Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on Graphics Interface

Atlas of connectivity maps

Ali Mahdavi-Amiri*, Faramarz Samavati

Department of Computer Science, University of Calgary, 2500 University Drive N.W., Calgary, Alberta, Canada T2N 1N4

ARTICLE INFO

Article history:
Received 24 June 2013
Received in revised form
26 August 2013
Accepted 14 September 2013
Available online 28 September 2013

Keywords:
Data structure
Semiregular
Connectivity map
Multiresolution
Subdivision

ABSTRACT

Semiregular models are now ubiquitous in computer graphics. These models are constructed by refining a model with an arbitrary initial connectivity. Due to the regularity enforced by the refinement, the vertices of semiregular models are mostly regular. To benefit from this regularity, it is desirable to have a data structure specifically designed for such models. We discuss how to design such a data structure, which we call the atlas of connectivity maps (ACM) for semiregular models. In an ACM, semiregular models are divided into regular patches. The connectivity between patches is captured at the coarsest resolution. In this paper, we discuss how to find these patches in a given semiregular model and how to set up the ACM. We also show some of the benefits of this data structure in applications such as the multiresolution framework. ACM can support a variety of different multiresolution frameworks including compact and smooth reverse subdivision methods. The efficiency of ACM is also compared with a standard implementation of half-edge.

1. Introduction

Semiregular models are very common in computer graphics [1]. These models are obtained by applying repetitive refinement on an arbitrary initial mesh or they may be constructed by a parametrization method (Fig. 1). Applying refinement on a mesh produces a large number of vertices. However, these vertices are mostly regular, with irregular vertices corresponding to extraordinary vertices of the initial unrefined model. This regular structure should be taken into consideration in order to efficiently capture the connectivity information of the model. Additionally, the geometry of the vertices (coming from sources such as subdivision schemes, projections, parametrization) should also be recorded.

In [2], we introduce ACM: Atlas of Connectivity Maps to efficiently capture the connectivity between the regular patches of a semiregular model. These patches, which can be obtained from an arbitrary refinement, are mapped onto a set of quadrilateral 2D domains. The connections between vertices and faces are captured by these 2D domains (connectivity maps) and their interconnections. The ACM can be used as an efficient data structure for semiregular meshes to handle connectivity queries.

In ACM, a coordinate system is assigned to each connectivity map such that each vertex has integer coordinates. These integer coordinates are used to index the faces and vertices and handle neighborhood queries. A hierarchical relationship exists between connectivity of vertices and faces at various resolutions. To establish this hierarchical relationship, we apply rotation, translation, and/or scaling to transform the coordinate system of one resolution to another. The vertices' 3D coordinates are stored in 2D arrays associated with each connectivity map and indexed by the vertices' connectivity map coordinates.

We also categorize regular refinements for quad meshes, and for each category, we propose methods to handle adjacency and hierarchical queries using our data structure. We then describe how to support triangle meshes and discuss applications such as subdivision and multiresolution.

In this paper, we extend our previous work [2] in several different ways. In [2], we describe how to set up an ACM for an initial coarse mesh with arbitrary connectivity. We then make the semiregular model by applying regular refinements on the initial coarse mesh. Here, we present how to set up an ACM for a given semiregular mesh and associate a connectivity map with each regular patch of the mesh. We also describe how to handle sharp features such as creases and corners.

One of the immediate applications of the ACM is the support of meshes resulting from different subdivision methods. In [2], we note that ACM is efficient both in terms of space and time at supporting connectivity queries of subdivision surfaces. A multiresolution framework, which allows one to transition between the high and low resolution versions of a model without losing details, can be developed by pairing subdivision with its reverse subdivision scheme. The ACM can also be efficiently used to support multiresolution frameworks.

In [2], we describe how to support Catmull–Clark, Loop, and $\sqrt{3}$ reverse subdivision. Here, we extend it to support $\sqrt{2}$ reverse

* Corresponding author. Tel.: +1 403 890 7560; fax: +1 403 284 4707.
E-mail address: amahdavi@ucalgary.ca (A. Mahdavi-Amiri).

0097-8493/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cag.2013.09.003

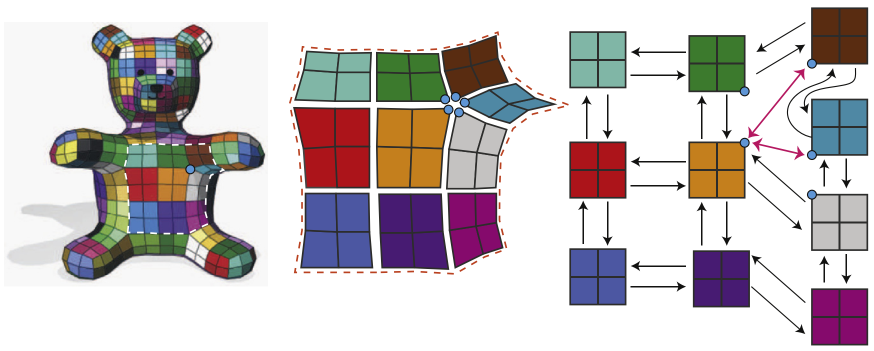

Fig. 1. A semiregular mesh with mostly regular vertices.

subdivision. In addition, supporting the recently developed "smooth reverse subdivision" multiresolution framework is also presented. We also provide the filters for $\sqrt{2}$ reverse and $\sqrt{3}$ reverse and smooth reverse subdivision.

We have compared the time and space efficiency of our data structure with alternatives in [2]. The half-edge data structure used for our comparison in [2] was implemented by ourselves and was not based on a standard implementation. Here, we report the speed of the half-edge data structure implemented in CGAL to make a comparison with a standard implementation of the half-edge data structure [3].

We organize the paper as follows: in Section 2 some related work is presented. We provide an overview and a detailed description of the ACM in Section 3. A method is provided in Section 4 to adapt the ACM to a given high resolution semiregular model. A representation for sharp features in the ACM is described in Section 5. In Section 6, an efficient technique for multiresolution representations for Catmull–Clark, Loop, $\sqrt{2}$, and $\sqrt{3}$ is proposed. We also describe how to handle smooth reverse subdivision using the ACM. We compare our work with CGAL as an efficient implementation of half-edge in Section 7. Future work and limitations are presented in Section 8 and we conclude in Section 9. We also provide the filters of $\sqrt{2}$ and $\sqrt{3}$ compact reverse subdivision in Appendix A and Appendix B respectively and the filters of smooth reverse $\sqrt{2}$ and $\sqrt{3}$ subdivision in Appendix C.

2. Related work

Data structures for semiregular models can be found primarily in the literature related to subdivision and multiresolution. We present work related to our proposed method, divided into two categories: subdivision and multiresolution.

Subdivision. Subdivision is a well-studied subject in computer graphics. There are many subdivision schemes, such as Loop, Catmull–Clark, Doo–Sabin, $\sqrt{2}$ and $\sqrt{3}$ subdivision [4–8]. Subdivision is typically a two-step process: one step of refinement followed by an averaging step. The relationship between lattices at different resolutions resulting from different types of refinement has been previously classified in [9,10]. Our categorization of refinements is similar to their work. However, we have classified subdivision to assist in designing an efficient data structure to address connectivity queries on an arbitrary connectivity model.

The half-edge data structure and its variations are commonly used to model subdivision surfaces [11]. These data structures are designed for general topological objects' adjacency queries. However, the half-edge data structure cannot be directly used for hierarchical access. Furthermore, it does not benefit from the regularity of subdivision and, therefore, for objects with a large number of vertices it becomes inefficient.

An alternative data structure that supports hierarchical operations is the quadtree [12]. Quadtrees are commonly used for hierarchical meshes, particularly for hierarchical editing applications [13]. Although quadtrees are quite effective at supporting hierarchy between resolutions, they need to store many pointers to maintain their nodes' conductivities and hierarchical dependencies. To overcome this inefficiency, indexing methods exist which assign a unique index to every node and discard the tree structure [12]. However, these indexing methods are primarily designed to support hierarchy and ignore adjacency relationships. Moreover, since quadtrees are designed to support 1-to-4 refinement, they cannot be directly used to support other refinements.

Patch-based refinement methods rely on data structures that are specifically designed for subdivision methods [14–18]. Here, meshes are divided into patches and subdivision is separately applied to each patch. Each patch is stored in an array and the connectivity between the patches' boundaries is handled using repetitive points at the boundary edges or a first resolution edge based data structure. These methods are mostly designed for a specific type of refinement or primitive shapes [15,18]. Some of these data structures use spiral 1D indexing for vertices [16,17]. Spiral indexing complicates neighborhood access, especially for non-immediate neighbors that are essential for applications like multiresolution. We instead use simple 2D domains to maintain connectivity information and extend patch-based methods to support all types of refinement.

Multiresolution. While subdivision generates high resolution objects, multiresolution provides a means to transition from high to low resolution and vice versa [19]. Some multiresolution frameworks, though not all, maintain the semiregularity of objects. This can be achieved by reversing the process of subdivision (i.e. via a reverse subdivision process) [20–22] or by considering a property of the coarse vertices, such as smoothness (computed via the Laplacian) [13]. Since both the Laplacian and reverse subdivision use local operators to coarsely sample the fine model, our proposed method can handle these operations.

Olsen et al. [20,21] provide a compact multiresolution framework using the concept of even/odd vertices. At different resolutions, the even/odd labeling distinguishes multiresolution details from coarse vertices. They use an edge based data structure to handle connectivity queries and a hashing method to map vertices to details or coarse vertices [23]. To show that our ACM can efficiently support multiresolution frameworks, we describe how to support the compact multiresolution proposed in [20,21] and compare the speed of our data structure with [23].

To adapt the half-edge structure to multiresolution frameworks, Kraemer et al. [24] modify this data structure by defining sequences of half-edges. Using this *multiresolution half-edge* structure, it is possible to support primal and dual schemes. However, this data structure requires a large amount of memory for high resolution models due to the storing of all edges and an extensive