanism [30–32]. The incorporation of the constraints via the Lagrange multiplier technique is straightforward, as the inertially-based degrees of freedom of the beam components, which embody both the rigid and deformation motions, are kinematically of the same sense as the physical coordinates of rigid body components.

Two types of constraint exist: holonomic or configuration constraints and non-holonomic or motion constraints.

Holonomic constraints are formulated as implicit functions of the displacement coordinates and eventually time.

دریافت فوری ← متن کامل مقاله

ISIArticles
مرجع مقالات تخصصی ایران

✓ امکان دانلود نسخه تمام متن مقالات انگلیسی
✓ امکان دانلود نسخه ترجمه شده مقالات
✓ پذیرش سفارش ترجمه تخصصی
✓ امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
✓ امکان دانلود رایگان ۲ صفحه اول هر مقاله
✓ امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
✓ دانلود فوری مقاله پس از پرداخت آنلاین
✓ پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات