Advances in Engineering Software 37 (2006) 287–311

ADVANCES IN ENGINEERING SOFTWARE

www.elsevier.com/locate/advengsoft

jNURBS: An object-oriented, symbolic framework for integrated, meshless analysis and optimal design

Xuefeng Zhang, Ganesh Subbarayan *

School of Mechanical Engineering, Purdue University, 585 Purdue Mall, West Lafayette, IN 47907-2088, USA

Received 15 February 2005; received in revised form 30 June 2005; accepted 9 August 2005
Available online 10 October 2005

Abstract

In this paper, we propose a design-analysis integrated CAD framework termed jNURBS. jNURBS, developed using the Java language, is an extensible, Object-Oriented framework that enables meshless analysis of physical behavior and optimal design. The geometry as well as the analysis fields (displacement, temperature, etc.) are described mathematically using a common representation, namely the Non-Uniform Rational B-Spline (NURBS). Thus, NURBS serves to design the geometry as well as carryout meshless analysis thereby integrating the design and analysis in an efficient manner. The program kernel provides tools to symbolically describe complex multi-physics problems, methods to manipulate the NURBS entities, a set of primitive NURBS entities, and an iterative optimization solver. The problems are symbolically defined using a newly developed high-level, natural language description through an interface termed JNS (jNURBS Script). A number of example problems, selected from meshless structural analysis, material microstructure simulation, shape optimal design, and equilibrium shape of droplets, are presented to demonstrate the effectiveness and versatility of the framework.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: NURBS; Meshless analysis; Object-oriented framework; Shape optimal design; Constructive solid analysis; Shapes of droplets; Microstructure design

1. Introduction

Many engineering problems involve large, iterative shape changes guided by evaluative analyses of physical behavior. Such problems occur in shape optimal design, free surface fluid dynamics, and in fracture simulation to name a few. In shape optimal design, the primary application considered in the present paper, one needs to iteratively modify the shape of the design domain based on the analysis results at each iteration. At the present time, finite element based codes have been developed for shape optimal design [1–5]. The finite element method, as is currently used in these tools, is arguably disadvantageous since the geometry and analysis field representations are not integrated. In other words, even a small change in geometry must be accompanied by recreation of the final geometry, re-meshing or mesh modification (provided the change is small) to recreate the discretized analysis field. This is because the tools do not exploit the information inherent in the CAD procedures used to construct the modified geometry. Although significant advances in automatic mesh generation have mitigated the challenge of re-meshing, the need for meshing, which is costly and sometimes involves manual intervention, remains a critical bottleneck. Meshless methods in general, and strategies that provide particularly tight integration with geometry representation have the potential to analyze the effects of large changes in geometry during optimal design in a natural way.

Currently, many meshless schemes have been developed [6] and successfully applied to many classes of problems; we cite the following as examples of this research: fracture [6], hyperelastic, elasto-plastic, rigid-to-flexible-body contact [7], free surface fluid dynamics [8] and shape optimal design [9,10]. In the existing literature, the architecture of meshless software packages appears to have received very limited attention. SAGE (Semi-Analytic Geometry Engine) [11] is a computer system for meshfree analysis employing the R-function method. It can be used to analyze two-dimensional heat transfer and plate natural vibration problems on multiply connected curved polygonal domains. SNAW (Structural Non-linear Analysis Workspace), developed by Hardee et al. [12], is applicable to hyperelastic and elasto-plastic materials, using explicit and implicit time integrations, and involving rigid-to-flexible-body frictional contact conditions. These tools are designed to solve limited, pre-defined types of problems.

* Corresponding author. Tel.: +1 765 494 9770; fax: +1 765 494 0531.
E-mail address: ganeshs@purdue.edu (G. Subbarayan).

0965-9978/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2005.08.001

Neither program appears to have special features that enable the efficient design of shapes or topologies or enable extensibility to new classes of problems governed by a different physical behavior.

In general, an important attribute of a flexible framework is its extensibility. Object-oriented programs in which the data and methods are tightly encapsulated enable and encourage modular design of software so that the modules or components can be reused. Therefore, object-oriented frameworks are generally accepted as significantly improving the extensibility and reusuability of software. Many codes often provide a set of user-defined interfaces, which can be used to extend the package capability to a *limited* set of new circumstances. In general, non-object-oriented programs, even when user-defined routines are provided as means for extension, may suffer from a lack of clarity in its data structure or perhaps even poor program design. Achieving extensibility with these tools is a difficult and time-consuming exercise. Since early 1990s, the development of object-oriented finite element frameworks has received attention in the literature; we cite [13–15] as examples of this research. These frameworks are focused on applying object-oriented programming features to the finite element method to achieve flexibility, extensibility and the maintainability at the cost of a relatively small loss in efficiency. Similar meshless frameworks, particularly those that address the added challenge of extensibility to model multiple physical behavior, and shape/topology optimization, do not appear to exist at the present time.

We describe in this paper a framework termed jNURBS, which is a flexible, extensible, design-analysis integrated meshless framework for shape and topology optimal design. jNURBS has the following features:

1. *Meshless*. A NURBS representation is used to define both the geometric entities as well as the analysis fields, which leads to a meshless integration of design and anlysis.
2. *Extensible*. Java language and Object-Oriented Programming (OOP) techniques were used in building the framework. Highly extensible geometry/field descriptions and a flexible optimization problem definition are included in the program kernel.
3. *Symbolic User Interface*. A high level language, JNS (jNURBS Script), was developed to provide an interface to jNURBS. The scripting language allows the symbolic definition and solution of new problems in a natural manner.

NURBS are a very general mathematical representation for parametric curves and surfaces; it is currently the industry standard for the representation and design of geometrical surfaces. The advantages of the NURBS representation include one common mathematical form for both standard analytical shapes (e.g. conics) and free form shapes that can be evaluated fast by numerically stable and accurate algorithms. Due to the efficiency of its polynomial basis, NURBS can represent complex geometries with fewer parameters [16]. Thus, the use of NURBS is expected to result in fewer degrees of freedom in jNURBS during both the analysis and the design phases. Since the analysis field representation is related to the geometry representation and does not require a meshing step, the method is naturally meshless.

In the following sections, the architecture and the underlying methodology of jNURBS are presented in detail. Example problems selected from meshless structural analysis, material microstructure simulation, shape optimal design, and fluid static problems concerning the equilibrium shapes of droplets are solved to demonstrate the versatility of the framework.

2. Overview of the architecture of jNURBS

In jNURBS, NURBS serves as both the geometry modeling tool during the design phase and the field modeling tool during the analysis phase. Therefore, the data structures and algorithms for manipulating NURBS entities form the kernel of the code, and the core of the geometry/field modeling framework (Section 3.1). In addition, the kernel includes an optimization framework. A non-linear optimization procedure using a Sequential Quadratic Programming (SQP) code, NLPQL [17], is used to carry out the numerical optimization.

jNURBS was progammed using the JAVA language [18,19]. The choice of the programming language was influenced by a desire for platform independence and by the promise of error free programming that the language constructs provide. Currently, the framework complies with JDK 1.4.2 and has been tested under Windows 2000/XP, SunOS 5.8, and SuSE Linux 8 environments. In the framework, a high level script language termed JNS (jNURBS Script) serves as the user interface. JNS has many of the common features of a general programming language including looping and logical control. More importantly, JNS uses an easy to understand syntax akin to a natural language description for entity definition and problem description (Section 4). The user can also extend JNS to new applications via a set of interfaces.

The architecture of the framework is illustrated in Fig. 1. The developed package contains more than 35,000 lines of Java code and implements over 200 classes/interfaces. Package *jNURBS.utils* is a collection of functions that include matrix operations, linear equation system solver, numerical quadrature algorithm and optimization algorithm. System constants related to the above operations such as the numerical value below which a number is to be treated as zero are also defined in this package. Packages *jNURBS.kernel* and *jNURBS.kernel.field*, which implement the geometry/field modeling framework and the optimization framework, together comprise the kernel of jNURBS. Package *jNURBS.Material* provides material model description. Package *jNURBS.Parser*, which is a symbolic parser/interpreter of JNS, implements the user interface to the framework. All application packages are constructed as extension packages with names *jNURBS.app.**. New application packages can be added to the framework as extension packages without modification to other components of the framework.

The kernel and the user interface are described in detail in Sections 3 and 4. Several applications of the packages are presented in Section 6.