A Sketch-Based Modeling Framework Based on Adaptive Meshes

Emilio Vital Brazil, Ronan Amorim, Mario Costa Sousa
Department of Computer Science
University Of Calgary
Canada

Luiz Velho, Luiz Henrique de Figueiredo
IMPA
Instituto Nacional de Matemática Pura e Aplicada
Brazil

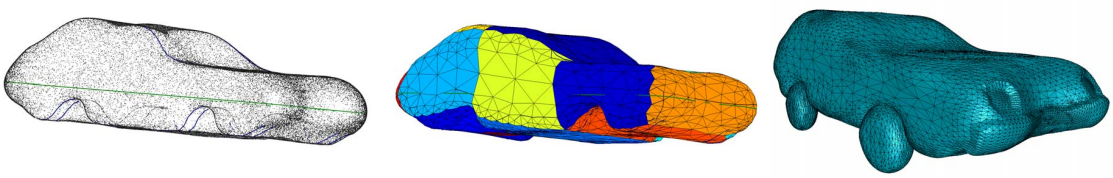

Fig. 1. Modeling a race car. Left to right: implicit model (HRBF), adaptive mesh, and final model augmented.

***Abstract*—In the last 15 years many systems for sketch-based modeling have been developed. Much of this work has focused on the final results and describes the solutions from a technical and practical point of view. In this paper we take a more theoretical approach to the problem of sketch-based surface modeling (SBSM) and introduce a framework for SBSM systems based on adaptive meshes. The main advantage of this approach is to split the modeling operators and the final representation, allowing the creation of SBSM systems suitable for specific domains with different demands. In addition, we present two systems built on top of this framework, one with the capability to control local and global changes to the model and one that follows domain constraints.**

***Keywords*-Sketch-Based Modeling; Adaptive Mesh; Geometric Modeling**

I. Introduction

Sketch-based modeling (SBM) is a well established research area encompassing work in different domains such as computer vision, human-computer interaction and artificial intelligence [1]. In the last fifteen years there has been an extensive body of work [2], [3], [4], [5], [6]. However, these systems are more concerned about the final results and do not discuss the theoretical aspects of the problem of how to build a sketch-based system. In contrast, we are proposing a framework tailored for sketch-based surface modeling (SBSM) that takes advantage of adaptive meshes.

We advocate that SBSM systems must be suited for each specific application: the specificities of a certain field require suitable mathematical representations for the domain model, which plays a central role in the characterization of SBSM applications. However, there are common requirements in many SBSM applications that can be abstracted to guide the definition of specific representations for specific domains. These requirements have the following three main aspects: (1) Dynamic; the surface will change during the process. (2) Interactive; the users must be able to see the model changes in interactive time. (3) Controlled freedom; some applications have specific rules of modeling, the systems must be able to incorporate these rules to guide the modeler, but without losing flexibility.

In general, adaptive meshes are associated with the ability to produce complex models using a smaller mesh. However, our proposed framework is based on adaptive meshes because they can be dynamic, allow rapid updates, and local control. Different schemes of adaptive meshes can be used to create a system using our framework, indeed the choice of the scheme must take into account the final application demands, e.g., represent features, change topology, smoothness.

With the purpose of studying and testing our framework, we developed a sketch-based system that approaches a common problem in many SBSM systems, that is, the lack of good control of global and local transformations. In Fig. 1, the race car is augmented with local deformations but with no changes to the overall shape. In order to create a real system we must create mathematical tools and simple computational solutions, e.g., for this system we developed a differential atlas for the sketched surface based on a simple label scheme which has theoretical guarantees. In addition, we describe how we also applied this framework to model geological layers using some rules to guide the expert in the modeling process.

II. Related Work

Constructive SBM systems directly map a set of 2D sketched input strokes to a 3D model without any previous knowledge about the model's geometry or topology [1]. This class of systems can be categorized by the two fundamental types of geometry being reconstructed: *linear* (i.e., lines, planes and polyhedra) or *free-form*. Linear SBM systems are typically oriented towards CAD and architecture applications;

1530-1834/14 $31.00 © 2014 IEEE
DOI 10.1109/SIBGRAPI.2014.16

two notable works that can be categorized as linear are Zeleznik et al. [7] and Jorge et al. [8]. Free-form SBM systems, on the other hand, are used in applications requiring modeling of more organic, natural structures; we can cite the seminal work of Igarashi et al. [2] on the now classical *Teddy* system and Nealen et al. [3].

There are many ways to represent surfaces in $\mathbb{R}^3$, the most common and general are implicit and parametric representations. However, in order to be used in computer graphics applications, these representations must be more specific and possess practical qualities. As examples we can cite the BlobTree [9], piecewise algebraic surface patches [10], convolution surfaces [11], generalized cylinders, polygonal meshes, subdivision surfaces, among others.

In the following, we discuss the main works in free-form Sketch-Based Surface Modeling that start from scratch under the light of its representations.

Teddy [2], Fibermesh [3], and Kara and Shimada [12] use triangular mesh as a base representation for their modeling systems. Teddy and Fibermesh start with a planar curve and create an inflated mesh based on the curve geometry. The Teddy system extrusion and cutting modeling operators cut a mesh part then create a new mesh patch, which is merged with the model. Similarly, based on the input sketches Fibermesh creates a new mesh and place it using optimization on differential coordinates, allowing the system to keep all previous strokes as constrains. Kara and Shimada also keep a set of 3D curves to define the final model. However, they use curve loops to define triangle-mesh patches that have minimum curvature, instead of optimizing across the whole mesh. These patches can be modified using physically-based deformation tools. These three systems are based on the triangular mesh representation and use it to build their modeling operators; as result, their advantages and limitations are directly related with that representation.

Parametric surfaces are defined by mapping a planar domain in 3D space. To work with parametric surface has some advantages, it is simple to obtain a good triangle mesh that approximates the model, easily maps textures, and provides continuous normal and curvature information, among others. Cherlin et al. [13] and Gingold et al. [4] use this representation to create sketch-based systems. The first one introduces two novel parametric surfaces based sketched curves. The latter converts sketches to generalized cylinders. However, both of these have issues with topology change and creating augmentations, these difficulties are mainly caused by the chosen parametric representations. By the same token, Nasri et al. [14] and Orbay and Kara [6] create their systems based on subdivision surfaces. Height field is another example of parametric surface; it present a fast and simple mapping of a 3D position as a function of a given 2D coordinate (i.e. $(x, y, f(x, y))$). Such representation is usually enough for most terrain comprising mountains and hills. However, height fields are not able to represent terrains with more complex geologic structures such as those containing overhanging cliffs or caves. The reason for this limitation is because height fields can only have a single 3D position associated with each 2D coordinate and clearly overhanging cliffs and caves need more than one 3D position. Hnaidi et al. [15] present a sketch-based system to model terrains. The characteristics of the terrain are defined by the user through a set of feature curves that are able to define ridges, river beds, and cliffs. Constraints on these curves define elevation, angle and noise parameters along them. These constraints are then defined for the entire domain by diffusion. When the smooth terrain is ready, details are added by a procedural noise generator. The final terrain is a height field that results from combining the smooth terrain with the details.

In contrast to parametric surfaces, implicit surfaces can easily change the topology. They can also provide a compact, flexible, and mathematically precise representation which is well suited to describe coarse shapes. Implicit surfaces enable global calculations such as point pertinence (i.e., whether a point is within the surface volume) and distance evaluation, and at the same time, they allow the user to obtain local differential properties, such as normals and curvature. Karpenko et al. [16] introduces variational implicit surfaces as representation to sketch-based surface modeling. Later Vital Brazil et al. [5] improve this formulation adding the normals as hard constraints. Schmidt et al. [17] use BloobTrees as a main representation of the ShapeShop system. Bernhardt et al. [18] build the Matisse system based on convolution surfaces. These systems share the main disadvantages known to implicit representations. Which are: (1) the standard graphic pipeline is not prepared to visualize implicit models, (2) few industrial processes use implicit surfaces, so then the final model must be converted and, (3) it is hard to control details. For (1) and (2) almost all systems polygonize the models (e.g., marching cubes), but there are many drawbacks in this approach, e.g., some methods do not guarantee the topology nor the mesh quality.

On the whole, much of this previous work is built on a specific representation and its drawbacks come from that. Inspired by that we are proposing a simple framework based on adaptive mesh to allow us mix different representations in one system. This paper is divided as follows. We give a overview about Adaptive meshes (Sec. III) and then discuss our framework (Sec. IV). In Section V-A we present the Detail Aware Sketch-Based Surface Modeling (DASS) system. We discuss the Geological Layer Modeler (GLaM) system in Section V-B, and finally we conclude and discuss future work in Section VI.

III. Adaptive Mesh Overview

The adaptive mesh is a polygonal mesh that has the ability of creation/deletion of vertices, edges or faces following predefined rules. The creation process is called refinement and the deletion is called simplification. The adaptive mesh scheme starts with the base mesh which is refined until it matches a stop criterion. Usually this criterion is associated with a maximum error threshold. In summary, one adaptive mesh must have a base mesh, a refinement/simplification criterion,

and a refinement/simplification rule. Since we are working with a dynamic system we also need an update rule.

In order to illustrate these concepts we will use the well known 4-8 adaptive mesh [19]. Its refinement process creates a vertex on one edge (edge split) this results the creation of more two edges and two faces; and the simplification deletes one vertex (edge weld). Suppose you want use the 4-8 to approximate a surface S, which given a point you know the distance between the point and the surface; and you know how to project points on S. One simple refinement criterion can be the distance of the middle point of the edge to S; and simplification can be the average distance on the vertex star. And finally the simplification rule can be the projection on the actual surface.

IV. Framework

We conceived the framework to be able to build a sketch-based system that has the following 2 major qualities. (1) Interactivity: the system must be able to show the model changes in interactive time. (2) Controlled freedom: some applications have specific rules of modeling, the systems have to be able to incorporate these rules to guide the modeler, but without losing flexibility. In addition, the framework must be general to be applied in different domains with different requirements. To define the framework we compass these 3 goals and the minimum requirements to create and handle an adaptive mesh [20]. We split the framework in three main components: initial shape descriptor, adaptive mesh, and editing operators. Fig. 2 illustrates the main information flow between these components.

Fig. 2. The framework for Sketch-based surface systems. The arrows depict the information flow.

First of all, we need an initial shape descriptor to be able to tessellate the coarsest mesh, which is called base mesh. For example, it could be the first inflated model of the Teddy or Fibermesh. The base mesh must have the same topology of the intent model and approximate the geometry. Geometry approximation has different meanings depending on the application, as a general rule it means that when a new vertex is created it can be correctly placed. For instance, for implicit surface, the base mesh have to be inside of the tubular neighborhood of the surface.

In the proposed framework the main roles of the adaptive mesh is to allow independent geometry representations for the editing operators and to keep the coherence in the modeling process. A positive side effect of using adaptive meshes is to be able to use the base-mesh as a natural parametrization of the surface, as discussed in Section V-A1.

The editing operators are the system parts that are responsible for all model modifications, such that the edited mesh is still an adaptive mesh. Much of the work of editing adaptive meshes is done by changing the rules and criteria described in the former paragraph. For instance, if it is a geometric editing the operator can be implemented as a new rule for vertex update and refinement, after that the mesh will be adapted for the new shape. Since the obtained mesh is an adaptive mesh, the editing loop restarts.

V. Applying the Framework

In this section we present two systems built on top of this framework. The first one approaches a common problem in many SBSM systems that is the lack of good control of global and local transformations. The *Detail Aware Sketch-Based Surface Modeling* (DASS, Section V-A) was created to allow us to validate our proposed framework, exploring the limitations of a system without a well defined task. The second system presented is the *Geological Layer Modeler (GLaM, Section V-B)*, which is a sketch-based system specialized for geology, it aims to help geophysicists to create subsurface models. It is a good illustration of controlled freedom, where the sketch-operators should be restricted to follow geological rules.

A. Detail Aware Sketch-Based Surface Modeling (DASS)

In the DASS system, we use a composite surface representation separated into levels of detail and properties. Blinn [21] introduces the idea of bump-mapping that stores geometric information at two levels, the base geometry and a displacement map which is used to create rendering effects; the same concept is found in [22] and [23]. They use two different types of data, with the first one defining the smooth geometry and the second one mapping the first in a parametric space that stores details (similar to a texture mapping).

It is important to remark the difference between our proposal system solutions and multi-resolution works [24] and manifold surface modeling [25]: multi-resolution works are concerned about *subdivision scheme* and we do not use subdivision nor multi-scale analysis. Instead, we use a 4-8 mesh, an adaptive mesh which nonetheless can simulate many subdivision schemes [26] (although we do not use it as such). Also, manifold modeling community approaches the problem of how to build and edit manifold structures starting from a mesh or a subdivision scheme. In contrast, we use the base mesh directly to construct such structure, and we developed simple rules to ensure correctness of the manifold structure when we apply editing operators.

In summary, the main goal of our prototype is to be able to control local editing without changing parts of the model out of the region of interest, and keeping details coherently when big deformations are introduced. Hence, we advocate that decomposing the model representation into a base surface that supports different types of properties is a powerful tool for sketch-based surface modeling.

Our framework starts with a coarse shape represented by an implicit surface; specifically, we use Hermite Radial Basis Function (HRBF) due to its supports for great variety of SBM operators, and also its good projection properties [5]. After

that, we create a method to construct a manifold structure for the implicit surface, which allows us to handle different parameters through the models, such as local augmentation, level of detail, color, among others. We use the 4-8 adaptive mesh to obtain good frequency control, and maintain coherence between global and local transformations.

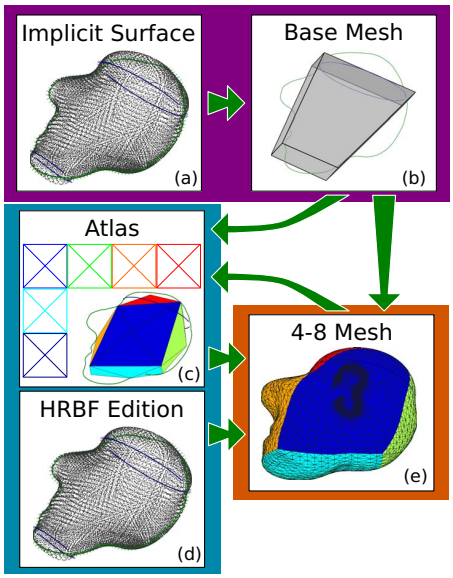

Fig. 3. The framework of DASS system. The color boxes are related with the theoretical framework in Fig. 2.

1) Adapted Framework: We start with the coarse form defined by an implicit surface; after that we build a base mesh that has the same topology and approximately the same geometry of the implicit surface. The base mesh induces an atlas and provides a 4-8 base mesh. The atlas is built using a partition of the set of faces of the mesh, and we use it to edit the model locally. The 4-8 mesh has two roles in the framework: to build a map between surface and atlas, and to visualize the final surface. After we have all parts, the 4-8 mesh is used to edit details that are saved in the atlas, and the atlas maps details onto the 4-8 mesh. In Fig. 3 we depict our framework.

The first step in the framework is to obtain a coarse shape of the final model (Fig. 3(a)(b)). We use the same implementation described in [5], in which the authors introduce a new representation for implicit surfaces and show how it can be used to support a collection of free-form modeling operations. This implicit representation, the variational Hermite Radial Basis Function (HRBF), fits well with our framework due to its good projection properties, as well as for its simplicity and compactness.

After we obtain our implicit surface $\mathcal{S}$, we create the manifold structure to represent our final model S. To handle parameters, we use an atlas $\mathcal{A}$ of S, i.e., $\mathcal{A} = \{\Omega_i, \phi_i\}_{i=0}^{k}$ such that $\Omega_i \subset \mathbb{R}^2$, and $\phi_i : \Omega_i \to S$ are homeomorphisms [27]. However, we have an implicit surface without information about the atlas. One possible way to tackle this problem could

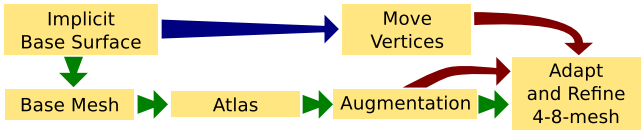

Fig. 4. Overview of DASS system work-flows: green arrows are the startup and topological change step sequence, blue arrow are stepped when the implicit surface is edited, and the red arrow is done when the mesh resolution changes.

be to create a polygon mesh and use one method to obtain a quad mesh [28]. However obtain a good mesh from an implicit function in interactive time is not a trivial task. There are many approaches to polygonize implicit surfaces, e.g. [29], [30], [31], but in order to find the correct topology of the model these approaches depend on user-specified parameters [29], [30], or require differential properties of the surface [31]. Apart from the topology issue, such methods neither guarantee the mesh quality nor have a direct way to build an atlas structure. As a result, we opted to develop a method that is based on our problem and desired surface characteristics.

First of all, we observe that there are two different scales of detail to be represented: the implicit surface (which is coarse) and the details (which are finer). The naive approach would be to use the finest scale of detail to define the mesh resolution. However, there are two issues associated with it: firstly, we do not know the finest scale a priori; and secondly, if the details appear in a small area of the model, memory and processing time will be wasted with a heavily refined mesh. To avoid the issues described in the former paragraph, we adopted a dynamic adaptive mesh, the semi-regular 4-8 mesh [20]: it allows for the control of where the mesh is to be fine and coarse, by using a simple error function.

Returning to the problem of parametrization of our implicit surface, now we wish for more than just a mesh: we need an adaptive mesh. The framework presented by [19] starts with a semi-regular 4-8 mesh and refines it to approximate surfaces using simple projection and error functions – from now on we say 4-8 mesh in place of semi-regular 4-8 mesh. To obtain a good approximation of the final surface, the 4-8-base-mesh must have the same topology and must approximate the geometry of the final surface. Thereupon our parametrization problem was reduced to the problems of how to find a good 4-8 base mesh and how to construct a good error function.

The parametrization of the implicit surface is built in three parts: base mesh (Fig. 3(b)), atlas (Fig. 3(c)), and semi-regular 4-8 mesh (Fig. 3(e)). Since the main focus of this work is not the implementation of this specific system, we leave the details for a technical report. In this technical report we present a base mesh with two roles in our system, inducing an atlas for the surface and creating a 4-8 mesh, describe a method to create an atlas for adaptive meshes based on stellar operators, and also discuss how build an error function for the 4-8 mesh that is sensitive to levels of detail. All technical details and mathematical proofs of this system are described [32].

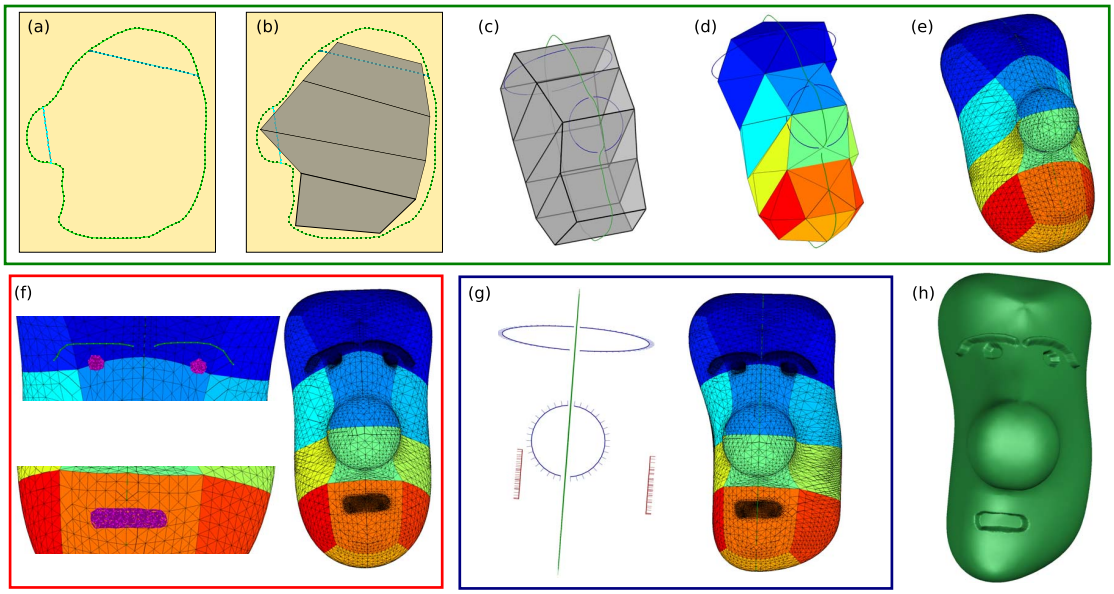

Fig. 5. Steps to model a head using DASS.

2) Work-flow and Results: In this section we present all pieces of DASS system working together. Our work-flows are based on the framework presented by [19] to adapt dynamic meshes. There are three different work-flows in this system: (1) the user starts the modeling system with a blank page, or by adding changes to the actual model topology, (2) the geometry of the implicit surface changes, and (3) the mesh resolution is recalculated (which usually happens when the height-maps are changed). The overview of the work-flow is depicted in Fig. 4.

The user starts the model with construction lines, creating samples that define an implicit surface (Fig. 5(a)) using the system described in [5]. After that, the user creates a planar version of the base mesh that approximates the geometry and has the same topology of the final model (Fig. 5(b)). Thus, the base mesh is transported to space (Fig. 5(c)). Now the base mesh is used to create an atlas structure (Fig. 5(d)) for a 4-8 mesh. This mesh is adapted and refined creating the first approximation of the final model (Fig. 5(e)). The steps described up to now are the common steps for all modeling sessions. They are represented by the green arrows in Fig. 4. In addition, these steps are illustrated in Fig. 6(a) and (b), and 7(a). Note that when we change the topology we also need to change the base mesh, restarting the process, e.g., in Fig. 6(a) and (b). If there is a predefined height-map, the model reaches the end of this stage with one or more layers of detail.

After the first approximation for the final surface, the user can edit the implicit surface and create/edit a height-map. When details are added on the surface, in almost all cases it implies that the resolution of the mesh is not fine enough to represent the new augmentation. In this case we must adapt and refine the mesh. In Fig. 5(f), 6(c), and 7(b): the user sketches a height-map over the surface and the mesh is refined to represent the geometry of the augmentation correctly. The user can change the implicit surface at any stage, and if the topology is still the same, then the system allows vertices to be moved without adaptation and refinement (in order to obtain a fast approximation). Since details are codified separately, they are moved consistently when implicit surfaces are edited. This is illustrated in Fig. 5(g), and 7(c), (e) and (f). Specifically in Fig. 7(e) and (f) we can compare good final results preserving the details despite the significant changes of the implicit surface. Sometimes, when only the implicit surface is changed, moving the vertices alone is not enough to reach the desired quality. In such cases, the user can adapt and refine the mesh decreasing the error threshold, as shown in Fig. 7(d). Here, the user initializes $\varepsilon = 10^{-3}$, and after some modeling steps a new threshold of 10^{-4} is chosen.

The modeling of each of the three models presented in this section took approximately 10 minutes, from the blank page stage up to the final mesh generation. All the results were generated on an 2.66 GHz Intel Xeon W3520, 12 gigabyte of RAM and OpenGL/nVIDIA GForce GTX 470 graphics. The most expensive step was creating the implicit surface, followed by the creation of the base mesh; on the other hand, processing of the augmentation and minor adjustments in the implicit surface had a minor impact on performance. The bottle neck is the mesh update; if the mesh has too many vertices (around 10k), one refinement step after an augmentation takes about 10 seconds. The final models of space car, terrain, head, and party balloon have 10k, 11k, 11k and 13k vertices respectively.

B. Geological Layer Modeler (GLaM)

Based on the framework presented in Section IV we developed a set of sketch-based interface and modeling operators

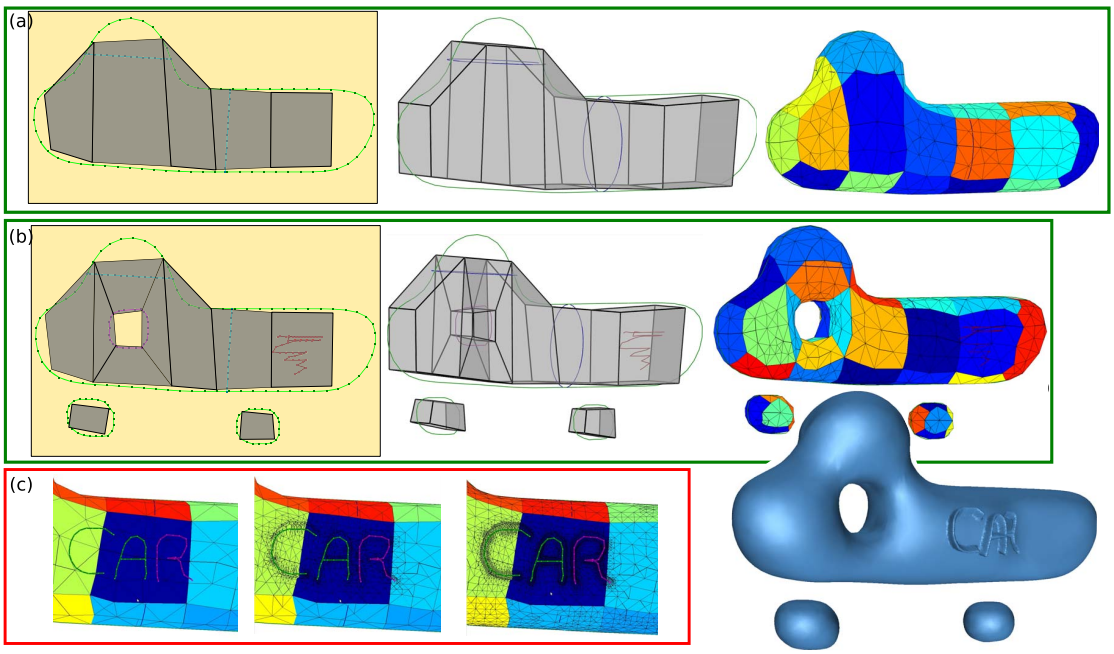

Fig. 6. Steps to model a space car using DASS.

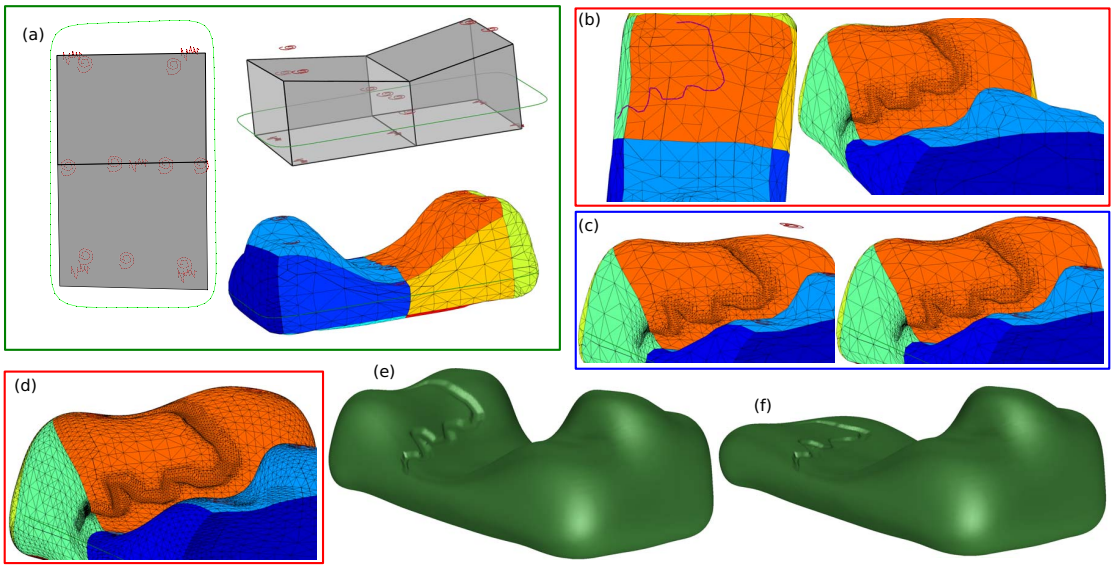

Fig. 7. Steps to model a terrain using DASS.

integrated in a system for the tasks of seismic interpretation and reservoir model building (Fig. 8). The GLaM system allows the user to sketch directly over the raw seismic reflection volume and its derived data. These data guide the expert in key tasks of seismic interpretation and building the structural framework of the reservoir. We propose a novel set of sketch-based modeling operators designed to meet the experts needs (geophysics and geology). Each operator has its own types of interaction, including drawing free-form strokes on the actual horizon surface and/or a seismic volume surface. The use of the framework was a key factor to incorporate all different operators required by the experts. All technical detail of this system are described in [33].

In contrast to DASS system, the GLaM initial shape descriptor is very simple because we are modeling horizons, so then the base mesh is a plane. Another important difference between the systems is GLaM is operator-driven, i.e., the system main features are defined by operators that change the mesh properties and these operators can be combined to create more complex ones. For example, to move one horizon up and

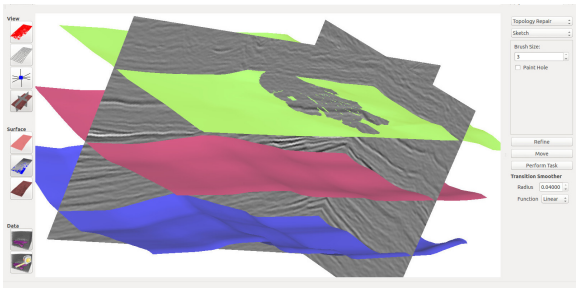

Fig. 8. GLaM system interface.

down is a operator that update the vertex positions and when this operator is combined with the sketch operator it starts to be a local operator. Since each operator input and output is a 4-8 mesh we create the editing loop as described in Section IV.

The operators in GLaM perform changes in the surface implementing adaptation to manipulate the 4-8 meshes. They have many inputs besides the mesh, e.g., input from mouse and keyboard in a filtered way with information of which surface and face (triangle) have been clicked. We were able to implement the operators as a completely separate module, making it easier to implement different operations. In addition, the composition of operators allows us to start from basic operators while keeping the whole system manageable and simple. Since the main purpose of this paper is to discuss the proposed framework we will not give many details about each implemented operator. Following we overview the main operators of GLaM prototype to illustrate the versatility of the proposed framework.

- *Topology Repair Operator* allows the user to create or delete holes on the horizons by texture manipulation. This operator is a good example of combination of simple operators, the first allows the user edit *hole texture* using brushes like an image, after s/he is satisfied with the result other two operators are used, one to refine the mesh around the holes and other to remove the vertex creating the final mesh with the desired topology (Fig. 9).

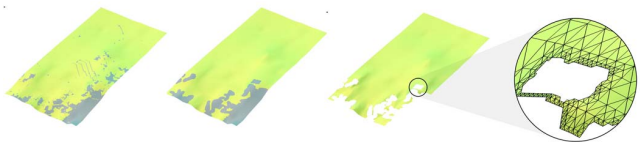

Fig. 9. Topology repair operator. Left to right: original mesh, after *hole texture* edition, and final mesh.

- *Feature Augmentation and Horizon Fault Deformation Operators* allow the user to create deformations using a set of sketches curves. These operators deform only the selected area using a parametric representation based on the distance to strokes to create final effects. The main differences between them are the meaning of the lines and the change of the mesh topology performed by the Horizon Fault operator (Fig. 10).

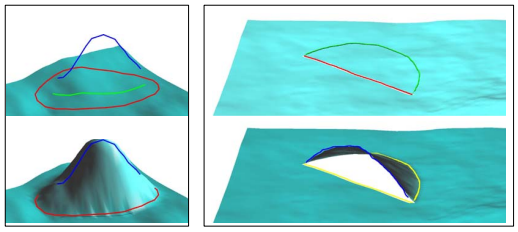

Fig. 10. Left: Feature Augmentation and right: Horizon Fault Deformation.

- *Magnetic Operator* is an operator created to improve a common task in traditional horizon extracting work flow, where the experts select a voxel to be used as a seed in a growing segmentation algorithm, resulting in a horizon patch. The magnetic operator uses a pre-segmented volume to snap a hole to the closest horizon patches, having the meaning of many seeds placed at the same time (Fig. 11).

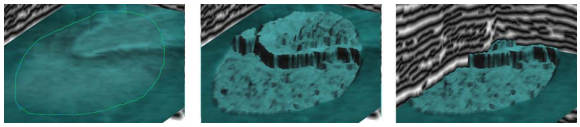

Fig. 11. Magnetic Operator with the pre-segmented volume.

- *Horizon Convex Sum and Coons Surface Operators* these operators create new surfaces inside the seismic volume. The first one uses 2 others horizons to create one between them. The latter allows the expert to draw strokes on the seismic data then it uses that to create a coons surface following the sketches (Fig. 12).

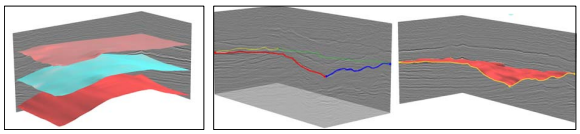

Fig. 12. Left:Horizon convex sum creates the surface at middle. Right: strokes and final coons surface.

It is important to remark that each of the presented operators has its own internal representation, e.g., parametric, implicit, height map. This flexibility along with the proposed framework allow us to build this system following the expert's desiderata. It is important to note that we are not claiming the GLaM as a contribution, it is only an illustrative example how the proposed framework can be used to create different sketch-based applications.

VI. Conclusion and Future Work

We proposed a framework to sketch-based surface modeling based on adaptive mesh. This framework was molded to build systems with three major characteristics: (1) Dynamic; the surface will change during the process. (2) Interactive; the users must be able to see the surface changes in interactive

time. (3) Controlled freedom; some applications have specific rules of modeling, the systems must incorporate these rules to guide the user, but without losing flexibility. As a proof of concept we present a sketch-based surface system based on the proposed framework, which approaches the problem of global changes versus local features, the DASS system. In addition, we illustrated the use of the framework applying that in a specific domain. The GLaM system, help us to show that the framework is flexible to build sketch-based modeling systems with very different requirements.

This work opens many interesting venues. One of the natural next steps is to use the framework in different domains and applications. The problem scratched by DASS system also has many interested open questions. For instance, we implemented a semi-automatic approach in which the user places the vertices to approximate the geometry and topology, followed by the base mesh creation in the space. This approach achieves good results, but, it only allows us to work in a single plane. Since the base mesh is responsible for the topology of the final model, we are restricted to topologies that can be handled in one plane. Thereupon, we plan to explore two approaches for the base mesh problem. Firstly, we intend to transport the actual semi-automatic solution to 3D, letting the user handle boxes directly in space. The main challenge of this approach is developing an effective interface. The other approach is to use a mesh simplification, for instance the method presented by Daniels et al. [34]. Although this approach is automatic, it starts with a dense mesh; we must then exchange the problem of how to find a base mesh for the problem of how to create a mesh with the correct topology. Concerning the atlas, we aim to develop mathematical and computational tools to handle the scale of the atlas as well as an interface to control predefined height-maps, and also algorithms that split the atlas if it has a high level of deformation in comparison to the surface.

ACKNOWLEDGEMENTS

We would like to thank our colleagues for their useful discussions and advice, in particular to Nicole Sultanum. We also thank the anonymous reviewers for their careful and valuable comments and suggestions. This research was supported in part by the NSERC/Alberta Innovates Technology Futures (AITF)/Foundation CMG Industrial Research Chair Program in Scalable Reservoir Visualization and by grants from the Brazilian funding agencies CNPq and CAPES/PDEE.

REFERENCES

[1] L. Olsen, F. F. Samavati, M. Costa Sousa, and J. Jorge, "Sketch-based modeling: a survey," *Comp. & Graph.*, vol. 33, no. 1, pp. 85–103, 2009.

[2] T. Igarashi, S. Matsuoka, and H. Tanaka, "Teddy: a sketching interface for 3D freeform design," in *SIGGRAPH '99*. ACM, 1999, pp. 409–416.

[3] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Fibermesh: designing freeform surfaces with 3D curves," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 41–50, 2007.

[4] Y. Gingold, T. Igarashi, and D. Zorin, "Structured annotations for 2D-to-3D modeling," *ACM Trans. Graph.*, vol. 28, no. 5, p. 148, 2009.

[5] E. Vital Brazil, I. Macêdo, M. Costa Sousa, L. H. de Figueiredo, and L. Velho, "Sketching variational Hermite-RBF implicits," in *SBIM '10*, 2010, pp. 1–8.

[6] G. Orbay and L. B. Kara, "Sketch-based surface design using malleable curve networks," *Comp. & Graph.*, vol. 6, no. 8, pp. 916 – 929, 2012.

[7] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes, "Sketch: an interface for sketching 3D scenes," in *Proc. of SIGGRAPH '96*. ACM, 1996, pp. 163–170.

[8] J. A. Jorge, N. F. Silva, and T. D. Cardoso, "Gides++," in *Proc. of 12th Encontro Português de Computação Gráfica*, 2003, http://vimmi.inesc-id.pt/~tfdc/gides++/index.html.

[9] B. Wyvill, A. Guy, and E. Galin, "Extending the CSG tree-warping, blending, and boolean operations in an implicit surface modeling system," *CGF*, vol. 18, no. 2, pp. 149–158, 1999.

[10] T. W. Sederberg, "Piecewise algebraic surface patches," *Computer Aided Geometric Design*, vol. 2, no. 1–3, pp. 53–59, 1985.

[11] J. Bloomenthal and K. Shoemake, "Convolution surfaces," in *SIGGRAPH '91*. ACM, 1991, pp. 251–256.

[12] L. B. Kara and K. Shimada, "Sketch-based 3D-shape creation for industrial styling design," *IEEE Comput. Graph. Appl.*, vol. 27, no. 1, pp. 60–71, 2007.

[13] J. J. Cherlin, F. Samavati, M. Costa Sousa, and J. A. Jorge, "Sketch-based modeling with few strokes," in *SCCG '05*. New York, NY, USA: ACM, 2005, pp. 137–145.

[14] A. Nasri, W. B. Karam, and F. Samavati, "Sketch-based subdivision models," in *SBIM '09*. ACM, 2009, pp. 53–60.

[15] H. Hnaidi, E. Guérin, S. Akkouche, A. Peytavie, and E. Galin, "Feature based terrain generation using diffusion equation," *CGF*, vol. 29, no. 7, pp. 2179–2186, 2010.

[16] O. Karpenko, J. F. Hughes, and R. Raskar, "Free-form sketching with variational implicit surfaces," *CGF*, vol. 21, pp. 585–594, 2002.

[17] R. Schmidt, B. Wyvill, M. Costa Sousa, and J. A. Jorge, "ShapeShop: Sketch-based solid modeling with blobtrees," in *SBIM '05*, 2005, pp. 53–62.

[18] A. Bernhardt, A. Pihuit, M.-P. Cani, and L. Barthe, "Matisse: Painting 2D regions for modeling free-form shapes," in *SBIM '08*, 2008, pp. 57–64.

[19] F. de Goes, S. Goldenstein, and L. Velho, "A simple and flexible framework to adapt dynamic meshes," *Comp. & Grap.*, vol. 32, no. 2, pp. 141–148, 2008.

[20] L. Velho, "A dynamic adaptive mesh library based on stellar operators," *Journal of graphics, gpu, and game tools*, vol. 9, no. 2, pp. 21–47, 2004.

[21] J. F. Blinn, "Simulation of wrinkled surfaces," in *Proc. of SIGGRAPH '78*. ACM, 1978, pp. 286–292.

[22] V. Krishnamurthy and M. Levoy, "Fitting smooth surfaces to dense polygon meshes," in *SIGGRAPH '96*. ACM, 1996, pp. 313–324.

[23] A. Lee, H. Moreton, and H. Hoppe, "Displaced subdivision surfaces," in *Proc. of SIGGRAPH '00*. ACM, 2000, pp. 85–94.

[24] D. Zorin, "Modeling with multiresolution subdivision surfaces," in *SIGGRAPH 2006*. ACM, 2006, pp. 30–50.

[25] C. Grimm and D. Zorin, "Surface modeling and parameterization with manifolds," in *ACM SIGGRAPH 2006 Courses*, ser. SIGGRAPH '06. New York, NY, USA: ACM, 2006, pp. 1–81.

[26] L. Velho, "Stellar subdivision grammars," in *SGP'03*. Eurographics Association, 2003, pp. 188–199.

[27] M. P. do Carmo, *Differential geometry of curves and surfaces*. Englewood Cliffs, N. J.: Prentice-Hall Inc., 1976.

[28] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, and D. Zorin, "State of the art in quad meshing," in *Eurographics STARS*, 2012.

[29] J. Bloomenthal, *An implicit surface polygonizer*. San Diego, CA, USA: Academic Press Professional, Inc., 1994, pp. 324–349.

[30] L. Velho, "Simple and efficient polygonization of implicit surfaces," *Journal of graphics, gpu, and game tools*, vol. 1, no. 2, pp. 5–24, 1996.

[31] B. T. Stander and J. C. Hart, "Guaranteeing the topology of an implicit surface polygonization for interactive modeling," in *SIGGRAPH 2005 Courses*. ACM, 2005.

[32] E. Vital Brazil, "Dass: Detail aware sketch-based surface modeling," *ArXiv Mathematics e-prints*, 2014, arXiv: 1406.7025.

[33] R. Amorim, E. Vital Brazil, D. Patel, and M. Costa Sousa, "Sketch modeling of seismic horizons from uncertainty," in *SBIM'12*. Eurographics Association, 2012, pp. 1–10.

[34] J. Daniels, C. T. Silva, J. Shepherd, and E. Cohen, "Quadrilateral mesh simplification," *ACM Trans. Graph.*, vol. 27, pp. 148:1–148:9, 2008.