Data structures supporting multi-region adaptive Isogeometric Analysis

Anna Perduta[1] **and Roman Putanowicz**[1]*
[1] *Institute for Computational Civil Engineering, Cracow University of Technology*
Warszawska 24 street, 31-155 Cracow, Poland
e-mail: Anna.Perduta@L5.pk.edu.pl

Abstract

Since the first paper published in 2005 Isogeometric Analysis has gained strong interest and found applications in many engineering problems. Despite advancement of the method there are still far fewer software implementations comparing to Finite Element Method. The paper presents an approach in development of data structures that allow for multi-region IGA with local mesh refinement (patch-based) and possible application in IGA-FEM models. The purpose of this paper is to share design concepts, that authors have created while developing an IGA package, which other researchers may find beneficial for their own simulation codes.

Keywords: isogeometric analysis, NURBS, topology, data structures

1. Introduction

Isogeometric Analysis[1] is a computational method that utilises Non-Uniform Rational B-Spline family of functions to describe shapes, both for CAD modelling as well as for the subsequent PDE analysis. The use of a common geometry representation aims at addressing well known problems of translating geometric models between subsequent steps of analysis and problems of automatic reliable mesh generation, especially with hexahedral elements. Tighter link between analysis kernel and CAD preprocessor allows for easier handling of problems for which geometry undergoes severe deformations or unknown boundary problems.

The use of NURBS as approximation base opens new possibilities for adaptive approaches, also serving well the popular hp-adaptive methods.

From mesh generation point of view the key feature of IGA is the provision of domain parametrisation, i.e. provision of curvilinear coordinate system that is boundary fitted. Such parametrisation is attainable even for complicated shapes or shapes with boundaries having corners. However single global parameterisation of the underlying domain is not feasible in general case. Thus the need to utilise multiple NURBS patches or NURBS solids. The multi-region modelling is also motivated by the requirements of local mesh refinement, complex topology, i.e. multiply-connected domains, or the ease of application of material properties, loads or fixtures.

At this point appears the problem central to our presentation – namely the fact that implementation of multi-region IGA method is much more involved on the design and implementation level than a single-region IGA. Similar sort of problems appear when attempting to link IGA with other computational methods.

By now IGA is relatively well described method, especially in the context of single-region NURBS parameterisation. One can even find relatively simple Matlab implementations, that can be treated as the starting point or a reference.

However the description of multi-region IGA, especially from software engineering point of view, as opposed to computational method description, are relatively rare or incomplete. If we compare this with high cost of the development of simulation codes, it yields the problem many researchers have to tackle. Of course one can rely on some ready tools and closed "black box" solutions, but to be on the "bleeding edge" of development of new methods a team should work out its own software in order to know it inside-out and to shape it according to its goals.

2. Topology versus Geometry

One of the key issues while designing our IGA kernel was the decision how the geometric model should be stored. We didn't want to locate our effort in development of a CAD engine nor had resources to buy one. The same time we wanted to ensure that we can handle fairly complex geometries in some future. Initial attempts to build the kernel highlighted clearly that we can build it by linking out-of-shelve components for handling topological data on one hand and geometric data on the other. By topological data we mean the information how the geometric domain is split into subdomains regardless of their shape. The geometric data in turn describe shape of edges, patches and volumes. This fairly trivial decomposition has important consequences for design and implementation, making our computational kernel very modular and amenable to refactoring.

To handle topological data we decided to use very comprehensive MOAB framework[2]. Mesh Oriented dAta Base is C++ framework being a part of SIGMA project. MOAB allows us to link any type of data to mesh entities and to build arbitrary adjacency relations between the entities. Based on MOAB we have introduced the concept of MetaMesh that has been described in [3].

Our initial design assumed that detailed geometric data about NURBS curves, surfaces, or solids, will be stored in somehow raw form (not encapsulated) on mesh entities of the MetaMesh. This has worked well when subdomains with different density of NURBS control meshes were linked by a sort of static condensation of dofs on the subdomain interfaces.

However attempts to implement other approaches to linking subdomains, for instance based on Nitsche method[4], and attempts to implement FEM and IGA blending, have shown that one needs much more abstract treatment of geometric data, that will not expose the "NURBS machinery" to the underlying topological framework. In other words instead of storing control nets, knots, weights, even packed in some object, one needs to intro-

* This work was financially supported by Polish Ministry of Science and Higher Education within the contract no. L5/130/2017/DS.

duce the abstract concept of geometric mapping from reference to physical domain. We encapsulate this concept within hierarchy of classes, with GeomMapper being the top level abstract class. The GeomMapper provides API to map parametric coordinates to real ones, calculate Jacobian of this mapping, etc. In case of IGA the reference domain is an n-dimensional unit hypercube, that is segment, square or cube, respectively. For such domains the mapping is constructed as a tensor product of 1D NURBS functions. In case of FEM the reference domains are usual FEM elements.

3. Main classes for extensible framework

In order to be able to fit IGA and FEM into one consistent framework, we have introduced in our design the following classes: MetaMesh, BasisFunctions, Discretisator and GeomMapper. Figure 1 shows relations between them.

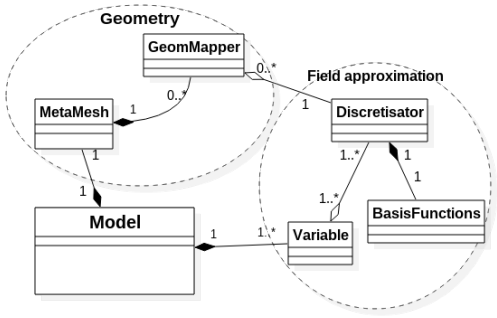

Figure 1: Relations between base data structures

The **MetatMesh** class is responsible for storing topology data and is implemented on the basis of moab::Core class from MOAB framework. **BasisFunctions** is an abstract base class for describing collocations of function and their derivatives in the parametric space. Specific concrete classes derived from it are NURBSFunctions and LagrangeBasisFunctions. **Discretisator** is an abstract base class that introduces the notion of degree of freedom. Because of our assumption of targeting various computer methods, the degrees of freedom are understood in mathematical sense and not necessarily tied to specific mesh entities. Discretisator also brings the notion of an element as a portion of parametric domain over which interpolation or integration operations are performed. In case of NURBS based discretisators the elements form structured grid and are stored implicitly via knot vectors, while in case of other discretisators the elements form unstructured grid and are stores explicitly. Finally **GeomMapper** is an abstract base class that provides interface for describing region shape. It uses Discretisator and specific geometric data, such as control point grid in case of NURBS based Discretisator. Discretisators are used in our framework to implement both mappings from parametric to physical domains and to provide approximation of the unknown fields. Because the selection of Discretisators for geometric mappers and unknown fields is independent we can easily handle various methods. For instance in case of IGA the discretisators for geometric maps and unknown fields are both based on NURBS.

4. Conclusions

The above in necessarily very succinct description, not reflecting full intricacies of the design decisions that have to be made. For instance one has to endow these topological-geometric entities with data and methods to support assembly of algebraic system. Careful considerations are necessary for the issues of identification end enumeration of degrees of freedom, their influence domains or integration domains. It is especially true in the presence of interfaces between locally refined regions, or when using domain decomposition approaches.

The lesson we have learned from implementing above described concepts is that once the code is ready, even if the API is properly documented (what in academic code is rather exception than rule), then the design rationales are somehow lost and it is hard to decipher them from the code itself. This forces researchers to reinvent the wheels or faces them with steep learning curves for third party code. The transparency of the simulation codes is the issue that recently has been widely raised, and we hope that sharing our experience is the step in the right direction.

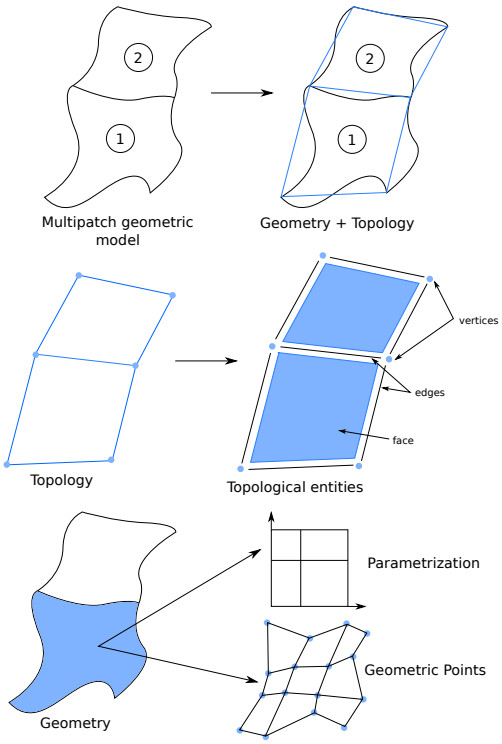

Figure 2: Decomposition of multi-region model into topology and geometry description

References

[1] Hughes, T.J.R. and Cottrell, J.A. and Bazilevs, Y., Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, *Computer methods in applied mechanics and engineering*, Vol. 194, 39, pp. 4135–4195, 2005.

[2] Tautges, T. J. and Meyers, R. and Merkley, K. and Stimpson, C. and Ernst, C., MOAB: A Mesh-Oriented Database, *SAND2004-1592* , Sandia National Laboratories, 2004.

[3] Perduta, A. and Putanowicz, R., Data structure for supporting patch refinement in adaptive isogeometric analysis, *CSE-EUC-DCABES 2016*, pp. 673-676, Paris, France, 2016.

[4] Nguyen, V.P. and Kerfriden, P. and Brino, M. and Bordas, S.P.A. and Bonisoli, E., Nitsche's method for two and three dimensional NURBS patch coupling, *Computational Mechanics*, Vol. 53, 6, pp. 1163–1182, 2014.