Interactive high-level models for 3D virtual shape creation & animation

Damien Rohmer

Communauté
UNIVERSITÉ Grenoble Alpes

Habilitation à Diriger des Recherches
De l'Université Grenoble Alpes

Spécialité : **Informatique**

Présentée par

Damien ROHMER

Modèles interactifs de haut niveau pour la création et l'animation de formes 3D virtuelles

Interactive high-level models for 3D virtual shape creation & animation

Soutenue publiquement le **26 Juin 2017**
devant le jury composé de :

Pierre Poulin
Professeur, Université de Montréal, Rapporteur
Jean-Michel Dischler
Professeur, Université de Strasbourg, Rapporteur
Laurent Grisoni
Professeur, Université de Lille, Rapporteur
Stefanie Hahmann
Professeur, Université de Grenoble, Examinatrice
Rémi Ronfard
Directeur de Recherche, Inria, Examinateur
Valérie Perrier
Professeur, Université de Grenoble, Examinatrice

Abstract

3D virtual models are ubiquitous, notably spread to the general public by entertainment applications such as animation cinema and video games, and more recently, through virtual and augmented reality, or personalized 3D design. Thus 3D modeling evolves towards the need for increasingly complex 3D content that must remain visually plausible, and further be interactively created and deformed through efficient approaches allowing high-level control. In this context, various 3D models fulfilling these requirements through the integration of a priori knowledge are introduced. These models use high level geometrical constraints, potentially interleaved with underlying lower level models based on simulation or procedural generation, leading to hybrid approaches. Two axes of contributions are presented. First, 3D shape creation, applied to developable surface design, hierarchical shape deformation, and medical applications. Secondly, 3D animation, applied to articulated characters, interactive crumpled paper model, and fluid animation sculpting.

Résumé

L'utilisation des modèles virtuels 3D est aujourd'hui généralisée dans les domaines du loisir tels que le cinéma d'animation et les jeux vidéos, ainsi que, plus récemment, par le biais de la réalité virtuelle et augmentée, ou encore le prototypage 3D s'ouvrant au grand public. Ces évolutions génèrent des besoins de création de contenus virtuels 3D de complexité croissante qui doivent être à la fois plausibles visuellement, mais également pouvoir être générés et déformés interactivement à l'aide d'approches rapides offrant un contrôle de haut niveau. Dans ce contexte, nous proposons différents modèles virtuels satisfaisants à ces critères en intégrant des "a priori" sous forme de contraintes géométriques de haut niveau, et s'interfaçant potentiellement de manière hybride sur des modèles sous-jacents de plus bas niveau basés sur une simulation ou une génération procédurale. Deux axes de contributions sont présentés. Premièrement l'application de ces modèles pour la création de contenu 3D par le biais du design de surface développable, la déformation de formes hiérarchiques, et la modélisation du développement embryonnaire. Et deuxièmement, en animation 3D, s'appliquant aux personnages virtuels articulés, à l'interaction de papier virtuel froissé, ainsi qu'à la sculpture d'animation de fluide.

Contents

CHAPTER 1

Context and general objective

Nowadays, 3D virtual models are ubiquitous, notably spread to the general public by entertainment applications. Indeed, entertainment applications are related to large markets, such as video games (∼$100 Bn/y)[1], movie production (∼$100 Bn/y)[2] including animation cinema and special effects, or even the growing industry of virtual and augmented reality (∼$4 Bn/y)[3] which are obvious examples of 3D virtual model usages. The application of 3D computer graphics goes beyond entertainment, and several other application domains are directly related to it. Without being exhaustive, we can note for instance CAD applications at the origin of 3D modeling systems [Krull, 1994]. Medical domain, which is increasingly relying on 3D visualization and currently developing toward virtual and augmented reality [Ma *et al.*, 2014, Ruthenbeck and Reynolds, 2015]. And more generally, physical sciences, which are acquiring or computing data related to physical, potentially time dependent, phenomenon and rely on 3D models [Lipşa *et al.*, 2012].

3D models, in developing within several applications, have a strong and increasing impact on the society. The "quality" of the graphics, i.e. the increasing the number of polygons that computer can handle, and the improvement of the rendering algorithms, have led to more accurate and attractive representation of virtual models which may be nowadays hardly distinguishable from reality, thus leveraging the possibility immerse our imagination within alternative virtual reality. 3D data acquisition has also greatly been developed through the use of laser scanners or vision based approaches, enabling to capture with high precision the geometry of the real world. But beyond the acquisition of rendering of static geometry, the interaction itself with the 3D models is also currently deeply evolving. As 3D objects may nowadays be very complex [Zbrush, 3DCoat] and shared within the community [SketchFab] they can tremendously benefit of being reused, deformed, and animated, in order to be plugged into other virtual scenes. Another example of recent evolution is the design of manufacturable objects, which was initially restricted to professional engineering applications [AutoCAD, CATIA], is,

[1] The Global Game Market 2016, Newzoo
[2] IBISWorld, 2017
[3] Etat des lieux du marché de la réalité virtuelle, CSA, 2016

nowadays spreading to the general public thanks to the development of affordable fabrication technologies such as 3D printers. The 3D digital design also extend to several types of objects spanning rigid articulated ones to fabric. The application of 3D models is thus evolving from 3D model designed by professional only to be visualized, **toward dynamic interactive models that can be designed, edited, shared, along the user wishes or constraints, and be incorporated within a virtual scene or eventually fabricated for real**.

Challenges and traditional approaches

This increasing and evolving usage of 3D model raise new challenges. Thus, the way we model and interact with 3D data must evolve in order to handle the demand for larger quantity and quality, while allowing the user to interact efficiently with them. Traditional methods to model 3D objects defined by their boundary surface can generally be classified within four categories which have their own advantages and limitations.

First, the **manual approach** where a digital artist - or a technical engineer in the case of manufacturable object - fully define the 3D surface using dedicated modeling software such 3DSMax, Maya, or Blender. The object is usually defined starting from a rough low resolution model, and then further refined using mesh operations (extrusion, vertex displacement, splitting, etc), or using parametric NURBS surface deformation, until reaching the desired level of detail. If the object needs to be animated, a rig -a set of handles locally modifying the surface- can be defined. The approach has the advantage of providing the complete freedom on the resulting object which can be directly edited at any level. This make this choice the privileged one when defining 3D articulated characters or static shapes for entertainment purpose. The counterpart of such freedom is the technical and tedious manipulation of the shape which greatly increase when the surface contains details. Moreover, the surface is not constraints to preserve physical properties such as length or volume. Thus, when a deformation is applied to the model, a leg, arm, or the garments wore by a character, may elongate, or even intersect with other parts, leading to loss of visual plausibility. Such artifacts must then be manually corrected by the artist through all the frames.

Second, **3D acquisition** is able to capture both extremely detailed static shape using laser sensors, and natural character animation using *motion capture* approaches. Such acquisition guarantee the plausibility of the result as it is directly captured from the real world and is therefore of high interest when accurate geometry and motion is required. Such technology is now widely spread in special effects movie productions, and video games with large budgets. Despite evident progress of the technologies, pure acquisition based approaches cannot fulfill the need for complex 3D model. First, the methods are indeed limited to real shape and motion, which make them unsuitable to create purely imaginary scenarios, or even real phenomena which are hard to acquire. Second, the acquisition corresponds to a specific scenario to be used *as it is* as the data obtained by sensors are usually raw -high density point sets, polygon soup, or per vertex displacement- it does not exhibit a high level structure, as found in artist designed 3D shape, to help its deformation. Modifying the geometry, or the animation, thus usually requires to redo a complete acquisition, which may involve time consuming set up and a high cost.

Third, **physically-based simulation** consists in solving the differential equation expressing the evolution of the system starting from an initial configuration. Simulation is able to accu-

rately compute the animation of complex phenomenon such the dynamic of surfacic shapes (i.e. falling and colliding rigid bodies, garments) and volumetric phenomenon (i.e. fluids, smoke) which may be hard to model manually or to capture. It ensures that the result is accurate with respect to the considered local equations and is therefore well adapted to model complex natural phenomenon. Still, the approach is limited to the domain of validity of the underlying equation, and may not be able to model imaginary scenes. Simulation also usually requires to iteratively solve large system of non linear equations, and is thus computationally costly and prone to numerical divergence. Finally, the result obtained by a simulation cannot be modified directly. Instead, modifying a simulation implies to either change the initial conditions, or to change the parameters of the equation model, and then run the simulation again. In addition to the slow process, tuning the parameters may be very tricky in practice as the relation between the resulting visual shape and the parameters, or the initial conditions, may be largely non linear. One can consider for instance the difficulty to set up precisely the appearance of a wave within a fluid simulation by tuning parameters such as the density and viscosity coefficients.

Fourth, **procedural approaches** are dedicated method able to generate structured shapes from a set of basic coded geometrical laws called procedures. These procedures can be iteratively called and can lead to the generation of complex and large geometrical scenes, using sometimes a few numbers of coded functions, and thus a light memory footprint. Procedural approaches are of main interest to synthesize potentially very large and details virtual scenes, such as terrains, trees, cities, or buildings from a few set of parameters. Although procedural approach is very efficient to synthesize geometry from scratch, is it not well suited for iterative modification in order to converge toward an expected result. Modifying individual elements is usually not possible in the procedural approaches as it may not fit to the synthesis process based on the procedures. Instead, controlling a scene must be handled through the set of indirect parameters, which may globally change the appearance of the scene at each modification, thus leading to a result which is hard to control locally.

Objective and methodology of our approach

Our general objective is to provide efficient approaches to model 3D shape and animation that ease the process of 3D content creation, and thus enhance creativity as a result. We claim that such approaches must satisfy the three following conditions

1. **Plausibility**: We call plausibility the subjective fact that the visual result has a geometrical shape and potentially behave under deformation like expected. In our case, we will usually consider that plausibility is strongly related to some preservation of geometrical properties such as, for instance, length preservation for garments, or preservation of the relative position for objects distribution. Indeed, plausibility depends both on the type of object we aim at representing, and on the user expertise and expectations with respect to the virtual model.

2. **Low computational cost**: We call a method *fast* if it enables the user to directly interact with the result of the system. Ideally a method should be *real time*, i.e. being able to compute at least 25 frames per seconds for graphic applications, but it may not be necessary for all applications. For instance, synthesizing 3D geometry from user sketches may take a few seconds without perturbing the design process.

3. **High level control**: Contrary to *low level modeling* where the user must manipulate directly all the vertices of the visual mesh, we call high level control, the possibility to interact with the model using intermediate *handles* enabling the user to express his intention efficiently on the 3D model to be generated or to be deformed, while automatically adjusting the result if necessary. Ideal handles must allow *as direct as possible* edits on the shape at any level of detail, for instance based on mouse displacement on top of the viewed surface, or sketches, at the opposite of indirect GUI scale-bar/value based parameters in the standard procedural or simulation approaches. Once these handles are modified, an efficient automatic computation must ensure that the resulting shape remains plausible, thus making this step dependent of the two previous conditions. Providing the adapted trade off between user freedom and automatic constraint preservation is one of the key of the efficiency of the modeling approach, and is highly dependent on the nature of the represented shape and on the user.

We propose in this work different models satisfying these criteria. Our approach relies on using a mixture of the previous approaches, namely, manual modeling, data based approaches, physical simulation, and procedural modeling, to develop higher level for model and animation descriptions. We call these approaches **hybrids**. For instance, in modeling a surface by geometrical conical section in a physical simulation, instead of a more general triangular mesh, we were able to ensure the developability of the simulated surface -i.e. isometrical with respect to its pattern-, while easing the numerical convergence of the simulation as they do not require to be stiff. Another example consists in introducing direct manual control within procedural deformation approach, leading to a new way of expressing direct control on complex hierarchical virtual objects while preserving their consistency automatically.

The general methodology of our approach, as illustrated in Fig. 1.1, consists in introducing some *a priori* knowledge of the represented object within the description of the virtual model itself. These a priori are usually expressed as geometrical constraints, limiting the number of degrees of freedom of the model in order to offer relevant level of control, and automatizing the preservation of its plausibility. Using such methodology led us to several contributions in modeling and animation whose key ideas are explained in the manuscript.

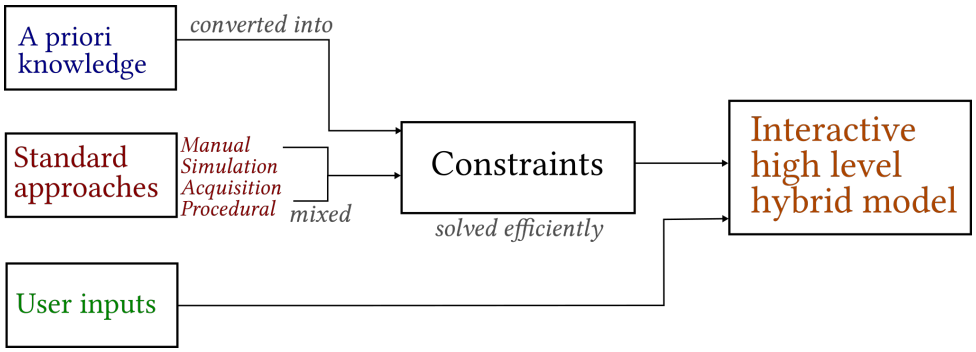


Figure 1.1: *General principle of our hybrid 3D model.*

In the context of 3D modeling, we first proposed a new method able to ease the design process of developable surfaces using input sketches (Chap. 2.1). Second, we proposed different methods to deform complex 3D models such as organic shape containing details, or assemblies. We further propose a general framework to handle hierarchical structure using a grammar formalism (Chap. 2.2). Third, in the context of medical sciences, we propose an ontology integrating the spatio-temporal description of the development of the human embryo (Chap. 2.3).

In the case of 3D animation, we propose an efficient implicit-based model for improving skinning deformation (Chap. 3.1). Second, we developed a complete model of interactive virtual paper handling crumpling, tearing, as well as its sound (Chap. 3.2). Third, we developed in the context of fluid animation, a copy & past sculpting approach with spatio-temporal elements enabling to easily modify an existing animation (Chap 3.3).

Time and structure organization

The different contributions mentioned in the manuscript have been developed between 2011 and 2017. They involved my research performed within several collaborations. First of all, the two PhD students that I co-supervised and who defended at the end of 2016. Ulysse Vimont, also co-supervised with Marie-Paule Cani, with whom we developed the notion of complex models and most of the contributions associated to it (Chap. 2.2.3 and Chap. 2.2.4). Ulysse also worked, in team with Pierre-Luc Manteaux, another PhD student of the Imagine team, on the fluid sculpting project (Chap. 3.3). Second, Camille Schreck, also co-supervised with Stefanie Hahmann, with whom we developed the entire model of interactive virtual paper (Chap. 3.2).

Other projects were developed as follow up of Master students internships. Amaury Jung, with whom we started the work of the sketch based developable surface modeling (Chap. 2.1.2) in collaboration with Marie-Paule Cani, Stefanie Hahmann, and a private fashion company. The continuity of the project to single view sketch is currently performed with Amelie Fondevilla, initially in Master, and currently in PhD (Chap. 2.1.3). We also started the specific work on the deformation of mesh containing details with Nicole Cogo (Chap. 2.2.2) that I completed later on. The embryo development model was developed within the Master project of Benoit Massé for the geometrical aspect, and Frederico Ulliana for the ontology, in collaboration with Olivier Palombi (Chap. 2.3).

Finally, the implicit skinning project (Chap. 3.1) has been developed in large part by Rodolphe Vaillant, PhD students at IRIT, Univ. Toulouse, with Loic Barthe and Marie-Paule Cani.

The following graph describes the temporal scale of each project and the involved students.

Note that several of these projects have been realized within national and international collaborations. My main external collaborators from other Universities where respectively.

- Niloy Mitra from University College London, and Michael Wand from Utrecht University, who were the main coordinators on the project of deformation and synthesis of new assemblies (Chap. 2.2.3).
- Adrien Bousseau from Inria Sophia Antipolis, with whom we collaborated on the single view sketch of developable surface (Chap.2.1.3).
- Loic Barthe from IRIT, Univ. Toulouse, and Gaël Guennebaud from Inria Bordeaux, who were the main coordinators on the implicit skinning project (Chap. 3.1).
- Charlie Wang from the Chinease University of Hong-Kong who developed a subpart, namely the isometric tracking, integrated in our geometrical crumpled paper model (Chap. 3.2.1).

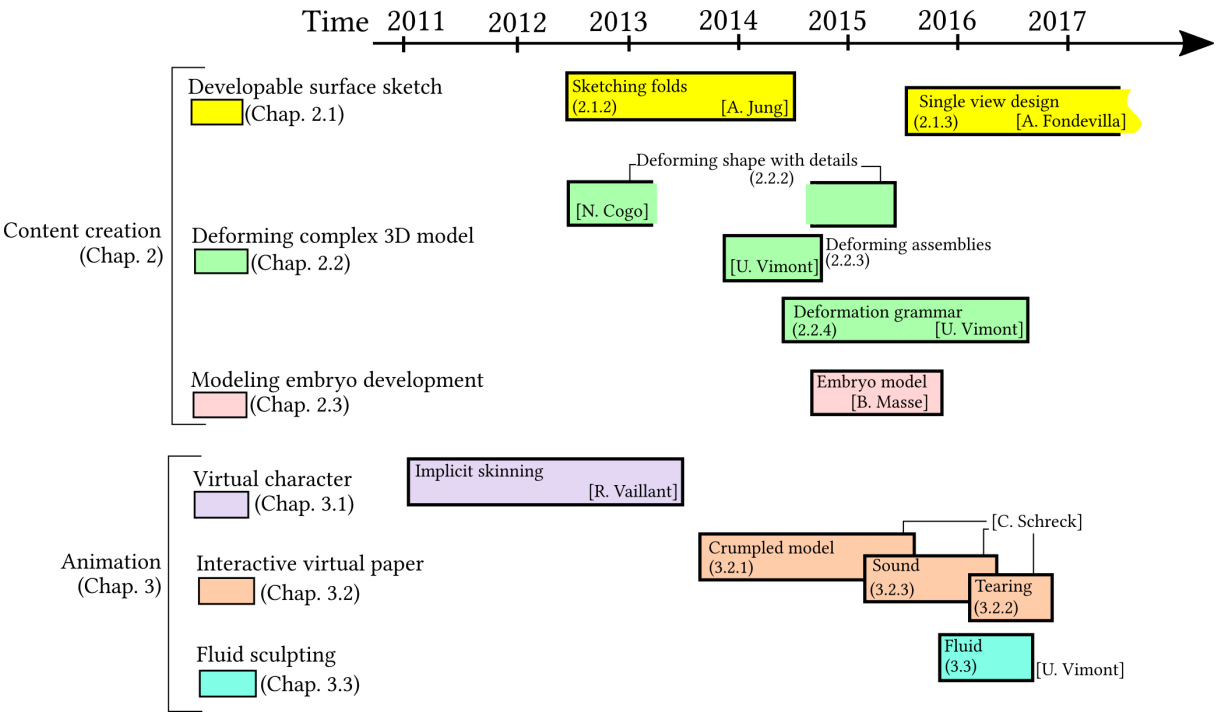


Figure 1.2: *Time spanned, and students involved, in the different projects.*

- Doug James from Stanford University, who collaborated on our sound model for paper crumpling (Chap. 3.2.3). His contribution allowed the integration of the sound spatialization in the model.
- Chris Wojtan from IST Austria, who collaborated on the fluid sculpting project (Chap. 3.3) thanks to his experience in fluid simulation.

Publications

The mentioned projects led to the following publications.

Peer Reviewed International Journal Publications

- Patterns from Photograph: Reverse-Engineering Developable Products. *Amélie Fondevilla, Adrien Bousseau, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* Accepted to **Computer & Graphics, Proc. Shape Modeling International**, 2017.
- Interactive Paper Tearing. *Camille Schreck, Damien Rohmer, Stefanie Hahmann.* **Computer Graphics Forum, Proc. EUROGRAPHICS**, 36(2), 2017.
- Deformation Grammars: Hierarchical Constraints Preservation under Deformation. *Ulysse Vimont, Damien Rohmer, Antoine Begault, Marie-Paule Cani.* **Computer Graphics Forum**. 2017
- Non-Smooth Developable Geometry for Interactively Animating Paper Crumpling. *Camille Schreck, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani, Shuo Jin, Charlie*

C.L. Wang, Jean-Francis Bloch. **ACM Transactions on Graphics**, 35(1), 2015.
Presented at ACM SIGGRAPH 2016.

- My Corporis Fabrica Embryo: An ontology-based 3D spatio-temporal modeling of human embryo development. *Pierre-Yves Rabattu, Benoit Massé, Federico Ulliana, Marie-Christine Rousset, Damien Rohmer, Jean-Claude Léon, Olivier Palombi.* **Journal of Biomedical Semantics**, 6(36), 2015.

- Sketching Folds: Developable Surfaces from Non-Planar Silhouettes. *Amaury Jung, Stefanie Hahmann, Damien Rohmer, Antoine Begault, Laurence Boissieux, Marie-Paule Cani.* **ACM Transaction on Graphics**, 34(5), 2015.
Presented at ACM SIGGRAPH Asia 2016.

- Real-Time Continuous Self Replicating Details for Shape Deformation. *Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* **Computer & Graphics, Proc. Shape Modeling International**, 51, 2015.

- Replaceable Substructures for Efficient Part-Based Modeling. *Han Liu, Ulysse Vimont, Michael Wand, Marie-Paule Cani, Stefanie Hahmann, Damien Rohmer, Niloy J. Mitra.* **Computer Graphics Forum, Proc. EUROGRAPHICS**, 34(2), 2015.

- Implicit Skinning: Real-Time Skin Deformation with Contact Modeling. *Rodolphe Vaillant, Loic Barthe, Gael Guennebaud, Marie-Paule Cani, Damien Rohmer, Brian Wyvill, Olivier Gourmel, Mathias Paulin.* **ACM Transactions on Graphics, Proc. ACM SIGGRAPH**, 32(4), 2013.

Peer Reviewed International Conference Publications

- Space-Time Sculpting of Liquid Animation. *Pierre-Luc Manteaux, Ulysse Vimont, Chris Wojtan, Damien Rohmer, Marie-Paule Cani.* **Motion In Games**, 2016.

- Real-time sound synthesis for paper material based on geometric analysis. *Camille Schreck, Damien Rohmer, Doug James, Stefanie Hahmann, Marie-Paule Cani.* **Symposium on Computer Animation**, 2016.

Other publications: Workshops and posters

- Towards developable products from a sketch. *Amélie Fondevilla, Adrien Bousseau, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* EUROGRAPHICS Poster, 2017.

- Modeling Symmetric Developable Surfaces from a Single Photo. *Amélie Fondevilla, Adrien Bousseau, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* Journées d'Informatique Graphique (JFIG), 2016.

- Interactively Animating Crumpled Paper. *Camille Schreck, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* Proc. WomENcourage, 2015.

- Synthèse de son de papier adaptée au mouvement et à la géométrie de la surface. *Camille Schreck, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* Journées AFIG, 2015.

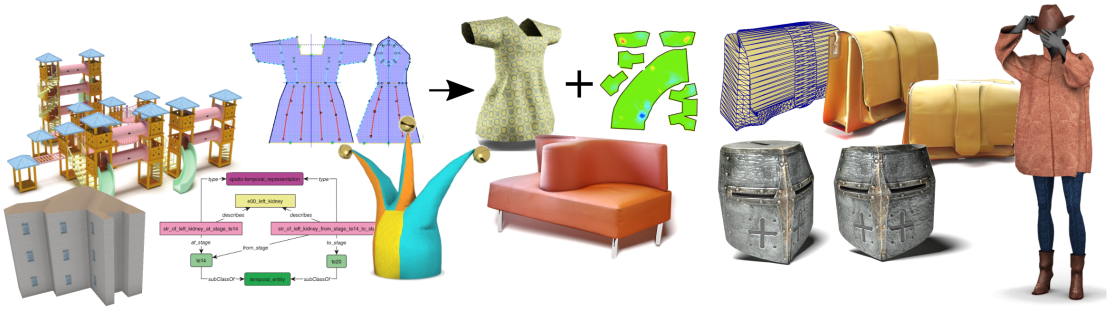
- Animation interactive de papier froissé. *Camille Schreck, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* Journées GTMG, 2015.
- Déformation d'objet complexe par graphe d'interaction. *Ulysse Vimont, Damien Rohmer, Marie-Paule Cani.* Journées GTMG, 2015.
- Duplication de détails pour la déformation de surfaces. *Nicole Cogo, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani.* Journées GTMG, 2015.
- Déformation de la peau d'un personnage avec prise en compte des contacts. *Rodolphe Vaillant, Loic Barthe, Guaël Guennebaud, Marie-Paule Cani, Damien Rohmer, Bryan Wyvill.* REFIG, 6(2), Journées AFIG, 2012.

CHAPTER 2

3D models for content creation

In this chapter, we present several efficient solution to create or deform 3D virtual objects characterized by geometrical properties which are notably hard to model using manual approaches. 3D shape creation is a complex domain requiring both artistic and technical skills, as well as professional expertise. These skills may be spread through several individuals, or even through several software which may be incompatible for each other. Thus, creating a 3D shape may require to go along, possibly long and costly, back and forth steps between the initial artistic direction, technical limitations, and geometrical constraints that the shape must satisfy to remain plausible, or to be physically fabricated.

We show in this chapter that the creation process can be simplified, enabling an efficient interaction between the initial design, and the generated 3D shape. Ideally, a 3D shape should be generated seamlessly from the initial designer wish that could directly and iteratively interact between his design and the 3D shape. To achieve such goal, the constraints that the final shape must satisfy should be automatically integrated in order to avoid re-specifying them for each editing step. Moreover, allowing direct control on the shape or the design while ensuring its plausibility is also essential. In all cases, we follow the following methodology consisting in first, extracting and understanding the specific knowledge and *a priori* that a plausible shape must exhibit. Secondly, to translate this knowledge into geometrical constraints. Thirdly, to efficiently integrate these constraints within the virtual 3D model.



2.1 Generating developable surface from sketches. Application to design.

The first application we consider is the design of products made from piece of pattern, which can typically be applied to fashion products made of cloth or leather material (i.e. boots, bags, hats). One of the key observation of such design process is that designers create their model by sketching them on a sheet of paper as a first step. These sketches depict the 2D projected view the final mounted product.

In the real manufacturing process, a professional pattern-maker, will create the first prototype of the real object in inferring the 2D patterns from the designer sketch. Once the 2D patterns are provided, their shape can be cut within the material, assembled and seamed together, to observe the real object. The resulting real prototype may indeed not fit exactly to the original design sketch which depicts a specific shape, style, and design folds. Thus, this process is actually iterative, and modifications must then be made on the 2D patterns, which are used themselves to generate another object in an iterative way.

Some observations can be made on this current pipeline. First, the process of generating pattern from a 2D sketch of the 3D model is a very complex manual and error prone task. This pattern making process is performed using a set of specific expert knowledge depending on experiences and brands, but no clear algorithm can been defined so far. Second, each iteration is actually made using real material leading to a time consuming and costly development. As a result, the designer cannot directly interact with the creation process in order to express its creativity while checking the feasibility of the resulting object. Such convergence only emerges from a slow iterative process. Our objective was to simplify such process in order to allow both, design creativity, and feasibility checking, in a short iterative process using virtual models. We proposed a method taking as input a set of 2D sketches, and able to automatically compute a virtual 3D quasi developable model and its associated 2D patterns. As a result, a real model can still be fabricated with real material using the 2D computed patterns, with the quantified guarantee that the model is feasible.

Before describing the general approach, let us first formalize the type of surface we consider in this work, namely developable surfaces, which will also be considered for animation latter on in Chap. 3.2.

2.1.1 General concepts on developable surface

We remind that garments are made of 2D patterns which are bent in space and assembled together to produce the final object. Each bent piece of pattern is associated with few stretch and compression which depends on the type of material. In the case of leather material, stretching is very limited at the macroscopic level, and we assume that the deformed 3D surface can be considered as isometric with respect to its 2D pattern.

The general class of surfaces that can be isometrically deformed toward a planar 2D pattern are called *developable*. This property allows developable surfaces to be easily fabricated by bending flat sheet of material, and thus, they are associated with a wide variety of applications going beyond garment design, such as for instance, architecture [Pottmann *et al.*, 2007], CAD modeling [Pérez and Suàrez, 2007], or paper structure [Solomon *et al.*, 2012].

A specific subclass of *developable* surfaces are the smooth C^2-developable surfaces, meaning that the surfaces can be defined by a $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ mapping that is twice differentiable. The C^2-developable surfaces can be defined by the two following equivalent properties [Carmo, 1976, Gray *et al.*, 2006]

- The surface has a zero Gaussian curvature everywhere. Which means geometrically that the surface can only be bent in one direction, while the orthogonal direction must necessarily be flat. As a corollary, the Gauss map, i.e. the image of the normals of the surface on the unit sphere is a curve (and not a general subset of the sphere surface as for other types of smooth surface).
- The surface is a ruled, i.e. sweep surface generated by a straight line section, with constant normal along the rulings. The surface can be parameterized by

$$S(u, v) = \varphi(u) + v\,\psi(u) \text{ with } \det(\varphi'(u), \psi(u), \psi'(u)) = 0\,.$$

As a corollary, C^2-developable surfaces can be either plane, generalized cylinders: all rulings are parallel, generalized cones: all rulings converge toward a single point, or tangent developable: rulings are tangent to a curve.

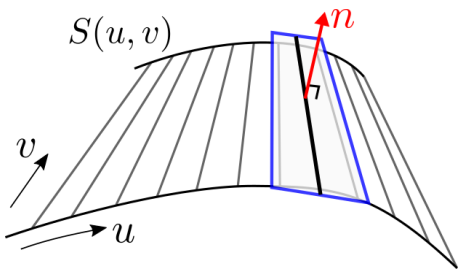


Figure 2.1: *Exemple of developable surface*

The ideal case of the theory of C^2 developable surface may however not be directly applicable to real material, and two hypotheses may not be fulfilled. First, natural materials have, to various extents, elastic properties leading to limited allowable stretching. As a result, their 3D mapping may not be exactly isometric with respect to their 2D pattern. Secondly, the C^2 smoothness assumption, implying continuous curvature variation, may not hold globally at the macroscopic surface level. Thin rigid sheets may exhibit fast changing curvature when folded, as it is for instance the case for paper material. These local variations may be modeled as singular curves or points, which lead to surface with C^1 or only C^0 continuity. Thus, the smooth approximation may only hold piecewisely over the surface.

While the specific case of singularity from C^0 continuity will be handled in our work on virtual paper model later in Chap. 3.2, we will consider in this current section a smooth surface with approximate isometry with respect to its pattern. We will call the 3D surface *approximately developable*, and will exhibit a low, but not necessarily zero, Gaussian curvature. Note that such approximate developability is mandatory to model surface folding that we call *interior folds*, which are not already depicted by the surface boundaries. Indeed, C^2 developable, being globally ruled surfaces, could not handle such interior folds.

2.1.2 Developable surface from multiple views sketches

***Note**: Further details on the following part can be found in [Jung et al., 2015].*

We now describe an efficient generation of a developable surface from annotated sketches depicting the object from two or three orthogonal views. We focus on leather products which

are supposed to be sufficiently rigid to stand in a stable position as sketched, and be hardly impacted by gravity and dynamic. Our method handle the following properties.

1. **Developability**. In order to be fabricated from its pattern, the generated 3D object should be developable with respect to its pattern. Our approach allows approximated surface developability which can be quantified and visualized.

2. **Silhouette matching**. The silhouette depicted on the 2D sketch relates to the designer expectation. This silhouette should thus be the one of the generated 3D model. Our approach is able to ensure that the silhouette of the 3D model exactly match to the one of the sketch.

3. **Designed folds**. The sketched shape may exhibit specific fold indications. These folds are part of the design and should be present in the final model. Our method ensures that the pattern lengths are such that, once mounted, each piece of surface is able to fold in the expected way.

4. **Non planar silhouette**. Points on the silhouettes of folded products may form sets of disconnected non-planar curves on the 3D model. One of our scientific contribution consists in proposing a new, so called, *zippering* algorithm, able to handle such features, thus allowing the necessary flexibility to model folds on developable surface.

Related works

On the one hand, two general approaches have been studied in the literature to model developable surfaces. First, the direct modeling using, for instance, 3D contour curves [Frey, 2004, Rose *et al.*, 2007], contour strips [Tang and Wang, 2005], 3D positional constraints [Peternell, 2004], geodesic curves [Bo and Wang, 2007], or folding and angular constraints [Solomon *et al.*, 2012], but these methods do not take into account 2D silhouette constraints. Second, in optimizing the developability of an already existing mesh, in minimizing for instance the angular defect around each vertex [Wang and Tang, 2004, Wang, 2008, Tang and Chen, 2009], or in approximating locally the surfaces with conical section [Decaudin *et al.*, 2006]. Note that our approach will extend this last method in our case in order to optimize the surface developability while preserving existing surface wrinkles.

On the other hand, sketches approaches have been used to reconstruct 3D models from single and multi-view inputs. While single view sketches have been able to generate 3D surfaces from geometrical constraints such as parallelism and orthogonality to model polygonal structures [Lipson and Shpitalni, 1996], or more recently, using cross sections to generate designed shapes [Xu *et al.*, 2014], but they require typical engineering inputs which are not standard in garments design. Sketching on top of existing 3D mannequins has also been successfully used [Turquin *et al.*, 2007, Robson *et al.*, 2011]. Multi-view sketches allows the user to handle 3D depth and were used to model freeform surfaces [Mori and Igarashi, 2007], or, closer to our approach, surfaces interpolating a given silhouette in constructing its visual hull [Rivers *et al.*, 2010]. We will consider the latter approach as an initial mesh surface for our method, and then extend it to handle designed folds and non planar silhouette, while enforcing the developability of the resulting surface.

General approach

We consider as input two annotated sketch depicting the front and side orthogonal projection of a model, with an additional third top view in some cases. We also consider four types of features annotated as stroke type on the sketch as depicted in Fig. 2.2

- The *silhouettes*, which are delimiting the 2D projection of the sketched model.
- The *borders*, representing the external boundaries of an open surface.
- The *seams*, where individual surface panels are sewn together.
- The *folds*, depicting the contours of a fold.

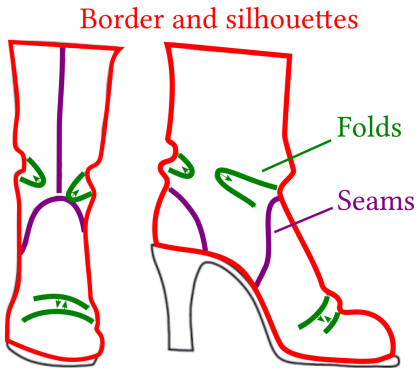


Figure 2.2: *Input annotated sketch depicting the border, silhouette, folds, and seams of the model under two orthogonal views.*

The overall approach works as follows (see Fig. 2.3). First, an initial draft 3D shape corresponding to the visual hull is synthesized from the two, or three, orthogonal view [Rivers *et al.*, 2010]. We call this hull mesh S_H. Although this mesh perfectly fit to the 2D silhouette, it doesn't correspond to a smooth developable surface. We then initialize a first draft surface in smoothing S_H using a standard laplacien smoothing [Taubin, 1995]. This draft, called S, is both not developable and do not fit to the silhouette.

We then propose an iterative optimization which interleaves two steps. A first step enforcing the surface to match the objective silhouette, and second step which optimizes the developability of the surface while preserving the folds. After a few iterations of such steps -usually less than 10- a quasi-developable surface is obtained, and the 2D patterns can be computed using standard parameterization tools [Sheffer *et al.*, 2005].

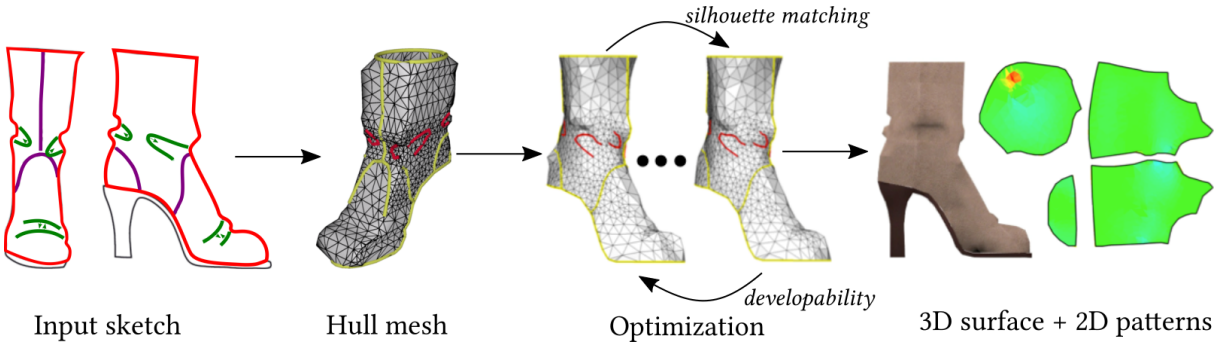


Figure 2.3: *General algorithm of our approach. The input sketch is converted into the hull model S_H. Then we perform a series of interleaved optimization steps with respect to developability and silhouette matching, resulting in the 3D surface and its 2D patterns.*

The general idea of our silhouette matching algorithm, illustrated in Fig. 2.4, consists in marching along the silhouette of the 2D orthogonal view, and selecting the most protruding edges of the current mesh with respect to the initial hull, while ensuring a full coverage of the silhouette. The vertices of the selected protruding edge are then set to have constrained coordinate

values from the initial hull in the viewed direction, thus corresponding to the sketched silhouette. Then, the actual projection of the vertices is performed as a global deformation applied to all the vertices of the mesh to satisfy the prescribed coordinate constraints, while minimizing, in a least square sense [Sumner and Popovic, 2014], the deformation of the triangles.

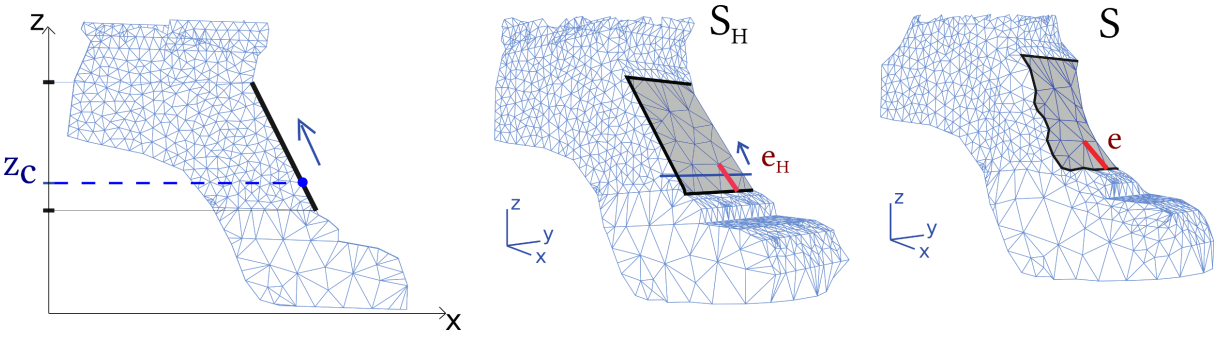


Figure 2.4: *Our silhouette matching algorithm. The current value z_c is propagated along the silhouette edge in the orthogonal view (bold black segment). Each most protruding edge $e \in S$ with respect to its image in the hull $e_H \in S_H$ is selected, such that the silhouette is fully covered. Note that the protuberance of an edge is computed as the signed shorted orthogonal distance between e and e_H.*

The second optimization step of our method consists in handling pre-design folds while optimizing the surface developability. Let us first explain how folds are handled.

As noted by Popa *et al.* [2009], visible contours of folds can be characterized by a contour where the surface exhibits a constant angle ranging between $[30°, 60°]$ with respect to the view point. In our case, we considered that the drawn fold lines on a given view corresponds to a constant angle of $60°$ with the respective viewpoint as seen in Fig. 2.5. This angle is set as an objective target rotation $\mathbf{R}_{\text{fold}}$ for each triangle crossing a fold line and its rotation axis is given by the edge belonging to the fold.

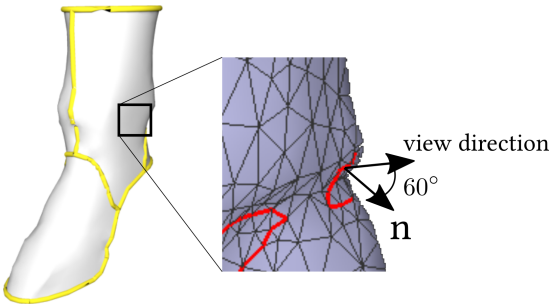


Figure 2.5: *Folds are defined by an angle of $60°$ between the surface normal and the view direction.*

In order to control the fold width, we spread this objective rotation to the neighboring triangles, and associates a weighting parameter ω equals to 1 on the fold, and smoothly decreasing toward 0 with respect to the geodesic distance to the fold line.

Once a target rotation has been set for the neighboring triangles to the folds, our objective is to derive a developability optimization compatible with such constraints. In our case, one of the most appropriate model consists in approximating locally the surface by cones (handling in degenerated case also cylindrical and planar surfaces). Cones can be characterized by having a constant normal angle θ with respect to a given axis A. Our approach consists in fitting the best axis angle A^* and normal angle θ^* for each triangle associated to its one ring neighborhood, to deduce another objective rotation $\mathbf{R}_{\text{developability}}$ maximizing the developability. Let us call $\hat{n}$ the averaged normal of a given triangle t in its neighborhood $N(t)$. The best axis angle A^* is computed as

$$A^* = \underset{\substack{A \\ \|A\|=1}}{\arg\min} \sum_{j \in N(t)} \left((n_j - \hat{n}) \cdot A\right)^2 , \tag{2.1}$$

where n_j is the normal of the neighboring triangle j. Note that the optimal value of this quadratic energy can be directly computed as the smallest singular value of the matrix associated to the values $n_j - \hat{n}$. Finally, each triangle can be associated with a target rotation $\mathbf{R}$ computed as a weighted spherical interpolation between $\mathbf{R}_{\text{fold}}$ and $\mathbf{R}_{\text{developability}}$ taking thus into account both developability and folds preservation. All vertices are then optimized using the deformation transfer approach [Sumner and Popovic, 2014] minimizing the expression

$$\sum_{\text{triangles}} \|P_{developable}\, P^{-1} - \mathbf{R}\|_F^2 \,. \tag{2.2}$$

where, P is the 3×3 matrix such that $P = (p_4 - p_1, p_4 - p_2, p_4 - p_3)$, with (p_1, p_2, p_3) being the vertices of a triangle of S, p_4 defined by offsetting p_1 in the normal direction of the triangle. $P_{developable}$ corresponds to the same 3×3 matrix expressed as the unknown vertices of the surface $S_{developable}$ being obtained after this developability optimization. And $\| \ \|_F$ is the Frobenius matrix norm.

Results

We show in Fig. 2.6 the interface developed for this project. The top figure shows the first part where the user draws the contour over an underlying scanned image. In order to ensure coherent dimensions between the orthogonal views, a snapping mechanism is shown by the set of lines spanning the different views. The bottom part of the image shows the result obtained from the completed sketch with the 3D model and the 2D pattern.

Figure 2.6: *Interface of our sketching algorithm*

We applied our algorithm on several examples of fashion products. Fig. 2.7 shows the result for the boot model used in our illustrations. Note that we could fabricate a real model of this

shoe in cutting the computed pattern inside fabric material. The approximate developability is visually quantified using color encoding as seen if Fig. 2.7 (c). The green color indicates perfect length preservation between the 3D model and the 2D pattern, and thus good developability property, while the red color indicates elongation of the 3D model. Note that elongation may be part of a design process, as in this case at the tip of the boot which may be thermoformed.

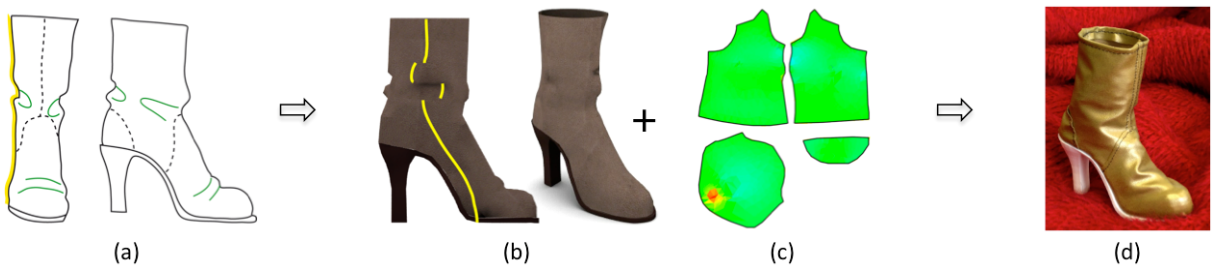


Figure 2.7: *Modeling a boot model from input annotated sketch (a). The resulting 3D model (b), and the computed 2D pattern (c) can be used to fabricate a real boot (d).*

All the examples were computed in less than 5s. This short computation time allows the user to have a fast feedback of his design. The color encoding of the developability error further provides interesting feedback to iterate over the model, suggesting to the user the need to add darts or new seams in some region.

An example of iterative design of hat is proposed in Fig. 2.8, where a first model exhibits non developable property at the top of the hat. In a second step, the user adds design folds in his sketch which changes the 3D model but doesn't solve the non developable property. In a third step, the user adds a new seam, leading to a new pattern piece at the top of the hat, thus allowing it to become quasi developable. At the current state, the interpretation of the non-developable part is fully let to the user, however, following some recent work on suggestive design [Umetani *et al.*, 2012], proposing such modification on the design sketch could be an interesting extension of the work.

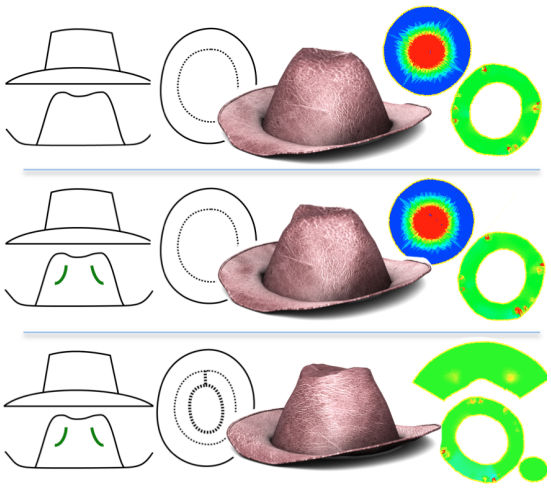


Figure 2.8: *Three successive iterations in designing a developable hat.*

We also show demonstrated our method can handle non planar silhouette as illustrated in Fig. 2.9, where continuous silhouette curves on the 2D sketch actually correspond to 3D discontinuous curves on the surface. This effects is generated by the large interior fold indicated in green on the sketch.

Finally, we show that the resulting designed model can be further used as input mesh as a cloth simulation as illustrated in Fig. 2.10 where the tunic and the coat set over a virtual mannequin. Note also the presence of the large folds on the surface which is part of the design.

Figure 2.9: *The blue purse exhibits non planar silhouette as depicted by the yellow curve.*

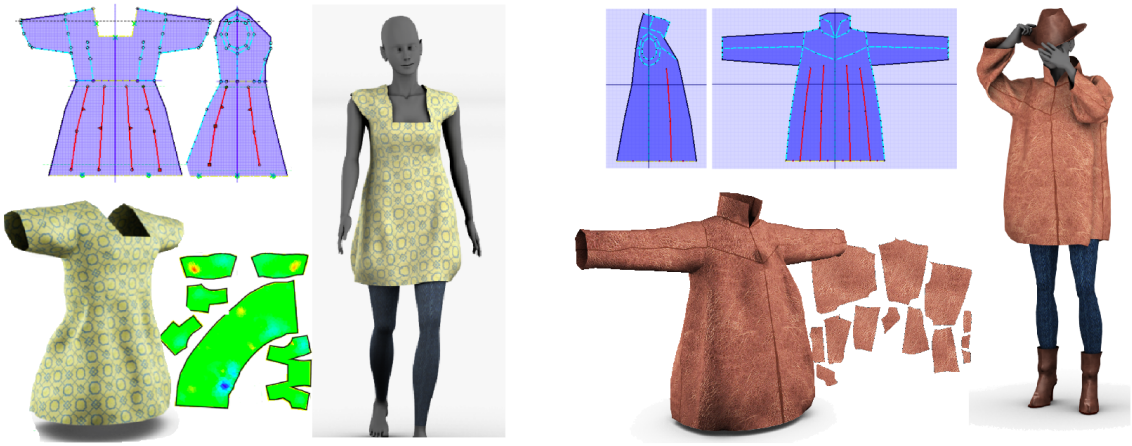


Figure 2.10: *Example of designed cloth plugged as input of an external cloth simulator to model animated dressed characters.*

2.1.3 Developable surface from a single view sketch

***Note**: Further details on the following part can be found in [Fondevilla et al., 2017].*

The previous approach described a method able to generate developable surface from two to three annotated sketches corresponding to orthogonal views. However, providing such input may not be trivial as the different sketches must be coherent each other, and thus require specific expertise. In order to ease the design of the user input, one could aim at generating a 3D model from a single view only. This could for instance allow the general public to model a 3D shape in annotating the silhouette on top of a real picture. The problem of generating 3D surface from a single 2D view is however more challenging as 3D depth must be fully inferred from 2D values only. In order to tackle this new challenge, we consider a reduced problem where we consider the following assumptions.

- The shape consists of a set of C^2 developable surfaces patches made of planar, cylindrical, and conical regions. Thus, each patch is a smooth surface without interior folds
- The shape is supposed to exhibits a global mirror symmetrical shape, allowing us to retrieve 3D depth information and occluded regions.
- Each patch of developable surface is supposed to have a symmetrical pattern, and the conical regions are supposed to be bounded by two parallel planes, ensuring that rulings intersect the boundaries with symmetrical angle in both 3D and 2D.

An interesting property of C^2 developable surface is that their visible contours are either made by its boundaries, or by straight silhouettes corresponding to their rulings [Fondevilla *et al.*, 2017]. Note that we consider the *silhouette*, following the definition of computer vision, as being the set of points with normals orthogonal to the viewing direction [Barrow and Tenenbaum, 1981]. Thanks to this property we propose an algorithm able to retrieve the ruled structure of the developable surfaces on its 2D projection as soon as a silhouette is visible, thus integrating the developability criteria as the core of our reconstruction method. In addition, the symmetry assumption allows to infer 3D depth information from a subset of symmetrical pairs of positions. Combining 3D information from a coarse set of points with the information of the rulings structure allows to compute the 3D information of developable surface using a global optimization algorithm.

In this case, as depicted in Fig. 2.11, the user depicts, on top of a single photograph, the visible borders of the object, the silhouettes, pairs of symmetric points, and position of occlusions. To infer the global symmetry, the user further draw two vectors which are the projection of two orthogonal basis vector of the symmetry plane.

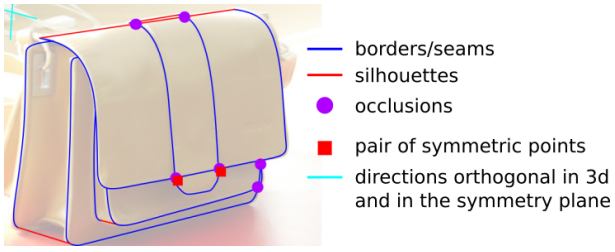


***Figure 2.11:** Annotation on top of a single viewed bag.*

Single view reconstruction algorithm

We summarize in the following the three main steps of our reconstruction algorithm (see Fig. 2.12).

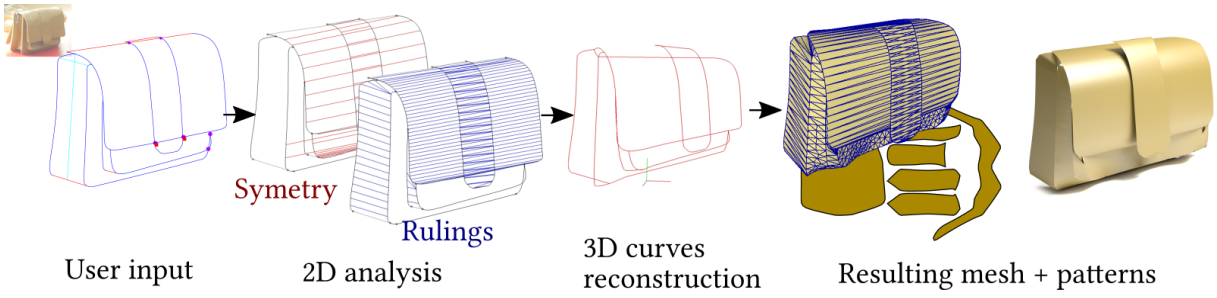


Figure 2.12: *Main steps of our single view developable surface reconstruction.*

In a first step, our approach perform an analysis of the 2D curves. This analysis computes, on one side, pointwise correspondences of pair of symmetric points with respect to the depicted symmetry plane, and on the other side, a set of 2D rulings on each depicted patch containing a silhouette. These rulings are computed as a modified time warping optimization [Keogh and Pazzani, 2001]. The algorithm starts from the annotated silhouette -which is a ruling-, and propagates the rulings in optimizing two criteria: equality between the extremal angles made between the rulings and the boundary curve, and collinearity of successive rulings.

In a second step, we compute the 3D coordinates of the curves as the optimal solution of a quadratic energy formulation. We considered an energy composed of four terms. Two terms were inspired from the approach from Xu *et al.* [Xu *et al.*, 2014], namely, the accuracy of the reconstruction with respect to the 2D sketch, and the so called, minimal variation, favoring a linear relation between the variation of the curve in the viewed plane and in depth direction. The two last energy terms we proposed are related to the symmetry, extending the work developed by Cordier *et al.* [2013], ensuring that points annotated as symmetrical in the previous step have symmetrical 3D coordinates, and a developability term, favoring constant tangent planes along the detected rulings.

In a third step, the reconstruction is filled by surfaces interpolating the 3D curve boundaries. The surfaces are either computed trivially when a patch contains rulings, or using a minimal means curvature interpolation [Botsch and Kobbelt, 2004]. Finally, the resulting mesh can be exactly symmetrized in duplicating the most visible side of the object, and the patterns can be extracted using a parameterization algorithm [Sheffer *et al.*, 2005].

Our approach is able to successfully reconstruct various objects made of developable patches as illustrated in Fig. 2.13 with a purse, tent, and helmet object model.

2.1.4 Discussion

We presented two approaches allowing to integrate developable a priori constraint within an annotated sketch based modeling system. While the first one requires two to three orthogonal views of the same object, and is able to handle general quasi-developable surface model, the second approach consider a more restrictive case of symmetric objects without internal folds,

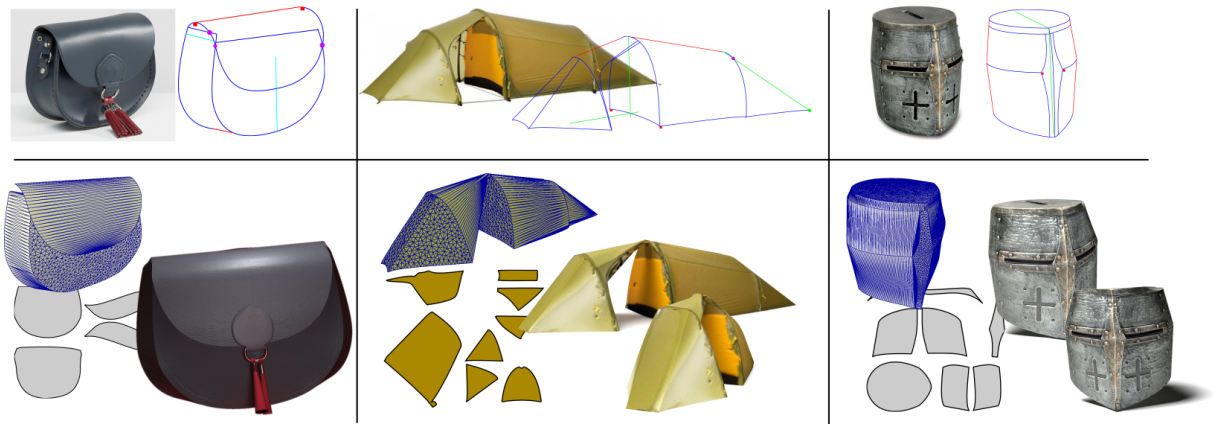


Figure 2.13: *Top row: inspiration photograph and input sketches. Bottom: Resulting 3D reconstruction and patterns. Note that the texture of the photograph image has been used to texture the 3D surfaces.*

made by pieces of planar, cylindrical, and conical section, while allowing to design the 3D model from only a single view. Both approaches, although not real time, remain sufficiently efficient, thanks to the restriction to the use of quadratic problem, to provide feedback to the user within a few seconds. This allows the user to interact with the result through a set of changes, or in adding annotations in its design, as an iterative process.

While single view approach allows a much wider type of application, ranging from existing photo annotation, to artist design sketches, we could note that the lack of depth information, associated with the assumption of orthographic projection, involves a higher sensibility of the result with respect to the given input. Therefore, the sketch may need to be modified several times in order to converge toward the expected result. Integrating new, possibly non linear, geometrical constraints such as orthogonality, or at least constrained angle relations, could be a way of improving the control on the result. Another avenue to improve such control would be the development of a more interactive user interface, allowing the sketch/rotate/sketch approach [Bae *et al.*, 2008]. A single sketch could be used to define a base 3D model, which could then be refined in the 3D space, notably with respect to the missing depth dimension.

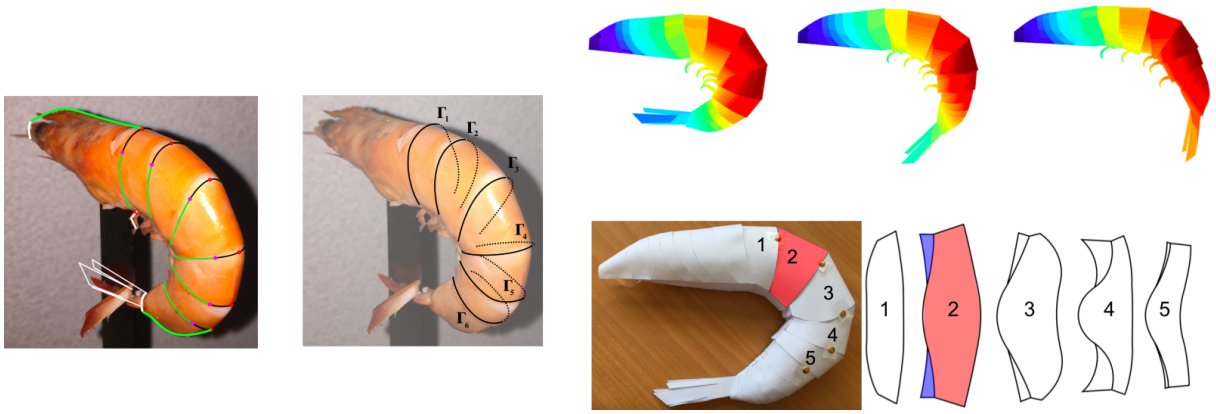


Figure 2.14: *Example of reconstruction of surfaces made of articulated rigid layers from a single view annotated sketch. From left to right: Annotation on top of the inspiring picture; Reconstructed curves separating the articulated rigid pieces; 3D virtual result (top), and real fabrication using 2D patterns.*

Finally, extending the reconstruction approach to other types of surfaces could also be developed. In this sense, we already started to investigate the generation of articulated objects made

by overlapping rigid layers which are tedious to model using standard tools. Such object may represent for instance animals with shells and armors, and exhibits common characteristics such as symmetry and cylindrical sections. Using our single view sketch methodology in annotating on top of existing pictures symmetrical data and circular contours of the rigid layers enable to easily 3D models that can be articulated, and possibly fabricated using the resulting patterns as illustrated in Fig. 2.14.

2.2 Deforming complex 3D models. Application to CG.

As explained in the introduction, the development of 3D modelers, computer power, and 3D acquisition systems, allows to model highly complex 3D shapes. These 3D shapes may exhibit structural characteristics such as layers of details with specific organization, or assembly of substructures, which may even be organized hierarchically. Generating such structured shapes can either be performed manually using 3D modelers, or using procedural generations, but in all cases their complexity require time and expertise.

Once generated, these shapes are usually saved, or shared, as a polygonal meshes. Meshes, while allowing efficient visualization, do not explicitly encode the high level structure of the shape, which, in return, is hardly retrievable from the polygonal data. As a result, mesh surface exhibiting underlying structure can be visualized easily, but cannot be modified. Thus, it cannot be reused in different configurations without taking the risk to destroy the global consistency of the shape, spoiling therefore its plausibility. Let us consider for instance the shape of a dragon-like animal with circulars picky scale structures on its back. One could wish for instance to reuse such a given shape, but adapt it in order to elongate its neck. While the global shape of the neck should be elongated, the details spikes should not, as they would be stretched, leading to a non-plausible dragon-like animal. Instead, the details could, for instance, be duplicated, such that their distribution still remain similar to the distribution in the original shape, and thus remaining a plausible shape after deformations. Such application may be of main interest for CG productions where we can, for instance, commonly meet a family of objects, or characters, exhibiting the same general aspect, but with slight geometrical variations, providing them an individual aspect and avoiding to look like a series of *clones*. This application is even more striking when the scene contains a crowd of characters or objects.

We can thus note that generating a 3D model exhibiting an underlying structure requires time and effort, which may be inherent to the complexity of the object. However, we claim that, once generated, the model should be easily deformable in a controllable manner such that it can be adapted in various situations. Then, the essential artistic, or engineering, effort would be spent in defining a given instance of the structure, or in the authoring of the general deformation, but would not be spend in the tedious task of duplication, and details arrangements to obtain a valid deformed instance.

In order to ensure an efficient way of deforming these complex models, we claim that the following properties should be met

- The plausibility of the object, which will be formally defined later by its *consistency*, should be preserved through the deformation.
- The user should be able to edit iteratively the model at different level of details, ranging from local to global scale.
- The user should be able to apply its edit in the order he wishes, and not solely from the standard *coarse to fine* scale.

2.2.1 Complex model

We call *complex model*, a model exhibiting a hierarchical structure. A complex model can be defined recursively as a set of *complex* and *simple* models, where a *simple model* does not exhibit hierarchical structure. The hierarchical structure of the model can be represented as a graph, where the root is a complex model, and the leaves are simple models. Intermediate elements, with parents and children, are other intermediate complex models. While simple models are necessarily associated to a set of parameters, called *internal parameters*, describing their own geometry, complex models may possibly be solely defined by their children, and thus not necessarily having a specific geometry by their own. Note that the hierarchical decomposition of a virtual object is inherently dependent of the semantic applied to the model, and the way we expect this shape to be deformed. Example of *complex models* may be trees consisting of a set of branches, which are themselves made of sub-branches, and so on, until reaching the leaves (see Fig. 2.15). On the other side, a set of disjoint elements such as a crowd of characters, can also be considered as a *complex model*.

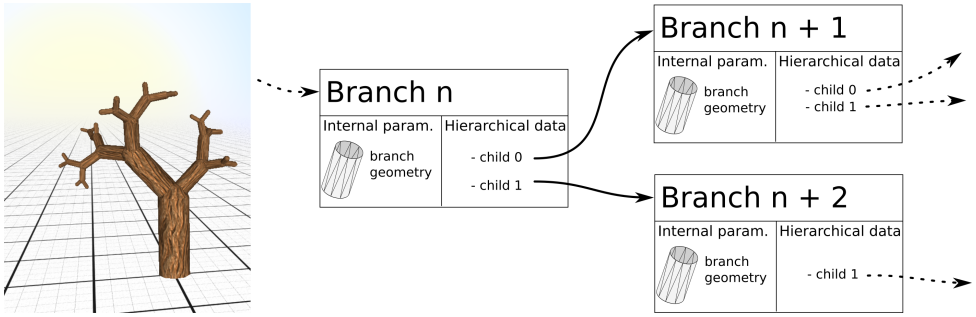


Figure 2.15: *Example of hierarchically defined object. Each level of the hierarchy, which is in this case all branches, contains internal parameters, and hierarchical data defines by their children.*

A complex model is set to be *consistent* if all structures are plausible by themselves, and that all relations between them are also plausible. We will formalize this notion of *consistency* in the specific case of complex object in Chap. 2.2.4.

We provide three contributions associated with such deformation. Two contributions concern a specific deformation allowing to preserve the consistency of the model in the respective case of, first, an organic shape with details, and, second, an object assembly with interconnected parts. Finally, we propose a last contribution formalizing a generic way, called *deformation grammars*, of handling a consistency preserving deformation of hierarchical complex models. Note that in all cases, the hierarchy of the model is provided and is already part of the model description. Computing automatically such structure is out of the scope of our work and we may refer to other works such as [Mitra *et al.*, 2010, Berner *et al.*, 2011].

2.2.2 Deforming a complex organic shape with details

***Note**: Further details on the following part can be found in [Rohmer* et al.*, 2015].*

We proposed a first approach allowing to seamlessly and interactively deform a 3D model, typically representing an organic shape with details exhibiting a specific organization. While the model can be freely deformed by local stretching and squeezing, the details on top of the

surface are automatically adapted such that their positional distribution mimic their original appearance. In order to allow the user to apply seamless deformations, we propose a fully continuous temporal method, meaning that the details will not suddenly appear or disappear on top of the base surface, but be smoothly synthesized or merged while the user manipulate the model.

Non-uniform scaling has been proposed in the literature to allow efficient deformation of objects presenting local high curvature elements in distributing distortion over regions without details [Kraevoy *et al.*, 2008, Dekkers and Kobbelt, 2014], or for man made objects, in order to preserved geometrical properties such as parallelism and orthogonality [Gal *et al.*, 2009]. However, these approaches do not allow automatic details replication in the stretched regions. Other approaches are able to replicate details for regular structures made of primitives such as spheres and cylinders [Owada *et al.*, 2006, Bokeloh *et al.*, 2011], but cannot be applied to organic shape. Interestingly, Alhasim *et al.* [2012] propose to encode the shape details as texture map along a smooth surface described by its linear medial axis, and use texture based approach to synthesize new stochastic details when stretched. This method adapts well to organic cylindrical type of objects with arbitrary high field stochastic details distribution, but does not handle easily non cylindrical underlying shape exhibiting local stretching and squeezing.

In our case, we consider the 3D object which is deformed to be represented in two parts. First, a smooth surface representing the general shape of the object without its details, and second, the set of details. Each detail is supposed to be fully defined by its position and a weight attribute representing for instance its height or width. It is further supposed that the details can be dynamically generated on top of the surface from their position and weight. Note that such object can be represented as a complex model represented by four elements as seen in Fig 2.16.

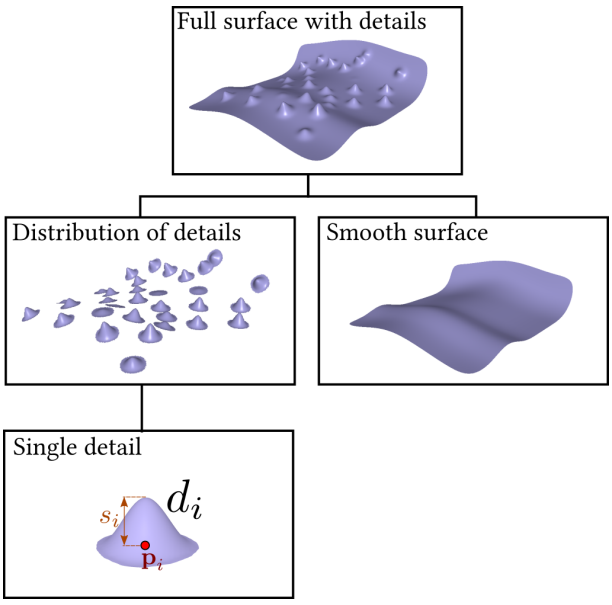


Figure 2.16: *Complex model associated to our structure. A detail surface consists of a distribution of details and a smooth surface without details. Each detail has specific geometrical parameters.*

The approach consists in being able to synthesize new details, or delete existing one, in order to make the pattern of their relative positional distribution looking like the original distribution

before deformation. As a precomputation step, we encode the set of details' positions as a mesh that we call $\mathcal{M}_d$ which can be computed as a Delaunay triangulation from these positions. Each triangle of $\mathcal{M}_d$ represents an area void of details between three close-by details, and is therefore an indicator of the density of details. Note that the mesh $\mathcal{M}_d$, is much coarser than the dense mesh representing the visual surface.

At interactive time, the user apply a deformation on the object, typically modeled as a freeform spatial deformation in our application. This deformation is applied both on the smooth surface and also on the position of the details, which are thus displaced on the surface. This displacement implies a modification of the distribution of details which can be quantified using the mesh $\mathcal{M}'_d$ based on the new position of details as seen in Fig. 2.17.

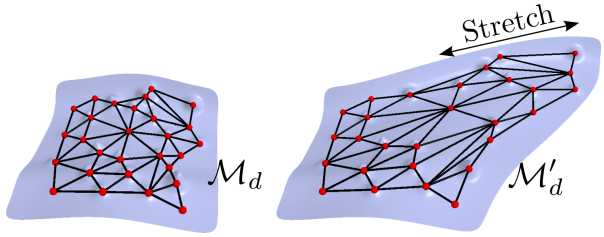


Figure 2.17: *The triangulation based on the details' positions allows to get a coarse indication of stretching and compression applied on the distribution of details.*

Let us describe the algorithm handling the insertion of new details which is illustrated in Fig. 2.18. We consider a stretched triangle $\triangle$ from $\mathcal{M}'_d$, i.e. a triangle with large area compared to its initial one. Our objective is to synthesize new details' positions within this large area in order to recover patterns that can be found in the original shape. The key idea of our algorithm

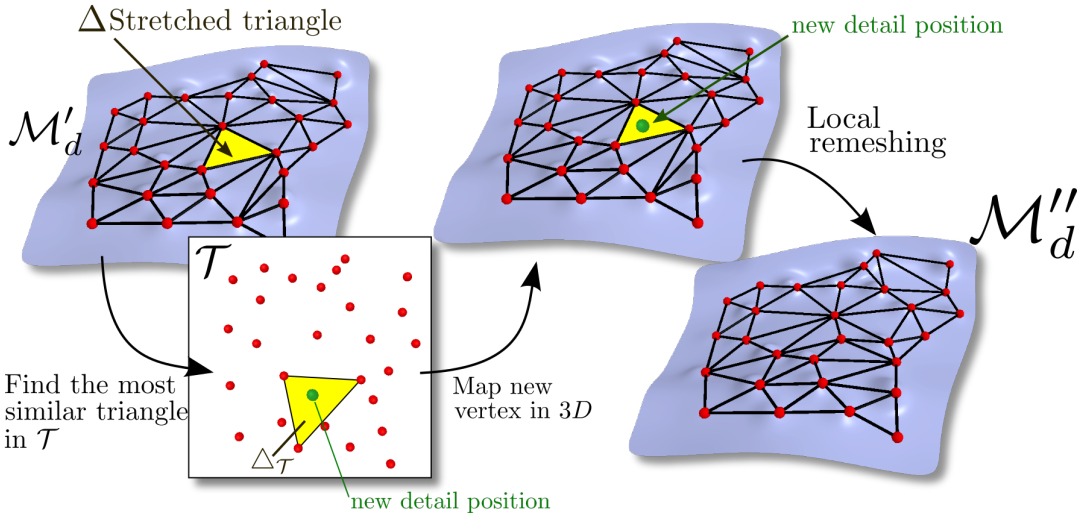


Figure 2.18: *Overview of the insertion algorithm. From left to right: A stretched triangle $\triangle$ is detected. Its most similar triangle $\triangle_{\mathcal{T}}$ is looked for within the set of all possible triangles $\mathcal{T}$. The relative position of all details belonging to $\triangle_{\mathcal{T}}$ is mapped onto the deformed mesh. Finally, a new triangulation of details is generated.*

is to analyze the set, called $\mathcal{T}$, of all possible triangles made by each triplet of detail's position of the original shape. This set of triangles encodes all local pattern of relative position of details' position within the original shape. We then look within $\mathcal{T}$ the triangle, called $\triangle_{\mathcal{T}}$, which is the most similar to $\triangle$, i.e. having the closest area and minimal edge angle. As $\triangle$ is stretched, we expect that $\triangle_{\mathcal{T}}$ covers a domain containing other details. Once $\triangle_{\mathcal{T}}$ found, we compute the relative barycentric coordinate of all details contained within the domain defined by $\triangle_{\mathcal{T}}$, and report these positions within the stretched triangle $\triangle$. They defined new detail positions, which necessarily represent existing pattern within the original shape. Finally, a

new triangulation $\mathcal{M}_d''$ is computed using a local Delaunay triangulation around the newly inserted details. The geometry of each new detail is however set with an initial weighting value $\omega = 0$, such that it doesn't suddenly appear visually. The weight is then smoothly increased toward its final value when the user further interact with the model, ensuring a smooth apparition of the details. Note that the number of triangles of $\mathcal{T}$ can be rather large, i.e. the cube of the number of details, therefore it is critical for an interactive application to be able to compute the similarity measurement between triangle efficiently. In our case, we precompute all area and minimal edge angle for all triangles of $\mathcal{T}$ and cache them in a 2D structured grid sorted with respect to these two parameters. Finding the most similar triangle can therefore be efficiently queried in this precomputed 2D grid structure, independently of the number of details.

We applied our algorithm, which also includes details merging in the opposite case when triangle $\triangle$ undergoes shrinkage, to three types of details. First when details are procedurally generated spikes, second given as meshes, and third, in modeling details as holes in the underlying mesh.

In Fig. 2.19, we show that our approach allows to deform the surface while continuously inserting new details during stretching by a factor of two. Through the deformation, the maximal change of the average edge length of the triangulation $\mathcal{M}_d$, thus quantifying the change of distance between two close details in the distribution, is about 10%. An example of a complete mesh deformation is also illustrated in Fig. 2.20 where a frog exhibiting details on its back is first elongated, and then inflated.

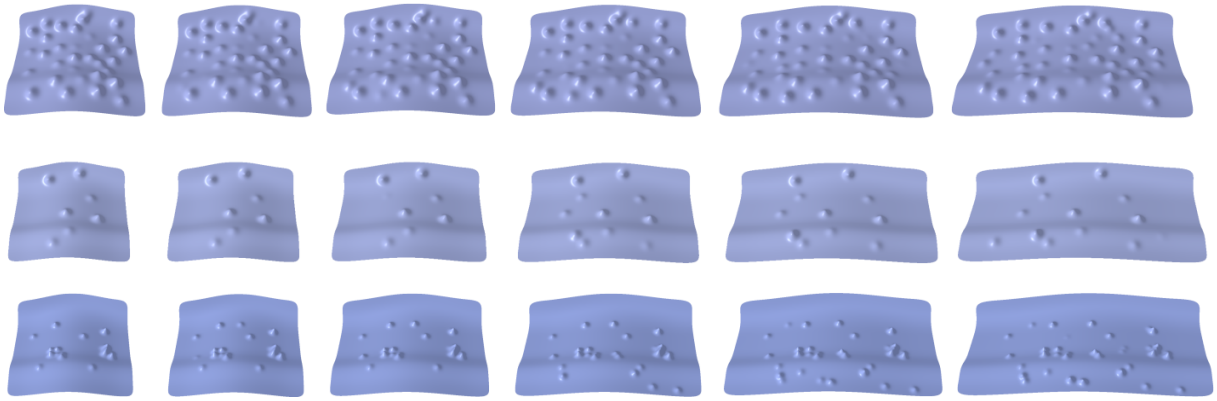


Figure 2.19: *Applying continuous stretching on, respectively, uniform distribution with high detail density (top), low detail density (middle), and non uniform distribution (bottom).*

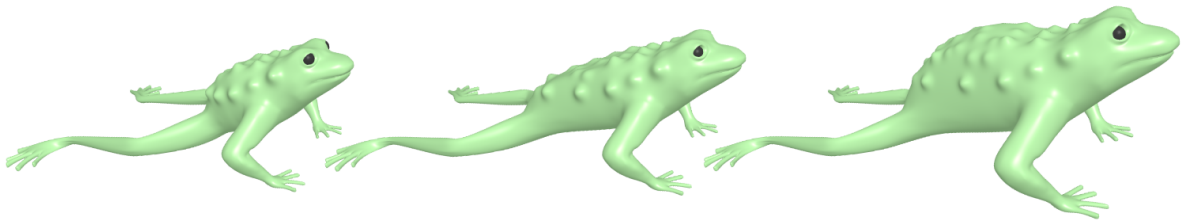


Figure 2.20: *Elongating and inflating the back of the frog model is associated to an automatic and seamless replication of details.*

2.2.3 Deforming and synthesizing new assemblies of interconnected parts

***Note**: Further details on the following part can be found in [Liu et al., 2015].*

A second specific type of deformation we tackled focuses on the deformation, or more precisely on the synthesis, of assemblies with interconnected parts. These assemblies, representing for example toys models made of rigid elements that can be assembled using a specific set of connectors. We developed an algorithm able to model new shape from examples [Funkhouser *et al.*, 2004], or more precisely, to automatically propose plausible shape variations from a given set of structures in analyzing their connection graphs, and synthesizing new ones.

Let us consider a set of $N \geq 1$ model assemblies, where each assembly is made of individual elements having a specific nature as illustrated in Fig. 2.21. Each element of a given nature is connected to other elements with similar or different nature. An element is said to be consistent with respect to their nature if is exhibits a connection with other elements that can be found in the initial assembly. A new assembly is thus consistent if it is fully composed of consistent elements.

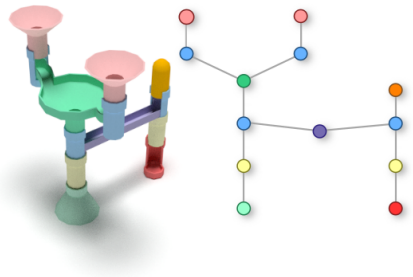


***Figure 2.21:** Example of complex model made of several elements with a specific nature. Each element and its connection can be represented as a graph.*

In order to synthesize a variety of consistent structure, we introduce the notion of *replaceable structure*, extending the notion of single *interchangeable component* from Kreavoy *et al* [Kreavoy *et al.*, 2007], as a subset of connected elements that can be consistently be replaced by another subset. Let us denote G_i the topological graph indicating the connection between the elements of the $i^{mboxth} \in [0, N-1]$ model. Each node of the graph G_i represents an element and is tagged by its nature, while edges represent the connection between elements. The objective of our approach is to synthesize a new graph G' such that subparts of elements of a graph G_i are replaced by another from G_j, while preserving the consistency, i.e. the elements have connections that already exist in the set of original models as illustrated in Fig. 2.22.

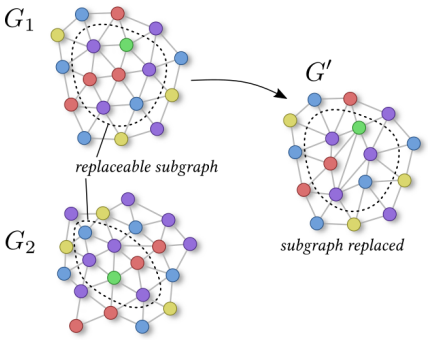


***Figure 2.22:** Two input graphs G_1 and G_2 are provided, and exhibits compatible subgraphs. The resulting graph G' is obtained in replacing the subgraph in G_1 by the one from G_2.*

While computing in general all subgraphs match is a NP-hard problem [Demaine and Demaine, 2007], we proposed an efficient method to extract a replaceable structure in our context. The main idea consists in working on the dual graph of G_i, thus explicitly encoding the connections as a node. In order to further reduce the complexity, we also assume a local ordering constraints on the connections around an element. Moreover, our approach do not aim at guaranteeing an exhaustive coverage of all subgraphs.

Once a new graph G' is computed in replacing the subsequent substructure, a geometrical embedding is computed using rigid transformation in order to visualize the resulting object. Finally, some global constraints are checked, such as ensuring that the supporting pieces are touching the floor, and that model requiring one entry and one exit are preserving such property. An example of result obtained on a child's game model using a single input model, and generating four output variations is illustrated in Fig. 2.23. In this example, only assemblies with one entry, and one or more exits are kept as valid. Another example of rolling toy illustrate the possibility to combine different models in Fig. 2.24. In this case, four different simple inputs are provided, and subsets of elements are combined in order to generate 16 new variations.

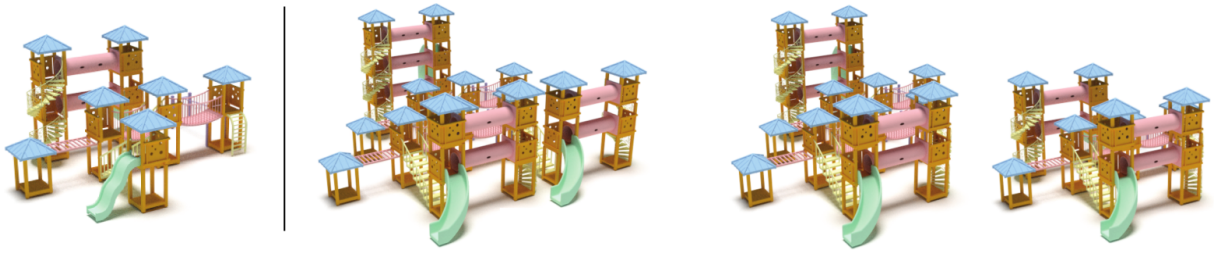


Figure 2.23: *Left: Original input model. Right: Three variations synthesized using replaceable structures.*

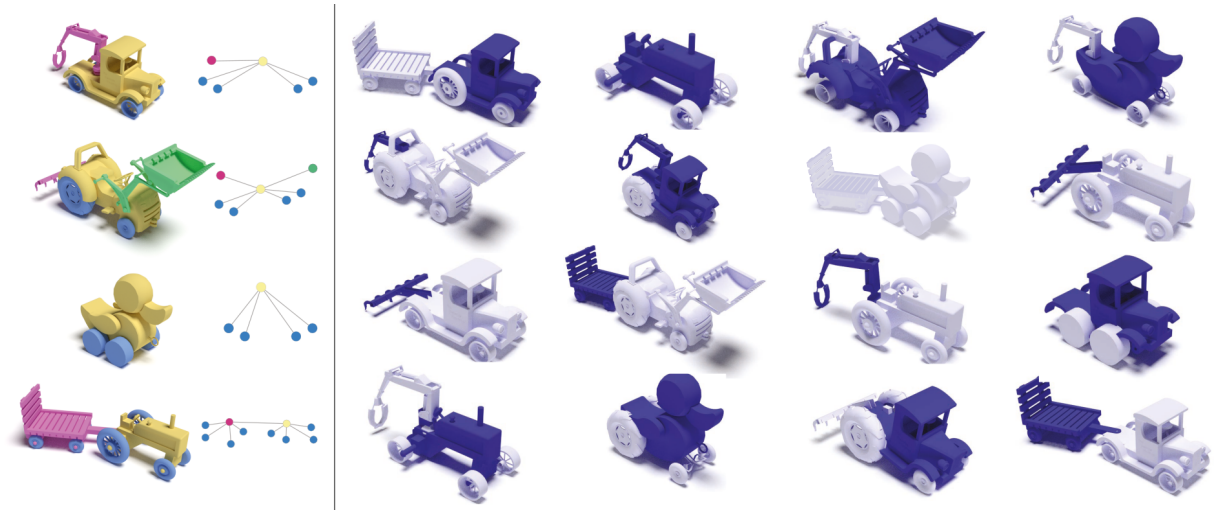


Figure 2.24: *Left: Four input models with common replaceable structures. Right: Resulting variations obtained in switching compatible parts between different input models.*

2.2.4 Deformation grammars

Note*: Further details on the following part can be found in [Vimont et al., 2017].*

Our third contribution consists in proposing a general approach to handle deformation on top of complex models. While several methods have been developed to deform specific complex objects using, for instance, procedural generation with dedicated parameters [Lipp *et al.*, 2011, Longay *et al.*, 2012, Barroso *et al.*, 2013], deformation preserving a given

a-priori geometric property [Sorkine and Alexa, 2007, Stanculescu *et al.*, 2011], or property extracted from data [Zheng *et al.*, 2011, Mitra *et al.*, 2013], there is no general framework handling all cases. As a result, deforming a new complex object while preserving a specific consistency, usually requires to re-develop an entire system dedicated to the specific object.

Our objective was to develop a generic and formal description able to handle consistent preserving deformation for arbitrary complex objects. We inspire from the methodology developed for the shape grammar [Stiny and Gips, 1972] describing a generic way of synthesizing geometry adapted to hierarchical structure. In our case, we however adapt the grammar formalism to handle deformations. We thus called our approach *deformation grammar*. Deformation grammar is a formalism applied on a hierarchical structure ensuring that an arbitrary input deformation, such as generic spatial deformation, can be interpreted into another deformation which preserves the global consistency of the shape on which the deformation is applied.

Consistency of a complex object

We first formalize the notion of consistency in the case of complex model. We call a simple model to be *consistent* if the set of *internal parameters* describing its geometry satisfy some criteria essential to its plausibility. For instance, the length between two positions must be given by a specific value, a shape must be spherical, or cylindrical. As simple models can be fully defined by their internal parameters, i.e. independently of all the other elements, we call this consistency to be the *internal consistency*. Complex object, on the other hand, is described by the relation between different sub-elements. Its own consistency thus depends on the consistency of its children, and the relation between them. Example of such dependency may be that sub-branches of a given branch of a tree should not intersect each other, a set of character should be evenly distributed, walls of a house should be orthogonal, or as illustrated previously: details distribution exhibiting a given pattern, and assembly of connected sub-elements having a specific nature. Thus, in addition to the *internal consistency*, we define the *hierarchical consistency* as the set of parameters that must be satisfied for a hierarchical relation to be plausible. A complex object is thus consistent when it satisfies its own *internal consistency* and its *hierarchical consistency*.

The problem of allowing a deformation to be applied on structured object can thus be reformulated as finding a deformation which preserves the *internal consistency* and *hierarchical consistency* of a complex model.

Deformation grammar for global deformation

The general overview of our method is based on the notion of deformation propagation through the hierarchical tree of the object as illustrated in Fig. 2.25. First, the user generates an objective deformation $D(\epsilon_{init})$ on the element ϵ_{init} that is supposed to be the root of the hierarchy, i.e. $D(\epsilon_{init})$ is a global deformation. As applying directly $D(\epsilon_{init})$ onto ϵ_{init} would, in general, destroy the consistency of the model, we modify $D(\epsilon_{init})$. This modification, called *interpretation* in the grammar formalism, generates two new deformations $D_{internal}(\epsilon_{init})$ and $D_{hierarchical}(\epsilon_{init})$ such that $D_{internal}(\epsilon_{init})$ preserves the *internal consistency* of ϵ_{init}, and $D_{hierarchical}(\epsilon_{init})$ aims at preserving the *hierarchical consistency*. Indeed, the hierarchical consistency may be recursively defined, and thus the deformation $D(\epsilon_{init})$

must be *propagated* to the children of ϵ_{init}. This deformation is then interpreted into $D(\epsilon_i)$, where $(\epsilon_i)_i$ are the children of ϵ_{init}. The process is iterated until reaching the terminal elements of the hierarchy. Note that each interpretation, i.e. modification of the deformation at a given level of the hierarchy enabling to preserve the internal or hierarchical consistency, is fully dependent on the model, and must be pre-defined by the user, using standard script code in our case.

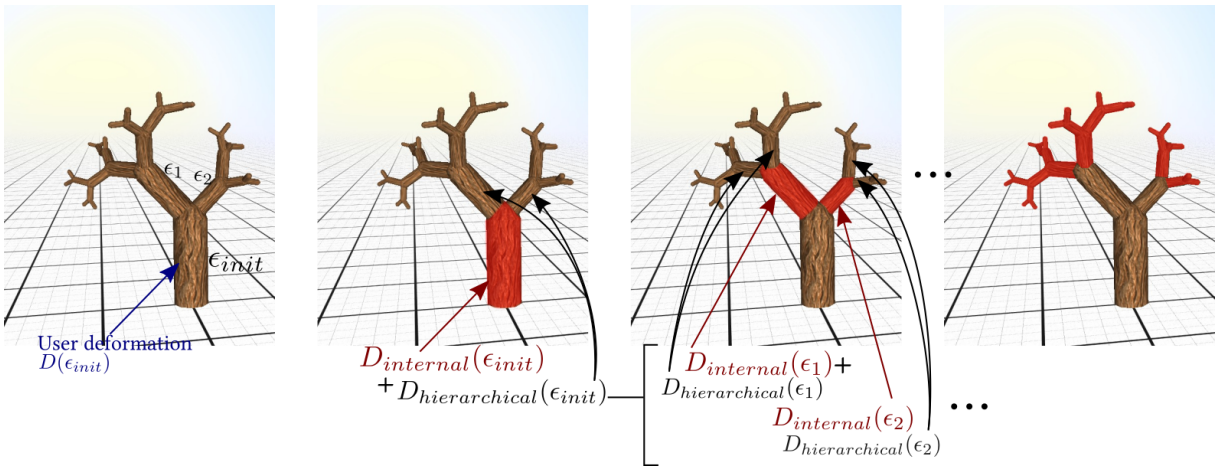


Figure 2.25: *Example of deformation propagation within the hierarchy of the shape. The initial user deformation is propagated recursively and interpreted at each step to preserve the internal and hierarchical consistency.*

Handling local deformation

Let us now consider the general case where ϵ_{init} is not the root of the hierarchy. The top-to-bottom algorithm allows to ensure the consistency starting at ϵ_{init}, but may not be consistent with respect to the parent of ϵ_{init}. For instance, a local deformation applied directly to a branch of a tree may result in the disconnection of this branch with respect to its trunk as illustrated in Fig. 2.26.

We therefore propose to propagate deformation upward within the hierarchy until reaching a specific element limiting the locality of the deformation, or in the worst case until the root. Specific rules are set depending on the origin of the deformation in order to avoid looping between upward and downward propagation.

Finally, persistent edition, meaning that a local user edit will not be erased by a subsequent more global edition can be encoded within this method. To do so, we tag each element that has been directly edited by the user, and forbid them to be modified by a deformation originated from a higher hierarchy level.

Results

Our approach has been fully formalized using grammar inspired notations in [Vimont *et al.*, 2017], and we implemented examples of hierarchical shape deformation for the several scenarios illustrated in Fig. 2.27 such as surfacic and volumic set of elements preserving a quasi-uniform distribution [Stanculescu *et al.*, 2011], structured building preserving orthogonality between walls and distribution of windows, trees preserving

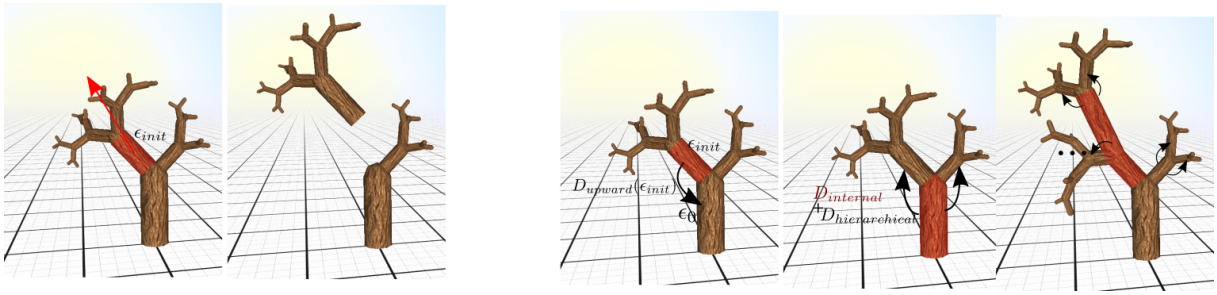


Figure 2.26: *Handling local deformation thanks to ascending propagation of deformation. Left: A user translation is applied to ϵ_{init} which is not the root of the hierarchy. Without the use of ascending rule, the trunk cannot ensure that the child branch remains connected to it, leading to a possible discontinuity. Right: The deformation applied to the branch is first propagated upward the hierarchy to the trunk, which, then propagates it downward to ensure the consistency preservation through all the steps. Note that in this case, we implement a subdivision of branches when its length exceeds a given threshold as consistency property, leading to the creation of this new branch on the left side of the tree.*

average branch lengths, and scene mixing these previous elements while handling arbitrary spatial user deformation as input.

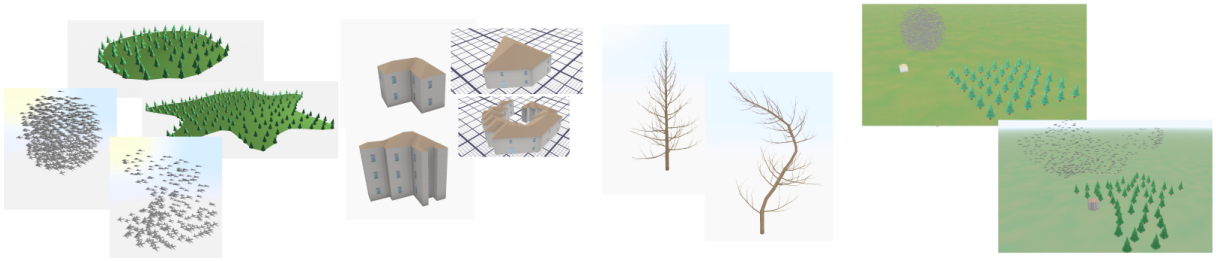


Figure 2.27: *Various results showing the initial, and the deformed shape. Note that in all cases, a generic spatial deformation is applied by the user using an interactive sculpting tool. These deformations are interpreted using our grammar formalism to specific object deformations to preserve the consistency on the underlying shape.*

2.2.5 Discussion

We presented different approaches able to handle complex shape deformation.
The first approach allows to seamlessly deform a detailed 3D consisting in a smooth underlying surface with point-wise distributed procedural details that adapts locally to the deformation. This approach had the advantage of allowing continuous deformation such that new details smoothly appears and disappears, leading to a more direct and natural way of controlling a virtual deformation, where the user can concentrate on the general high level deformation. The approach, designed to allow very fast deformation computation, remains however limited with respect to its plausibility to homogeneous pointwise details distribution. In order to handle more complex details distribution, we already started to investigate the use of histogram measurement that we extended from 2D cartesian distance to curvilinear one on manifold surfaces. This approach, although costlier to compute, allows to capture and preserve distance correlation between the set of details positions that can present structured groups and clusters.
Our second approach allows to synthesize new variations of shape assemblies from existing ones. The plausibility on the topological level is handled via an associated labeled graph indi-

cating possible connection between elements, and must be further checked with respect to the geometrical embedding. The range of possible objects spans a large class of man made objects made of assemblies. The approach, however, doesn't allow direct user control, and the computation cost associated with the extraction of compatible subgraphs remains high. Extending such approach toward more user guided control, or to other types of objects such as organic ones could also be investigated.
Finally, we proposed a generic approach to handle deformation of hierarchically defined objects in decoupling the constraints applied on the local, non hierarchical parameters, to the constraints applied onto the hierarchy, under a grammar formalism. The approach is shown to be very general in the case of hierarchical objects, and other standard existing deformation approaches can be reformulated within this formalism. Note that shape grammar provides the generic structure to follow in order to propagate the deformation through a given hierarchy. However, it doesn't provide an automatic way of computing the hierarchy of the model, nor the interpretation of the deformation for a given level and type of deformation. These steps remain potentially complex to develop. In our case, we believe such definition must be developed in two stages. First, the final user, i.e. commonly an artist, must define precisely the way he wants to deform the complex model in order to design the hierarchy and its consistency. Secondly, a technical scientist must implement as code, or at least script, the necessary functions able to interpret the given deformations for the various elements of the hierarchy. Automatizing such approach from data using, for instance, inverse procedural modeling [Aliaga *et al.*, 2016], would be a nice extension to generalize a general an automatic method to handle complex model deformation.

In a general way, our methodology allows to reuse existing shape to generate new ones, instead of synthesizing each one from scratch. With the growing demand of 3D detailed shape, we believe that the general idea to be able to reuse and adapt already existing shapes interactively will allow to increase efficiency of 3D modeling, and could further enhance the value of shape sharing through the community.

2.3 Modeling embryo development. Application to medical sciences.

***Note**: Further details on the following part can be found in [Rabattu et al., 2015].*

A third application where we have applied 3D modeling is medical sciences, and most specifically to the modeling of embryo development. During this development, the embryo undergoes large 3D deformation, involving both geometrical and topological modifications, starting from the cells structure into complex model with interacting organs. These fascinating 3D deformations are however hard to understand in their globality. Traditional anatomical books which are used by medical students describe these transformations at specific time steps using text and 2D drawings. The following limitations can however be noted. First, the general knowledge of the embryo development is spread through several books and may exhibit heterogeneous descriptions. Second, the time varying development can only be described by static frame, and the actual deformation must be inferred by the student. Third, the description by 2D images does not allow to rotate the model in order to better apprehend its 3D structure. To tackle these limitations, our insight is to mix the knowledge structuration into a unified ontology including a spatio-temporal description to allow 3D animated visualization.

Ontology is formal way of representing knowledge, where objects and concepts can be made in relations [Gruber, 1993]. Ontology may physically be stored as data bases and queried in order to access to this structured knowledge, and automatic reasoners may be used when the ontology uses OWL [Sirin *et al.*, 2007] or RDF description which is widely used in the Semantic Web. Ontologies have been successfully developed in medical science for their ability to handle the complex relationship between biological entities [OBO Foundry, NCBO Bioportal]. The Foundational Model of Anatomy (FMA) [Rosse and Jr, 2003] describes the reference structural organization for human anatomy. However, ontologies have, so far, being restricted to conceptual and textual description. Attempts have been made to include visual representation in linking medical 3D images to the data such as in the case of the mouse embryo [Brune *et al.*, 1999] or within the anatomical ontology MyCF [Palombi *et al.*, 2014] including 3D representations linked to anatomical entities. Still these works limit the link between the ontology and the 3D data to external links. At the opposite we proposed the first fully integration of animated 3D anatomical models within the elements of the ontology.

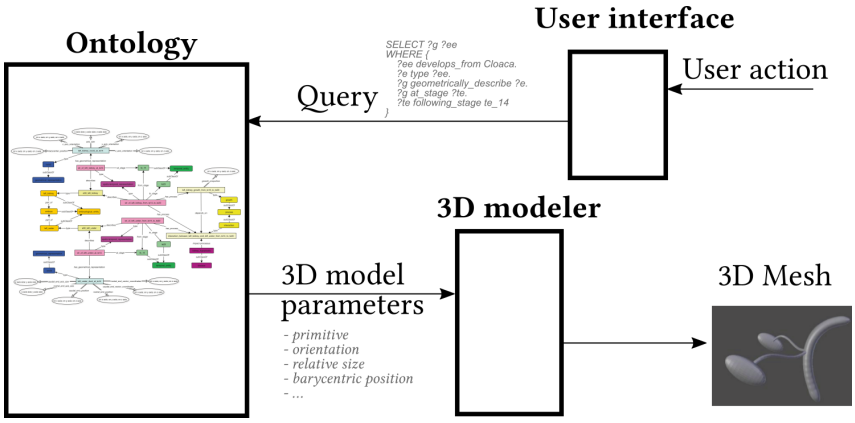


***Figure 2.28:** Overview of our interactive ontology based visualizer. The user action, such as time set, or pathology selection, triggers a query interpreted onto the ontology, and result in a set of vectorial parameters which are used to synthesize a 3D mesh.*

The general idea consists in inserting within the ontology a high level vectorial description of the geometry and their temporal events, in order to be able to procedurally synthesize the 3D model in querying the ontology. The general process of our approach is described in Fig. 2.28.

2.3.1 Spatio-temporal ontology

Our ontology contains six types of entities. First, the embryological entity describes anatomical structure as RDF classes, such as for instance left/right kidney or ureter (Fig. 2.29 top). These entities have properties such as *part of, has part, develops from* describing the relation between the structures. Second, the temporal entities refers to the Carnegie stages, which are specific number of days from the gestation. Third, the geometrical components are describing the shape of an organ. We handled six types of basic geometrical primitives, namely point, line, plane, ovoid, cylinder, and duct. Each geometrical component contains its internal parameters, as well as its embedding within the embryo given by a barycentric position, axis orientation, and scale proportion (Fig. 2.29 middle). Fourth, the spatio temporal entities are linking geometrical components to temporal entities (Fig. 2.29 bottom). Similarly to key frame of animations, they are the central element of the ontology as they provide the geometrical description at a specific time. Examples of spatio temporal entities are for instant left kidney between day 14 and day 20. Fifth, the process models the deformation between temporal entities. We handle six processes, namely growth, migration, rotation, interaction, division, fixation.

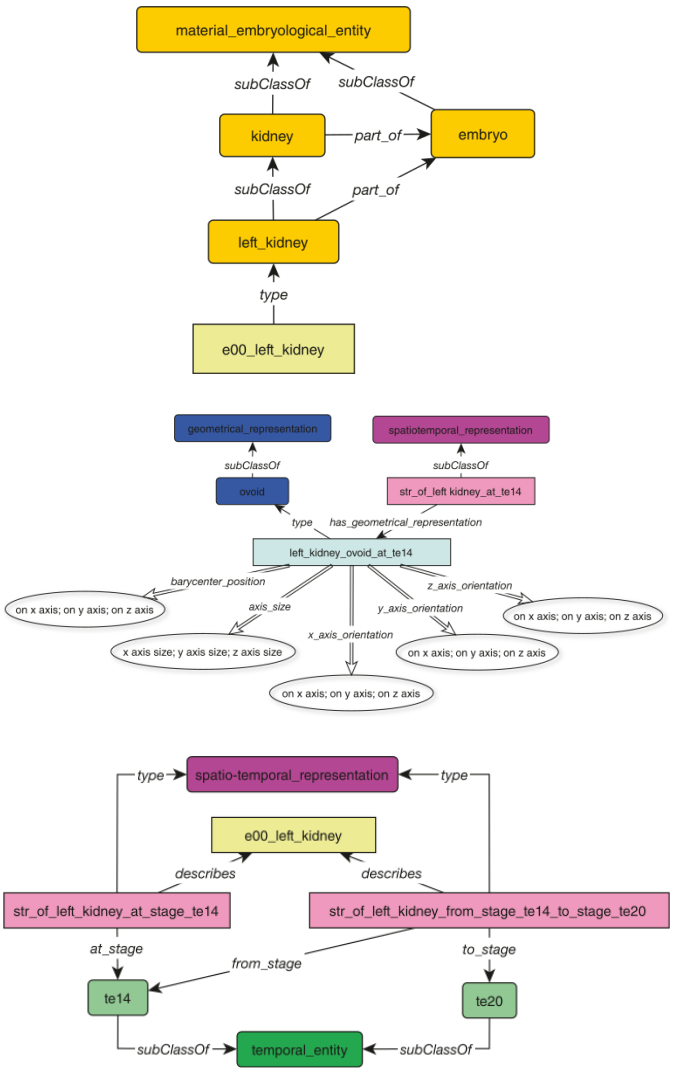


Figure 2.29: *Three types of entities and relations of our ontology representing respectively, from top to bottom, the anatomical structure, the geometrical component, and the spatio temporal entities.*

2.3.2 3D development visualization

We developed the ontological data base for the development of the urinary system represented by the set of simplified described primitives allowing to visualize on demand the growth and migration of the kidney and ureters between stages 13 to 27. The visualization of some steps are illustrated in Fig. 2.30 and are obtained from a procedural generation developed in Blender scripts interpreting result of the query onto the ontology. The geometry and the animation is

fully driven by the ontology.

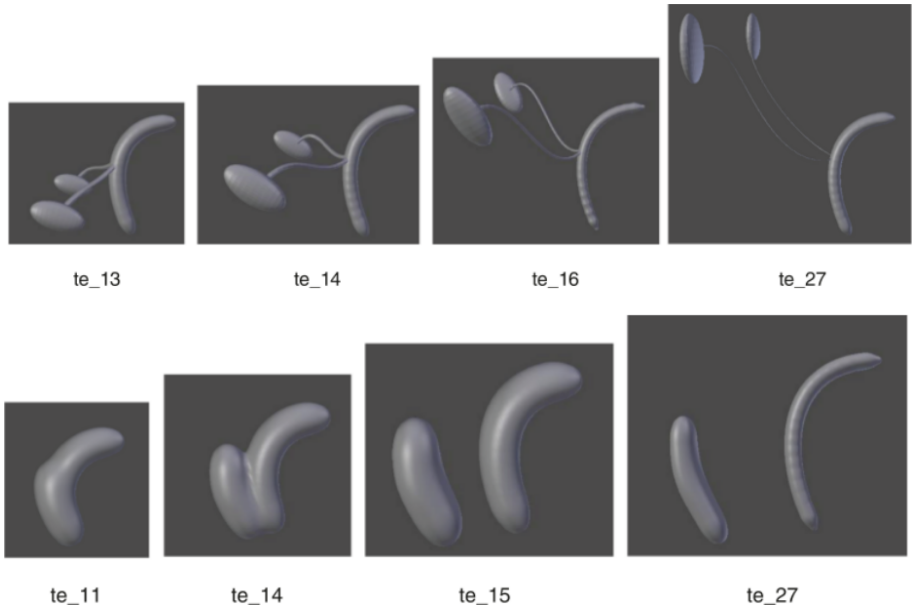


Figure 2.30: *Top: Evolution of the migration, rotation, and growth of kidneys and ureters. Bottom: Modelization of the cloaca division process.*

We further introduced a sixth class in the ontology modeling disease. Disease describes the activation of inactivation of some processes, and thus allows to visualize the resulting effect on the geometry (see Fig. 2.31).

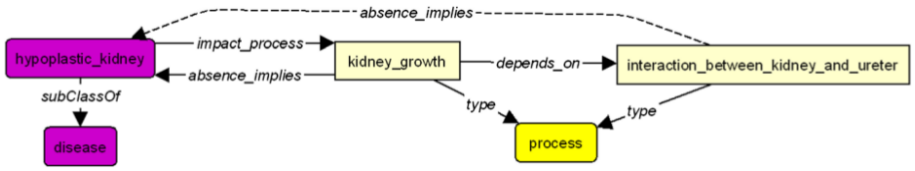


Figure 2.31: *Modeling pathology through the relation between organs.*

2.3.3 Discussion

We proposed a modeling approach allowing to synthesize a coherent surface model from an ontology, used as a generic way to handle complex models. At the opposite of the other procedural modeling approach, the generation of the mesh surface is not encoded as code, or script, but within an ontology data base encoding the parameters of the model and its behavior. This ontology can be queried, and reasoners can be applied to it, in order to deliver as output, geometrical parameters allowing to synthesize a 3D model.

At the current stage of this project, only simple and coarse procedural geometry was handled. Indeed, the resulting shape appearance doesn't compete with dedicated organs 3D models generated by hand, however, our approach can be seen as a first proof of concept, and open the door to a more complex modeling approach.

First, the coarse geometry could be further used to trigger a more detailed geometry. Approaches such as shape embedding [Garcia *et al.*, 2013] or directable simula-

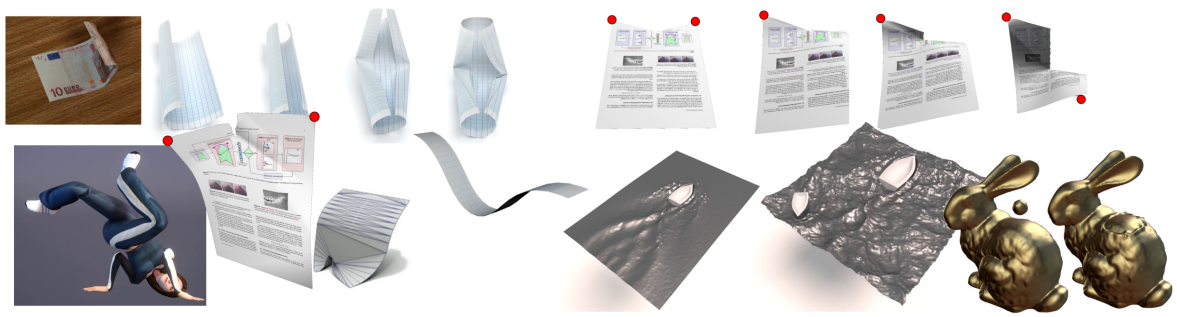
tion [Bergou *et al.*, 2007] could be used to mix coarse volume defined by the ontology with a more details local surface model. Secondly, the use of the ontology allows several possible avenue of future researches. Ontologies are well suited to handle large data base of interconnected components. This can be of main interest to integrate increasingly detail description requires to model human anatomy, but could be extended to other types of applications requiring graphics such as biological models, or CAD modeling. While the ontology may contain a very deep level of details, the interactive query, will only target a subset of it. Thus, a single centralized ontology system could be used to express various levels of details to different applications such as anatomy teaching, medical visualization, or even pathology studies when coupled to a physically based simulation. Note that this selection of adequate level of details can be performed automatically on the ontology thanks to its coupling with resonators. Interaction with ontology based model is also an interesting area of research. Indeed, advanced user interaction will require to trigger textual queries rather than manipulating directly surface parameters. Such development, with strong connection with web technologies, raises new challenges, but may allow in the future higher level interactions for shapes and their dynamic behaviors.

CHAPTER 3

3D models for animation

We present, in the second chapter, different interactive models adapted to 3D animation. Due to the addition of an extra time dimension, designing virtual animation is even more challenging than static shape design. The need for automatic tools to ensure plausible result, and interactive feedback, is thus even more striking.

We propose to ease the generation of virtual model animation, in enriching existing animated model with higher level parameters allowing to better preserve their plausibility, while keeping, or even increasing in some cases, their computational efficiency. Therefore, our animated model, does not only consists of a single low level polygonal surface model, or to a set a physical parameters, but actually contains more information about the nature of the object to be deformed. The interest of our enriched model relies on mixing several representations in a hybrid manner, such as for instance mixing surface and volume representation in order to improve the shape appearance and to handle collision, physically based model with generalized cones allowing to animate developable surfaces, or in adding higher level spatio-temporal representation on top of a set of meshes to extend sculpting metaphor to animated scenes.



3.1 Virtual characters: Improving skinning deformation.

***Note**: Further details on the following part can be found in [Vaillant et al., 2013].*

Character animation is a major part of computer animation requiring advanced authoring system to model various characters poses in virtual scenes. The most popular technique to animate character is the geometric skinning method. Skinning consists in embedding an animation skeleton within the 3D shape, i.e. a set of frames that can be interactively rotated and acting as proxy elements modeling the articulation of the character. Each frame is influencing locally the shape via the so called skinning weights. The deformation acting on a vertex of the shape is then computed as a linear average of all influencing weighted frames, leading to the so called linear blend skinning (LBS) [Magnenat-Thalmann *et al.*, 1988, Lander, 1999]. LBS is widely spread in animation software thanks to its generality and high computational speed, but suffers from strong artifacts under large rotation thus limiting its plausibility. The first one, called *candy wrapper* is associated with the collapse occurring when twisting a cylindrical geometry around $180°$. The second, called *collapsing elbow* relates to the shrinkage occurring at large deformation, usually visible on the elbow of the characters. Finally, skinning doesn't prevent surface intersection which may result in self-colliding model during deformation.

Several attempts have been made to improve skinning deformation. Examples based approaches [Lewis *et al.*, 2000, Kry *et al.*, 2002, Mohr and Gleicher, 2003] are able to limit artifacts, but requires artist's designed example poses. Enriching skinning with physical simulation [McAdams *et al.*, 2011, Kavan and Sorkine, 2012, Gao *et al.*, 2014] also enable to improve the plausibility of the result, but doesn't reach real time result. Dedicated improvements to preserve the shape volume [von Funck *et al.*, 2008, Rohmer *et al.*, 2009, Angelidis and Singh, 2007] were also explored but doesn't solve the other artifacts. Considering non linear blending has also been explored using log-matrix [Alexa, 2002], spherical blending [Kavan and Zára, 2005], and most notably using dual quaternion [Kavan *et al.*, 2007] which has been successfully integrated in standard animation software. Dual quaternion solves the *candy-wrapper* artifacts while allowing efficient computation time, but at the opposite of the LBS exhibiting shrinkage, tends to inflate articulated regions, and doesn't prevent self-intersection.

3.1.1 Mixing skinning deformation with implicit modeling

Our approach aims at solving these two remaining issues as an extra correction step on top of a LBS or dual quaternion skinning. The general idea consists in embedding the deformed skinned mesh within a smooth scalar field approximating the mesh as an implicit surface. This scalar field is built from different pieces approximating the rigid parts of the surface and blend together. During the animation, each scalar field piece is rigidly deformed in following the skeleton animation, and thus allows to synthesize a new global field. However, thanks to the property of implicit blending operators, the global field doesn't undergo the same artifacts as the skinned mesh, notably the articulated regions don't bulge nor shrink. This allows to correct the skinned surface in projecting the vertices of the mesh in the direction of the isosurface following the gradient of the field function, thus correcting extra bulging or shrinkage.

We tackled two technical challenges related to implicit modeling in this work. First, we proposed an efficient approximation of a mesh with an isosurface using Hermite Radial Basis

Functions (HRBF) [Macêdo *et al.*, 2011] which interpolates positions $\mathbf{p}_k$ and normals $\mathbf{n}_k$ from a set of N samples vertices of the mesh surface. This field function F at position $\mathbf{p}$ is computed as

$$F(\mathbf{p}) = \sum_{k=1}^{N} \Big(\alpha_k \, \phi(\|\mathbf{p} - \mathbf{p}_k\|) + \boldsymbol{\beta}_k \cdot \nabla\phi(\|\mathbf{p} - \mathbf{p}_k\|) \Big) \,, \tag{3.1}$$

where we considered the radial basis function $\phi(x) = x^3$, and the α_k, and $\boldsymbol{\beta}_k$ are computed as the solution of a linear system such that $F(\mathbf{p_k}) = 0$, and $\nabla F(\mathbf{p_k}) = \mathbf{n}_k$.

Second, the isosurface represents a low resolution of the mesh surface, thus projecting the vertices exactly onto the isosurface would destroy the details of the original surface. We solve this problem in encoding for each vertex the exact value of the field function in the original pose. Each vertex having possibly a different value, encoding their *offset* with respect to the low resolution isosurface. Then, at run time, the vertices are projected along the gradient of the field function until reaching their respective stored value using the iterative scheme

$$\mathbf{p}^{i+1} = \mathbf{p}^{i} + \Delta L \;\; \big(F(\mathbf{p}^{i}) - \mathrm{iso} \big) \;\; \frac{\nabla F(\mathbf{p}^{i})}{\|\nabla F(\mathbf{p}^{i})\|^2} \,, \tag{3.2}$$

where $\mathbf{p}^i$ corresponds to the i^{th} iteration applied on a position $\mathbf{p}$ chose to be a vertex of the skinned mesh, ΔL is a tradeoff parameter between speed and accuracy, and iso is the vertex dependent isovalue stored in the rest pose.

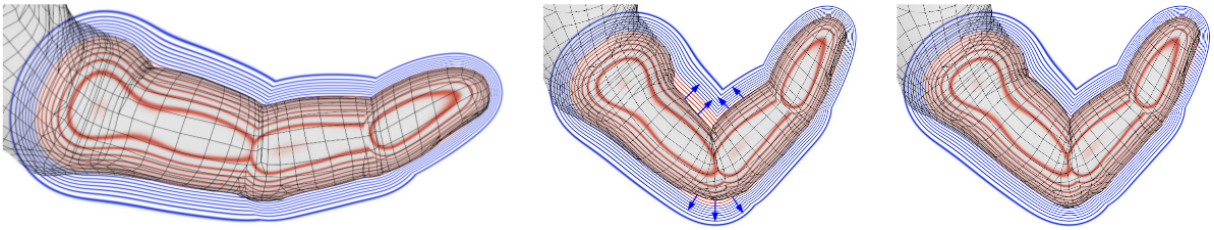


Figure 3.1: *Left: the surface representing a finger is embedded within a field function. Each vertex of the mesh is associated to its exact field value in space. Middle: The mesh surface is deformed using standard skinning, while the field function is deformed piecewise rigidly, and each piece is blended using implicit operators. Right: Each vertex of the mesh is projected onto its original field function value, correcting skinning artifacts.*

As the projection of the vertices along the field function gradient is performed as an iterative marching algorithm, it is the most expensive operation to perform at run time, but can be implemented in parallel on the GPU in storing the field function within a low resolution grid structure. A positive side effect of this iterative projection process consists in stopping the marching algorithm when reaching a gradient discontinuity of the field function -detected as a large angle between consecutive gradients directions-, representing a contact surface. Then, for not supplementary computation costs, the method can seamlessly prevent self collision of the surface at the vertex level. Another positive aspect of the use of implicit model is related to the choice of blending operator between the fields generated by each part of the mesh. Using union [Ricci, 1973], gradient-based blending, or gradient-based bulge-in-contact [Cani, 1993, Gourmel *et al.*, 2013] allows to model different types of skin deformation at articulation varying from sharp union, to local bulging that mimics for instance tissue inflation in bent fingers.

We illustrate in Fig. 3.2 our results in comparison of LBS and dual quaternion. Our implicit skinning approach allows more plausible articulation in better preserving the rigid aspect of the character knee, while preventing self collision.

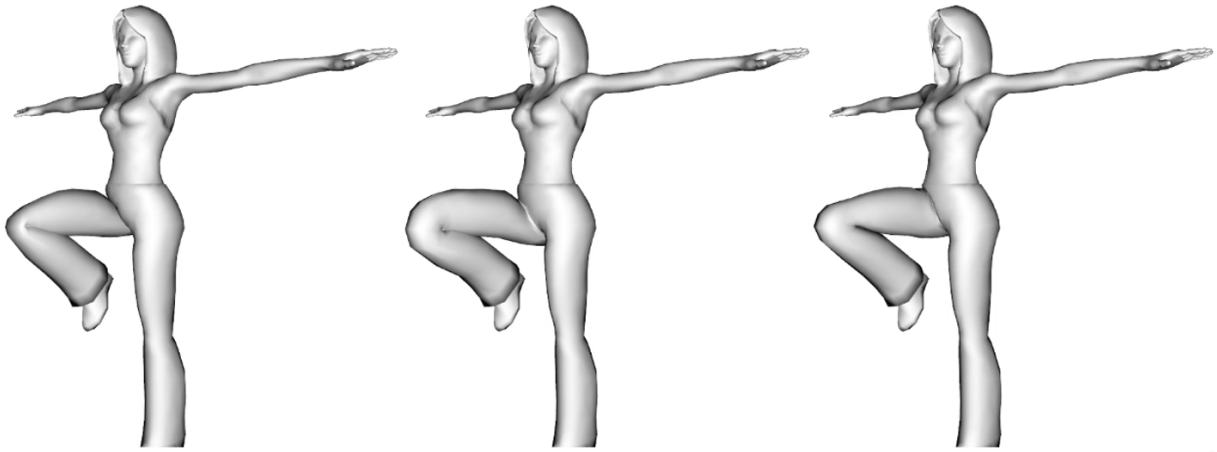


Figure 3.2: *Comparison between skinning based on LBS (left), dual quaternion (middle), and our implicit skinning result (right).*

We further illustrate in Fig. 3.3 the possible control on the visual appearance in the articulated region using various implicit blending operator able to smoothly vary between sharp union to local bulgings.

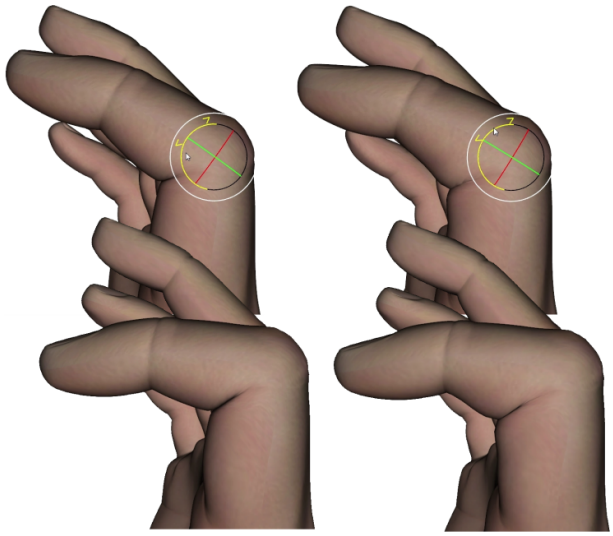


Figure 3.3: *Left: Our result using union operator for the field functions. Right: Our result using gradient-based bulge-in-contact operator, note the bulge appearing on the finger.*

Finally, we show more complex examples of full character deformation in Fig. 3.4 demonstrating that the implicit skinning approach can robustly handle complex character deformation. In all examples ranging between 4000 to 30000 vertices, our GPU implementation run at least at 15 FPS, thus allowing interactive animation.

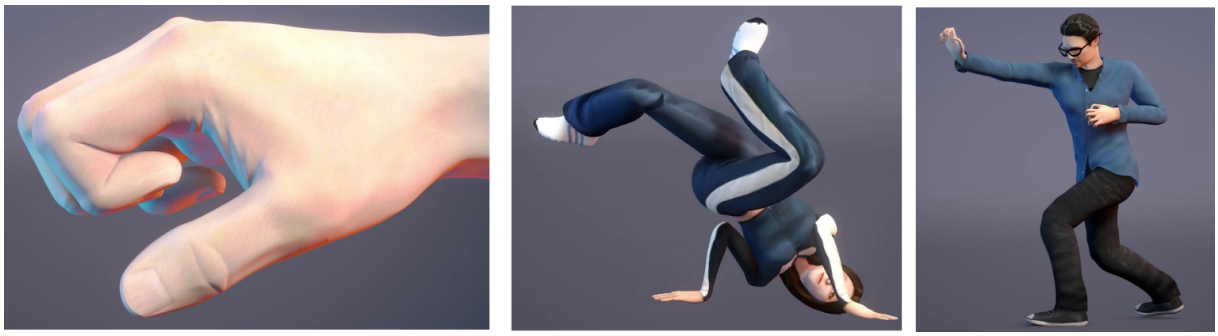


Figure 3.4: *Example of real case character animation exhibiting large angular deformation using implicit skinning.*

3.1.2 Discussion

We proposed a method coupling mesh surface representation and volumetric one. This hybrid representation mixing surface and volume has the advantage of improving the plausibility of the deformed object, notably in the organic type of articulation, as well as handling collisions, while keeping an efficient computational time. This hybrid model further allows to add an extra level of control through the choice of blending operators on top of the skinning deformation, varying from a sharp union, to smooth or bulging appearance. The approach may however requires some tuning in order to achieve the expected behavior, notably through the selection of adequate sample points for the implicit field, and is still more expensive in computation time and memory storage than raw skinning. This extra computation may be handled by modeling software, as proved by the transfer project integrating this method within the commercial software Modo, but may limit its applicability in video games.

The method has already been extended to handle skin elasticity in allowing the surface to slide along the tangential space [Vaillant *et al.*, 2014], thus increasing even more the plausibility of the result. Another possible way of improvement could be to tackle the control of the resulting shape, in allowing to interactively reshape the blending function of the spatial field, using for instance profile curve sketches, or proxy geometry.

Extending the implicit field embedding to other applications is also a possible avenue of research. We notably started to investigate through the beginning of the ANR FoldDyn in collaboration with the University of Toulouse, and Inria Bordeaux, the use of implicit formulation coupled with an underlying cloth simulation to handle multi-layer cloth animation, where the volume representation can help to handle collisions and wrinkles in an efficient and robust way.

3.2 Developable surfaces: Interactive virtual paper.

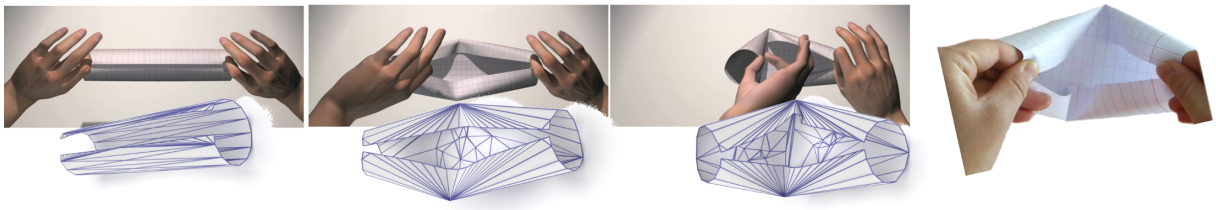


Figure 3.5: *Crumpling interactively a virtual sheet of paper using our approach. The right-most image shows a real sheet of paper.*

Paper is a common daily life material. Virtual animated paper is however, and surprisingly, hardly seen in 3D virtual environment. This can be explained by the difficulty to model efficiently animated crumpled paper. At a macroscopic level, excepted when torn, paper material may be modeled as a inextensible surface, and thus developable with respect to its pattern. This inextensibility property can be translated into non-linear constraints of length preservation which imply the use of costly optimization algorithms. Moreover, when crumpled, paper may exhibit sharp features that appear dynamically, typically, singular points called d-cones [Amar and Pomeau, 1997]. As a consequence, very dense meshes or adaptive remeshing are usually required, increasing even more the computational cost.

Our objective is to propose a new model of virtual paper that can be manipulated by the user, for instance, in applying interactively displacement to some specific position of the paper surface. The model of surface must react in bending, or crumpling if needed in a plausible way, why ensuring a fast interactive visual response.

Existing approaches may handle paper like surface deformation using thin shell physically based simulations [Grinspun *et al.*, 2003, Gingold *et al.*, 2004a, Burgoon *et al.*, 2006, Patkar *et al.*, 2015]. However, these simulations do not automatically handle crumpling that must be manually specified. To automatically handle crumpling, breakable springs were introduced in a coarse mass spring simulation [Kang *et al.*, 2009], but exhibits artifacts along the spring edges. Adaptive remeshing coupled to thin shell simulation, has recently been able to achieve high quality crumpled surfaces, but at the price of high computational costs ranging to minutes per simulated frame [Narain *et al.*, 2013].

Geometrical modeling specifically adapted to developable surface has also been studied. On top of works already mentioned in Chap. 2.1 for static shapes, interactive origami modeling [Solomon *et al.*, 2012, Bo *et al.*, 2016] was also studied, but requires the user to describe the sharp folds. Closer to our approach, Kergosien *et al.* [1994] propose model for an early model for bending and creasing of paper-like surfaces in some specific deformation scenario, and Hwang *et al.* [2015] propose an interactive manipulation of conical structure, although restricted to smooth developable surfaces.

In our case, we consider an initially flat surface which is interactively deformed via a set of handles. Each handle acts as a positional constraint a leads to a surface deformation which is computed using our surface model described in the following.

3.2.1 Hybrid crumpling deformable model based on generalized cone

***Note**: Further details on the following part can be found in [Schreck et al., 2015].*

We introduce a new hybrid model, interlacing physically based simulation with a new geometrical structure based on generalized cones as illustrated in Fig. 3.6.
As stated in Chap. 2.1, paper may be modeled by piecewise C^2-developable surfaces. Each piece being delimited by the sharp features, leading therefore to a surface globally C^0 only. The key idea of our approach consists in modeling each piece as either a plane, a generalized cylinder (rulings parallel to each other), or a generalized cone (rulings converging toward a single apex) as illustrated in Fig. 3.7. The separation between different generalized cylinder is modeled by a singular point, representing a d-cone structure.

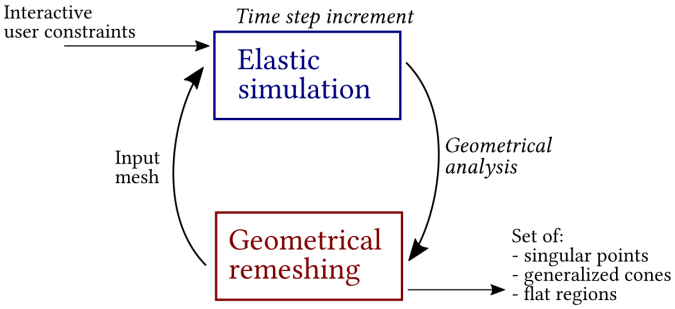


***Figure 3.6:** Overview of our hybrid model introducing a geometrical analysis and remeshing step interlaced with a standard elastic physically based simulation.*

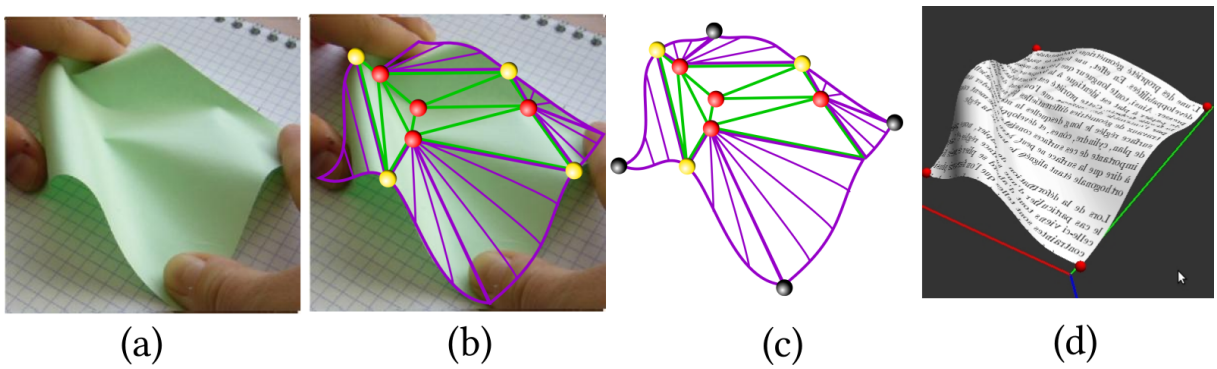


***Figure 3.7:** Our geometrical model of paper surface. (a): Real deformed paper. (b): Our model drawn over the real paper, and shown apart in (c). Generalized cylinders and cones are shown in purple, the red and yellow points indicate singularities, and the green regions are triangles defined by singularities. Black dots in (c) indicates user positional constraints. (d): A virtual model mimicking the real example on the left.*

This geometrical model is plugged as a standard coarse mesh into an elastic FEM physically based simulator [ARCSim] allowing to compute the main deformation applied to the surface with respect to the user constraints. Then, the geometrical model handles the developability constraint, and is used to update the connectivity of the underlying mesh.

This hybrid model has several advantages. Each geometrical primitive is naturally a developable surface with rulings defined explicitly in the case of generalized cylinders and cones. These rulings, when converging toward an apex defined within the domain of the surface, indicates the position of a singular point, corresponding therefore to a sharp feature. This singularity can thus be explicitly handled, and tracked through the deformation in order to model plastic behavior of the material, i.e. the persistence of the singularity once created. These rulings are further used to update the connectivity of the mesh used in the physically based simulator. More precisely, the edges of the mesh are set to be rules of the developable prim-

itives. Aligning the edges with respect to the rules has the advantage of adapting optimally the connectivity to the curvature of the surface, and then allowing to model accurately curved surfaces with very few polygons. Moreover, this naturally generates a more rigid behavior of the surface along the rulings when simulated using elastic elements, which is compatible with the behavior of real developable surface.

The technical challenge consists in computing the rulings of each piece of surface. We proposed an algorithm based on rule propagation working as follows and illustrated on a real case scenario in Fig. 3.8. First, the surface is initially deformed by an elastic deformation computed by the physically based simulator. This elastic behavior introduce small local compression on the surface. Let us call $\mathbf{d}$ the main direction of compression. Indeed, developable surface shouldn't undergo compression, but instead, should bump in this direction, and thus exhibits rulings aligned with $\mathbf{d}^{\perp}$.

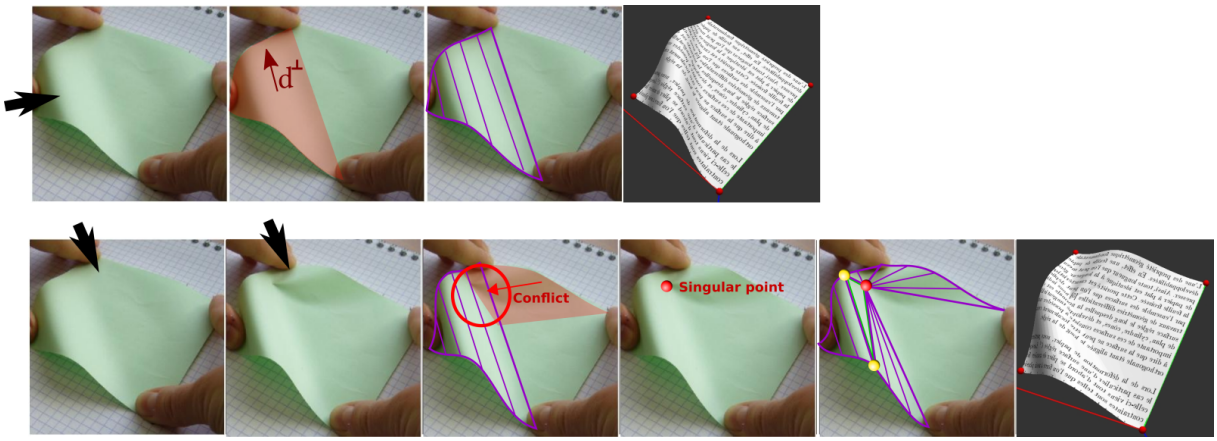


Figure 3.8: *Illustration of the generation of the rulings and singular points on a real case scenario of a piece of paper handled at its four extremities, and the black arrow indicates the displacement of a finger. The top row shows the rulings generation leading to a smooth surface bulging, while the second row illustrates the introduction of the singular point.*

Our algorithm consists in computing such rulings. Thanks to the developability of the surface, all computation can be made on the 2D pattern. We initiate and propagate parallel candidate ruling starting from the compressed area in the $\mathbf{d}^{\perp}$ direction until reaching the boundary of the surface, or other existing rulings. Then two different cases can be encountered. First, if the candidate rulings reach the boundary on their two extremities, meaning that the surface is deformed from locally planar to locally cylindrical, a smooth generalized cylindrical surface is instantiated (Fig. 3.8 first row). Second, if the candidate ruling meet another existing ruling, meaning that other parts of the surface is already bent in another direction, a singular point is introduced in the area where the rules are crossing (Fig. 3.8 second row). Note that the precise position of the singular point is selected using a random factor within the conflicting region. This singular point is set to be the apex of surrounding conical regions. Thus, another set of rulings converging toward this apex are generated in order to model them. The overview of the rulings propagation system is illustrated in Fig. 3.9.

Our paper model is able to reproduce some experimental behavior of real paper. We show in Fig. 3.10 a specific scenario where a user first bend a sheet of paper in approaching two opposite edges, and then pinch one of the edge in the orthogonal direction of the bending, thus creating a singular point. Note that our model allows to seamlessly adapts the rulings to the deformation of the paper, thus helping the underlying elastic simulation to reproduce

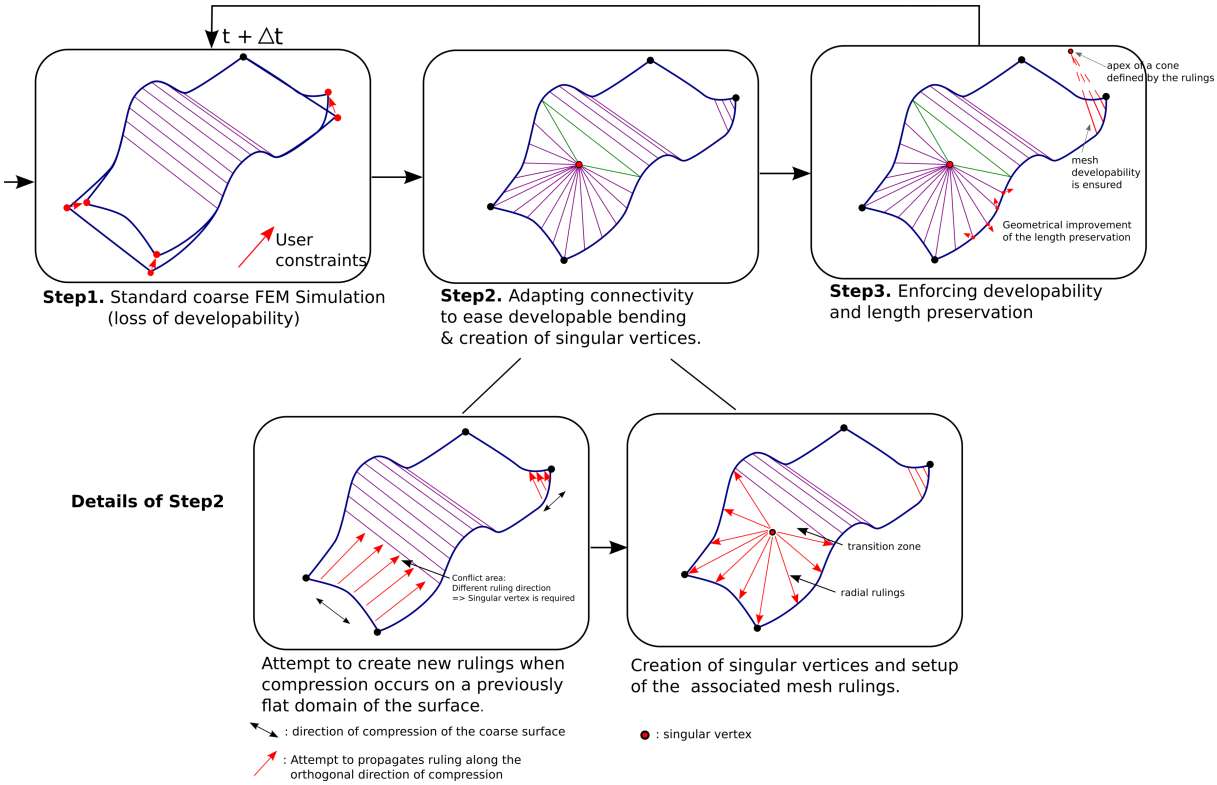


Figure 3.9: *Overview of our approach to generate singular points and rulings using a propagation algorithm.*

such deformation while keeping a very coarse triangulation.

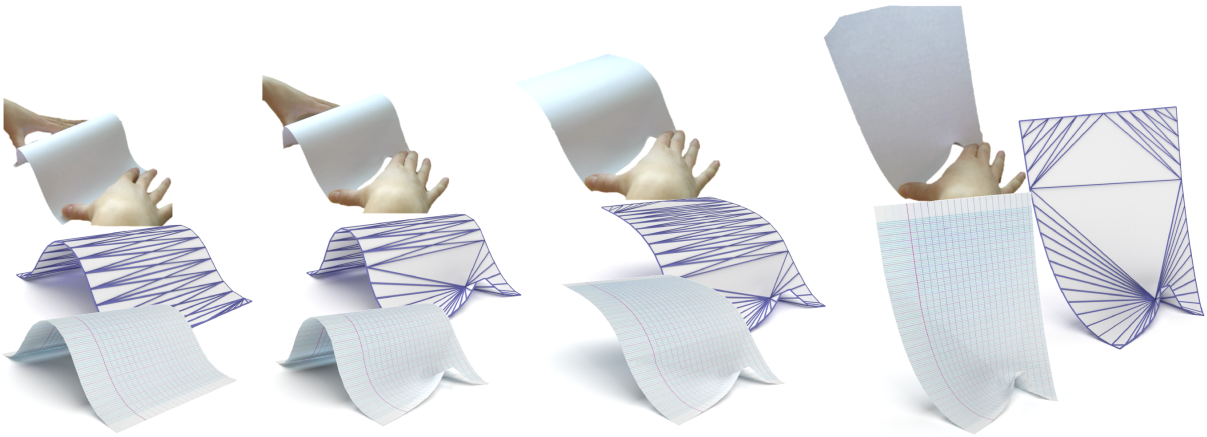


Figure 3.10: *Top: Real sheet of paper exhibiting a singular point when deformed. Bottom: Virtual model mimicking a similar deformation illustrating the underlying coarse triangulation of the surface, and a textured final result.*

We illustrate in Fig. 3.5 a more complex case where a sheet of paper is bent as a cylindrical shape, and then crumpled. In addition to being controlled by the elastic simulation, our model further depends on a threshold parameter limiting the maximal amount of admissible compression before starting the ruling propagation algorithm. Decreasing this parameter toward zero allow to model a more rigid behavior, where the sheet of paper crumples more than bend. We illustrate in Fig. 3.11 (left) a result obtained under the same input constraint, but with different threshold value mimicking various material rigidity.

Finally, we compared our results to standard finite element simulation, one using homogeneous triangulation, and a second using the state of the art remeshing approach from Narain et al. [2013]. Fig. 3.11 (right) shows that our approach and the simulation can exhibit similar behavior, but our approach is one to two orders of magnitude faster.

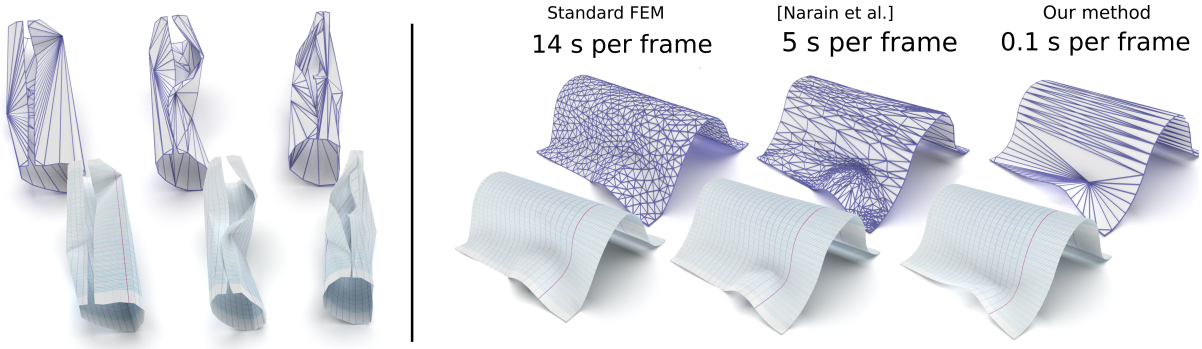


Figure 3.11: *Left: Modeling increasing rigidity behavior of paper from left to right. Right: Comparison between standard finite element simulation (left), Narain et al. approach (middle), and our result (right) for similar result quality.*

3.2.2 Interactive surface tearing

Note*: Further details on the following part can be found in [Schreck et al., 2017].*

We extended this model to handle interactive tearing in the case where the user is manipulating a sheet of paper with his hands corresponding to daily life scenarios. Existing tearing approaches usually focus on either fully user directly defines cuts such as for modeling surgical cuts [Sifakis *et al.*, 2007], or for sudden impact phenomenon such as fracture for ductile or brittle materials [O'Brien and Hodgins, 1999, Gingold *et al.*, 2004b]. Paper tearing is different as the tear is indirectly generated by the constraints that applies the user with his hands, potentially far from the tearing position. Moreover, the trajectory of the tear remains quite intuitively controllable as the tearing curve on the sheet can be seen as a continuous function of the user hand displacement.
Our method aims at computing efficiently two properties depending on the forces applied on the sheet. First, the localization of the creation, or extension, of a tear. Second, the orientation of the tear propagation. These two steps follow the remeshing described in the previous chapter as seen in Fig. 3.12.

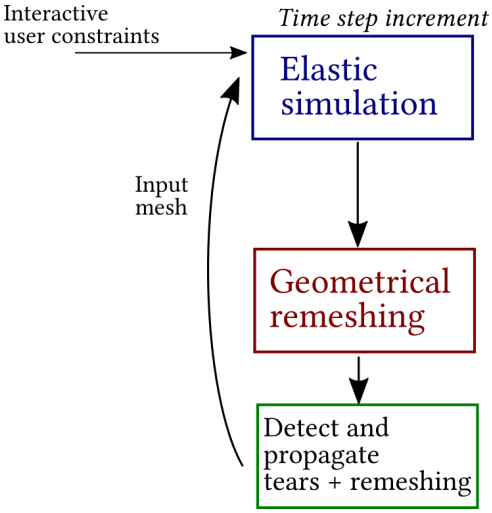


Figure 3.12: *The creation and propagation of the tearing is an extra step added after on top of our previous pipe-line. Note that the tearing introduce another step of remeshing around newly introduced tears.*

The localization of the creation of a tear, or the continuation of an existing one, depends on the so called *Griffith criterion*, stating that a tear propagates on a thin sheet of constant thickness

if the energy release rate E_r, expressing the amount of energy released by an infinitesimal fracture propagation is greater or equals to the fracture energy E_f. E_r can be expressed as a function of the forces acting on the extremity of the tear, while E_r is intrinsic to the material. While existing approaches require to compute these energies for all the vertices of the mesh [Pfaff *et al.*, 2014], we instead take advantage of our coarse geometrical model to predefine only a few potential points where the tear can start, namely, the singular points on the border of the surface which are the junction between two conical regions, the points exhibiting an inward angle on the boundary, and the finger positions, which are points where thin sheets stress can concentrate.

Our computation of the tear orientation is heavily relying on the following material science studies observation from [O'Keffe, 1994, Roman, 2013]. Let us consider a sheet of paper handled by two point A and B, where their respective pre-images on the pattern are $\overline{A}$ and $\overline{B}$. The sheet is pulled apart such that a tearing is initiated. We call C, the extremity of the tearing, respectively $\overline{C}$ its position on the pattern. Roman [2013] observes that the direction of propagation of the tear is the bisector of the lines $\overline{AC}$ and $\overline{BC}$ as illustrated in Fig 3.13 left. The key idea of our method is to apply this observation to a more general system where an existing tear is surrounded by arbitrary forces. Considering this general case where a tear extremity is surrounded by several forces, we cluster the forces into two equivalent and opposite total forces F_{left} and F_{right} as seen in Fig 3.13 right. These two equivalent forces can be seen as generated by two *virtual* points pulling the surface apart. Then, the tearing propagation is performed along the bisector direction of the 2D images of these forces $\overline{F}_{left}$ and $\overline{F}_{right}$.

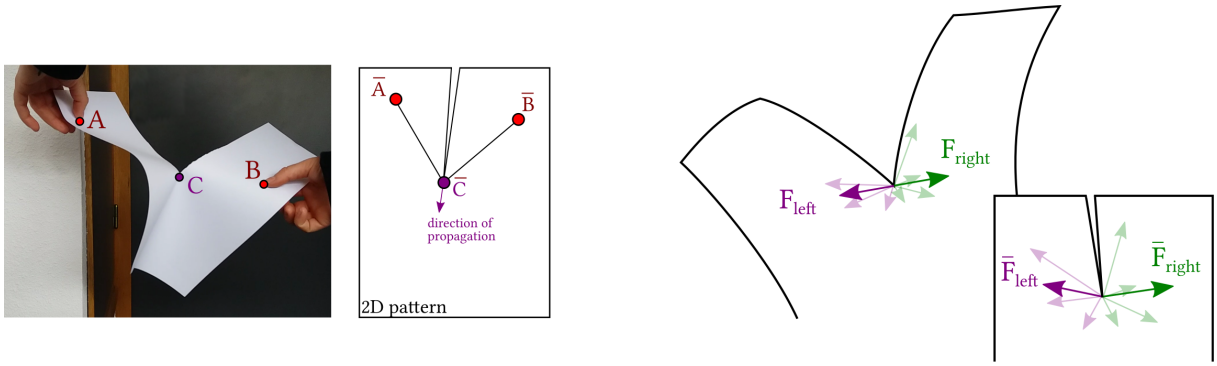


Figure 3.13: *Left: When tearing a paper in two points A and B, the direction of propagation of the tear at point C is given by the bisector of AC and BC in the 2D pattern space. Right: Our approach clustering an arbitrary set of forces acting on a tear extremity as two opposite forces F_{left} and F_{right}.*

Inspired from O'Keffe [1994], we proposed the following formulation from the energy E_r extending the formulation to equivalent forces F_{left} and F_{right}

$$E_r = 2F\cos(\theta/2)/e\ ,$$

where $F = (\|F_{left}\| + \|F_{right}\|)/2$, θ is the angle between $\overline{F}_{left}$ and $\overline{F}_{right}$, and e is the surface thickness. The fracture energy is defined as

$$E_f = K\,T_{paper}\ ,$$

where K is a constant value (in J/m^2) expressing the intrinsic resistance of the material toward tearing, and T_{paper} is a procedural noise texture with values $\in [0, 1]$ modeling the stochastic distribution of fibers within the paper surface, and thus its non homogeneous resistance to fracture.

Finally, tearing are propagated in the computed bisector direction along a small distance at each time step, and the surface is remeshed accordingly. To improve the visual appearance, we followed the approach from Lejemble *et al.* [2015] to model details along the tear extending it to texture based approach.

Figure 3.14 shows a sequence of deformations with two fixed constraints symmetrical in both sides of an edge point with inward angles, and leading to a straight tearing trajectory as expected by the theory [O'Keffe, 1994]. Fig. 3.15 illustrated two different scenarios where a tear is initiated at a junction point (point on the border separating two different generalized cone), and a tear propagating at the apex of a generalized cone. Fig. 3.16 left shows an example where the user actively control the trajectory of the tear in rotating iteratively his left, and then right hands, leading to this zigzag pattern. Finally, Fig. 3.16 right, illustrates a scenario where the fingers of the user change of position (with respect to the paper pattern) through the animation, leading to multiple tears.

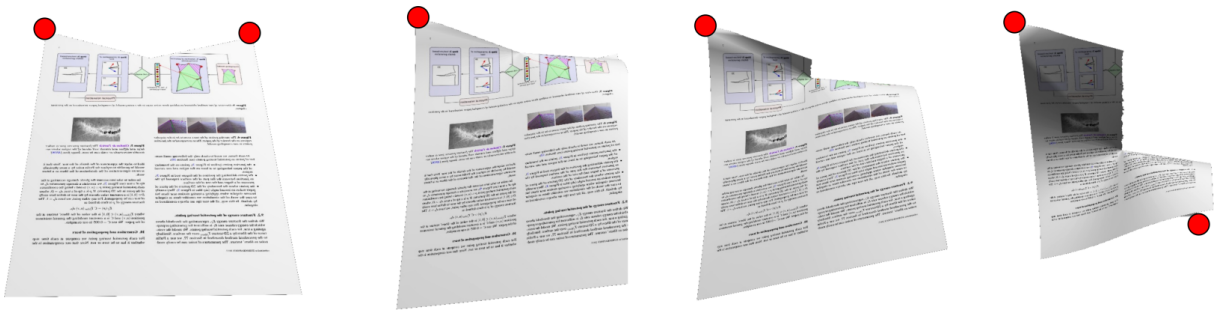


Figure 3.14: *Animated sequence of paper tearing starting at a junction point.*

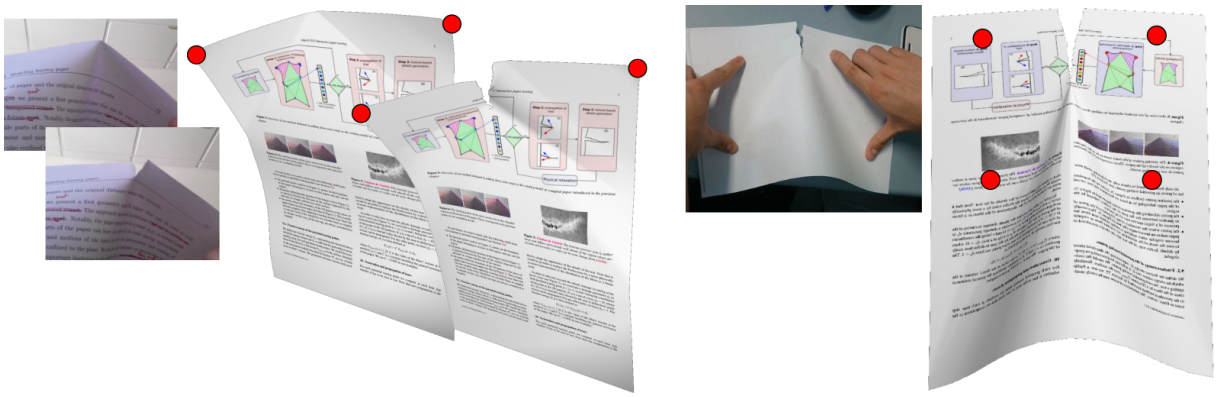


Figure 3.15: *Comparing real paper tearing with our mode in the case where the tear starts at a junction point (left), and when it is confounded with the apex of a generalized cone (right).*

3.2.3 Real time sound synthesis

Note*: Further details on the following part can be found in [Schreck et al., 2016].*

Finally, we further proposed a extension to real time sound synthesis method for paper material. While sound may not traditionally seen as part of 3D modeling and animation, it is still an important aspect of plausibility for the immersive aspect of the virtual world in general. We therefore believe that interactive and plausible sound is actually an indirect way to improve the plausibility of a 3D model.

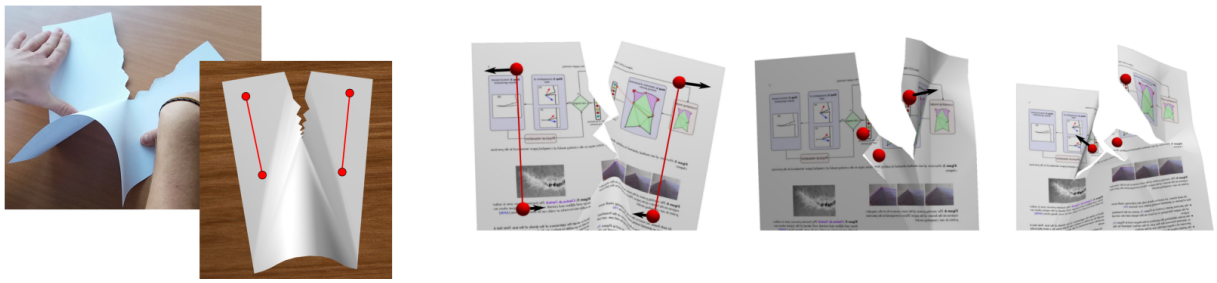


Figure 3.16: *Left: Generating a zigzag pattern in rotating alternatively the left and right hand. Right: Creating three tears in changing the position of the hands with respect to the pattern during the animation (the black arrows indicate the displacement of the fingers).*

Paper sound, although very familiar, is a complex combination of different sound styles. It is a mix between continuous noisy sound produced by frictional sliding, and discrete events produced by geometric bending and crumpling processes. One of the key specific characteristics of paper sound, at the opposite of cloth sound for instance [An *et al.*, 2012], is its strong dependence to its geometry. The challenge we tackled was the real time synthesis of this variety of sounds while taking into account their shape dependent nature.

Physically-based sound synthesis is able to simulate the vibration propagation at the origin of the sound, [O'Brien *et al.*, 2001] but requires very high computational cost which is not compatible with interactive applications. Modal analysis is an efficient approach to compute the vibrations mode and can drastically improve the efficiency of the computation for rigid objects [Adrien, 1991, van den Doel *et al.*, 2001]. However, modal analysis doesn't extend well to interactive deformable objects such as paper, as their vibration modes depend on their dynamic shape. In order to ensure efficient computation, we considered an example based approaches, called concatenative sound synthesis, where a bank of basic sounds are pre-recorded, and then adapted and mixed during run time to match the animation scenario. Our method inspires from the work of An *et al.* [2012] developed for garment sounds, but extends it to geometry dependent sound synthesis.

We identified that the variety of paper sound can be classified into two main types of sounds. First, the friction sounds, caused by regions of the surface sliding along either another object, or another part of the same surface, which can be modeled as a colored stochastic noise. Second, the sounds related to surface deformation, generated in regions where the bending direction changes suddenly, thus related to a change of curvature sign. Such transition between two equilibrium states produces a sudden burst of energy which is quickly dampened. This sound related to deformation can itself be split into two main subcategories as illustrated in Fig. 3.17, whether the deformation occurs on a smooth region, leading a long *flap* type of sound lasting a few hundred milliseconds, or to an already crumpled region leading to the sudden *crac* type sound lasting a few milliseconds. We also proposed a simplified model accounting for the relation between the shape of the surface and its sound. Based on different experiments, we found that the average length of the rulings l_r and the length of the free border l_c of the paper were parameters influencing the friction, and *flap* sound type.

Based on the previous remarks, we proposed a sound synthesis algorithm working as follows. In a precomputation step, a set of basic friction and curvature inversion sounds are recorded for various values of l_r, and l_c, and then stored in a data base. At run time, the geometrical surface is deformed using the method described in Chap. 3.2.1, and analyzed to catch sound producing events, namely collisions with the surface or another one, or a sudden change of

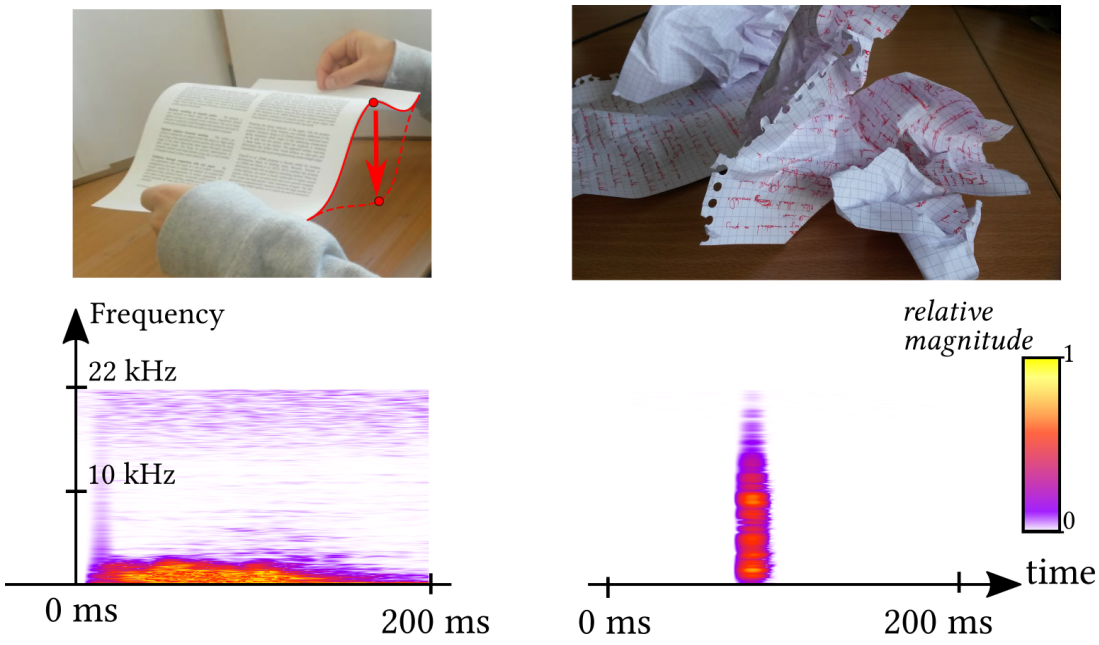


Figure 3.17: *Two types of crumpling sounds are considered and illustrated by their spectrograms on the bottom. Left: A change of curvature is applied to a smooth surface associated to a flap sound. Right: Typical short clac sound when manipulating a highly crumpled paper.*

sign of curvature. Theses events are associated to specific regions called *resonators*, related to regions which can freely oscillate to act as the source of the sound associated to the event. As illustrated in Fig. 3.18, the *resonators* are computed using a flood-fill algorithm starting for the event location, and propagated until reaching an edge with high curvature, indicating a stiff constraint on the surface, thus blocking the vibration propagation. Each resonator, corresponding to a part of a generalized cone, or constrained triangle, is then characterized with its respective l_r and l_c value depending on its geometry, which triggers the start of a sound exhibiting similar value from the data base.

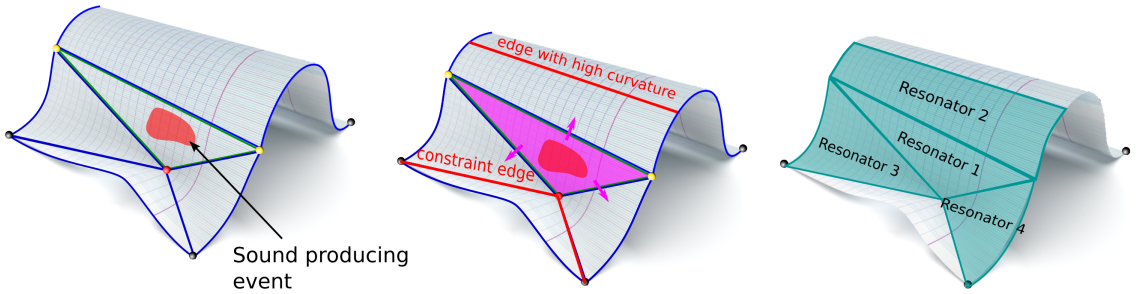


Figure 3.18: *A sudden change of geometry with inversion of curvature sign is an indicator that a sound must be generated. A flood fill algorithm computes the sound producing surface until reaching an edge with high curvature or a constraint edge linked to singularity. The resonators are then geometrically analyzed with respect to their values l_r and l_c corresponding to a pre-recoded sound to be played.*

At run time, several sounds can be mixed in parallel to model friction with bending and crumpling event, and can adapt to generic deformation scenario as illustrated in Fig. 3.19. Our data base of sounds contained 250 bending and crumpling sounds sorted by their respective l_r and l_c value, and 65 friction sounds between paper and a table in wood. The computation of an

event and the associated resonators ranged for about 1-50 ms, which allows interactive scenarios and stayed below the computational cost of the geometrical deformation of the paper surface.

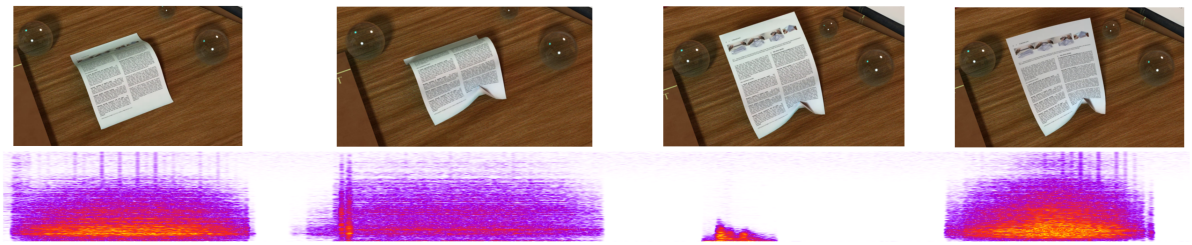


Figure 3.19: *Animation of deforming sheet of paper with associated sound illustrated by the spectrograms.*

3.2.4 Discussion

We presented a complete model for interactive virtual paper animation in coupling a developable geometrical model on top of an underlying elastic physical simulation. Our model is able to handle non smooth surface crumpling, and tearing, while keeping a very coarse dynamically adaptable triangulation, as well as synthesizing the sound of the deformed surface.

Our model has been compared to real case scenarios and show plausible visual appearance for simple deformation scenarios, although not reaching predictive simulation behavior due to a rough underlying elastic simulation model. The amount of crumpling applied to the surface is also limited, notably, due to the lack of collision handling which would be necessary to model highly crumpled paper behavior. Therefore, completely crumpling a sheet of paper as a ball like structure cannot be modeled at this stage.

Several avenues of continuation and improvement of this project could be developed in addition of collision handling. Real paper exhibits anisotropic behavior due to preferential fibers orientation set during its fabrication. This anisotropy could be taken into account for the bending model as well as within the tearing propagation algorithm. Our model also assumes d-cone to be represented by a single point, while its local geometry may be represented more accurately with a semi-circular shape. Handling sharp folds could also fill the gap between our crumpling model and origami modeling.

It is worth noticing that one of the main applications of our model can be virtual world, where the user could, for instance, in an immersive environment, interactively manipulate sheets of paper. Although our model allows interactive feedback, it is not yet reaching real time deformations. The bottleneck of our model remains the underlying elastic simulation that we used as a black box, despite the few number of triangles handled. A more adapted simulation model, able to handle efficiently few elongated triangles, may improve drastically the computational efficiency. Another interesting side we didn't tackle in this project concerns the user interface system. We either deformed our model using single handled controlled by the mouse, or, in scripting trajectory when several handles must be displaced at the same time. Proposing a system able to interact more directly with hands could improve the immersive feeling of the system. Interfacing our deformation into a multi-touch screen could be a first way of interacting in a more direct manner. Tracking the hands trajectory, while using a real sheet of paper as tangible support could be another way to explore.

3.3 Fluids: Spatio-temporal sculpting of fluid animation.

***Note**: Further details on the following part can be found in [Manteaux et al., 2016].*

Fluids, and specifically liquids, are commonly encounters in CG production [Opstal *et al.*, 2014, Reish *et al.*, 2016]. Fluid surface animation may exhibit very intricate dynamic shape patterns and drastic topological changes such as in the case of droplets splashing in and out of water. These characteristics make controllable animation of fluid very challenging.

Fluids are usually computed using two main approaches. The first one is physically based simulation, either based on Eulerian model [Stam, 1999] usually derived from the Navier-Stokes equation, either on Lagragian particule based approach [Müller *et al.*, 2003] with, for instance, the popular Smooth Particle Hydrodynamics (SPH) model [Desbrun and Cani, 1996]. Fluid simulation is of main interest when modeling a fluid interacting with external solid object such as rocks, or moving boats which can be seen as boundary conditions. Fluid simulations are however particularly expensive in terms of computation time and memory cost, and their parameters such as fluid viscosity are known to be hard to tune in order to achieve a desired effect. The second popular approach is procedural animation, which is able to model at interactive rate a surface animation. However, procedural approach is usually restricted to very specific scenarios such as open ocean waves [Hinsinger *et al.*, 2002, Tessendorf, 2004, Horvath, 2015], and cannot easily adapt to interact with other objects.

Generating a high resolution simulation on top of an existing low resolution one has been studied for smoke [Nielsen *et al.*, 2009, Yuan *et al.*, 2011] and liquid [Nielsen and Bridson, 2011], thus helping to tune simulations, but doesn't allow direct user control. More users guided approaches include the definition of a target shape which attracts the fluid toward it during the simulation [Treuille *et al.*, 2003, Shi and Yu, 2005, Raveendran *et al.*, 2012], or the user depiction of velocity field [Kim *et al.*, 2006] and control particles [Madill and Mould, 2013]. In the spirit of user direct control, Schpok *et al.* [2005] propose a method where the user can place and parameterize animated smoke primitives in a scene, and Pan *et al.* [2013] propose a sketch based approach to edit waves profiles.

Our insight was to allow direct fluid animation sculpting. More specifically, to allow the user to select animated elements from an existing fluid animation such as splashes, or waves, potentially modify them in space and time, and finally merge them within the same or another fluid animation. This approach generalizes the well-known *copy and past* paradigm to animated elements, called *space time features*. We specifically design our approach to take raw animated mesh sequences as inputs. Such mesh sequences may be generated either by volume or particle simulation, or either by procedural generation, and can be mixed using our approach, thus allowing different types of inputs to be assembled in a single, feature rich, resulting animation.

3.3.1 Space time features model

Let us describe our notion of space time feature. Fluid animations may contain specific visual events occurring in a limited spatial support and time sequence. We call a visual event, a geometrical region of the fluid surface with high curvature value, at the opposite of a rest bed of water associated to a flat surface. This event has a spatial support, described as the contiguous region exhibiting high curvature, and a limited life time. Both support and geometry of the

visual event may vary at each time frame. The set of all spatial support associated to the respective time frames is called a *space-time feature*.

We classify the space-time feature between two categories with respect to the data-structure used to represent them as illustrated in Fig. 3.20. First, the isolated features, which are typically individual droplets falling under gravity until reaching a bed of fluid. Each droplet can be represented by a boundary free mesh at a given time step. Second, the differential features, which are local details on top of a bed of water, and typically represent waves. Each of these features can be modeled, at a given time step, by a 2D map as seen in Fig. 3.21 representing a deformation to be applied on top of a smooth underlying surface.

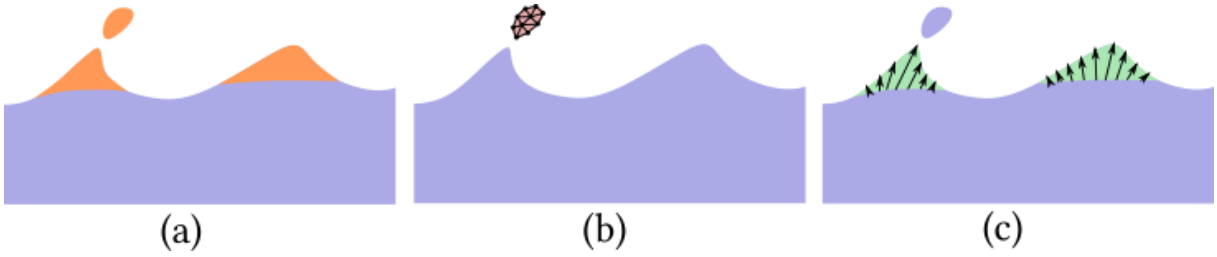


Figure 3.20: *Space-time features (a) are visually depicted by high curvature regions and can be classified as isolated features for droplets (b), or differential features for waves (c).*

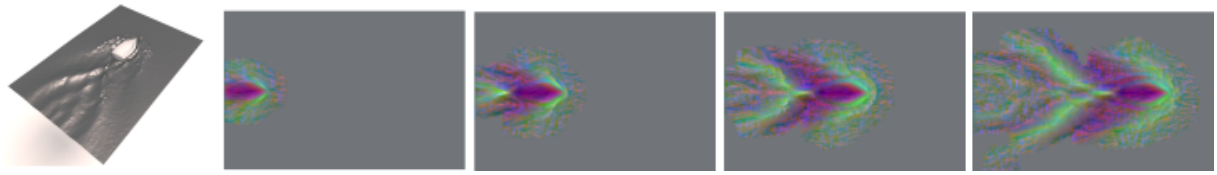


Figure 3.21: *Differential map encoding the wave associated to the displacement of a boat (left-most image) over a bed of water at different time step (right).*

The computation of the space-time feature is performed interactively in four steps. First, the absolute value of the mean curvature is computed at each vertex at every frame, and the resulting curvature map is segmented. Second, the user may adapt the level of detail of the segmented region in performing morphological erosion and dilatation. Third, the segmented regions, which are computed independently at each frame, are then aggregated in the time domain. We considered a similarity metric between two regions at consecutive time frame taking into account the proximity of the center of mass, surface, and possibly volume for isolated features. All regions are viewed as node of a graph, and the aggregation is then computed as the minimal cost of a vertex-disjoint path covering algorithm. Fourth, each region which is segmented in space and time represents a space-time feature, and each of them are stored either in their mesh, or, differential form.

3.3.2 Space-time feature sculpting

Once computed, the space-time features can be used as sculpting material for fluid animation following an select-edit-paste pipe-line. In our system (see interface in Fig. 3.22), the user can visually select one, or more, space-time feature. Once selected, features are then cached and ready to be locally modified independently of the rest of the animation. We propose the following action that can be applied to a feature.

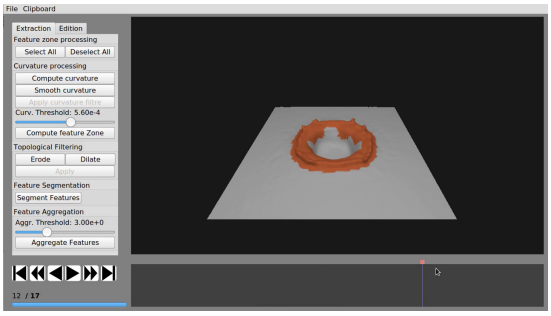


Figure 3.22: *Interface of our system to select-edit-past fluid animation. Here a splashing wave has been segmented.*

- *Space-time deformation*: the user can change the spatial position, orientation and scale of the feature, which will act at all time frames of the feature life time. Time deformation includes the change of the initial frame associated to the feature, as well as increasing or slowing down its speed.
- *Trajectory editing*: the user can indicate specific key positions at different time frame, allowing to edit the trajectory of the feature.
- *Fade in and out*: ensure that a differential feature smoothly appears, or disappears, which allow to smoothly past them in existing animation.
- *Export and import*: allows to save, or reused, stored data-base of existing space-time features.
- *Paste*: the final action of pasting a cached space-time feature onto the same, or another, fluid animation. The paste action is trivially implemented for isolated feature in inserting the mesh structure within the animation, and requires to apply the deformation map in the case of a differential feature.

Our system can typically be used to take as inputs several effects, possibly computed from different methods, and merge them into a final enriched animation. We model this way a set of boats traversing an ocean type of fluid as seen in Fig. 3.23. This animation was built using two different inputs. First, the waves associated to the traversing boat were computed using a FLIP simulation [Zhu and Bridson, 2005] on top of a rest bed of water. The 2D differential maps representing the wave of the boat are shown in Fig. 3.21. Second, the ocean waves were generated procedurally [Tessendorf, 2004] and exhibits numerous small scale details. The boat wake was extracted and pasted into the ocean animation at three different positions with different scales and orientations.

A second application consists in creating an artificial, but plausible scene. In Fig. 3.24, a liquid drop is falling on a bunny which reacts as a fluid. In this case, a droplet and its splashing waves were extracted from a fluid simulation, and then pasted into a static model of bunny interpreted as an underlying bed of water. Note that such scenario doesn't fit to the physically based pipe-line of standard fluid simulation tools, but can easily be modeled with our approach.

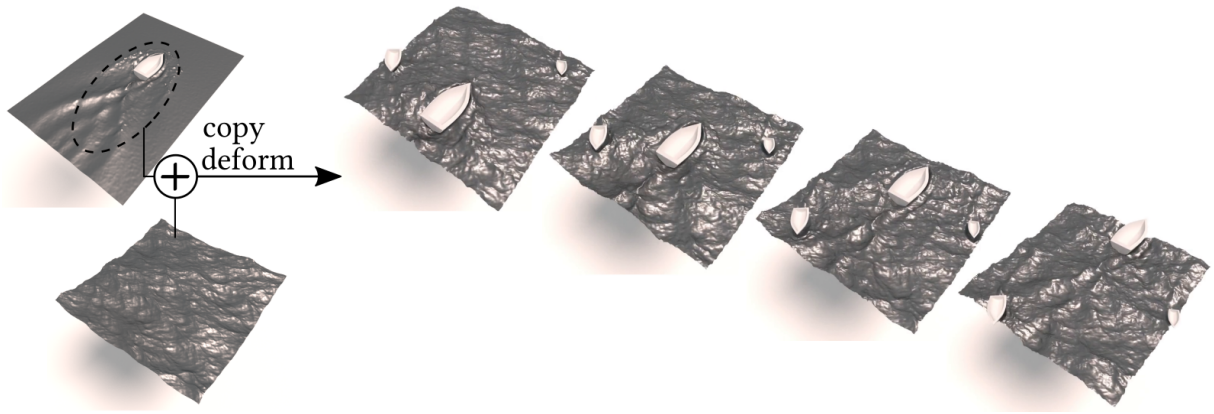


Figure 3.23: *An input FLIP simulation (top left) and procedural animation (bottom left) are used to animate several boats traversing and interacting with an ocean like animation (right).*

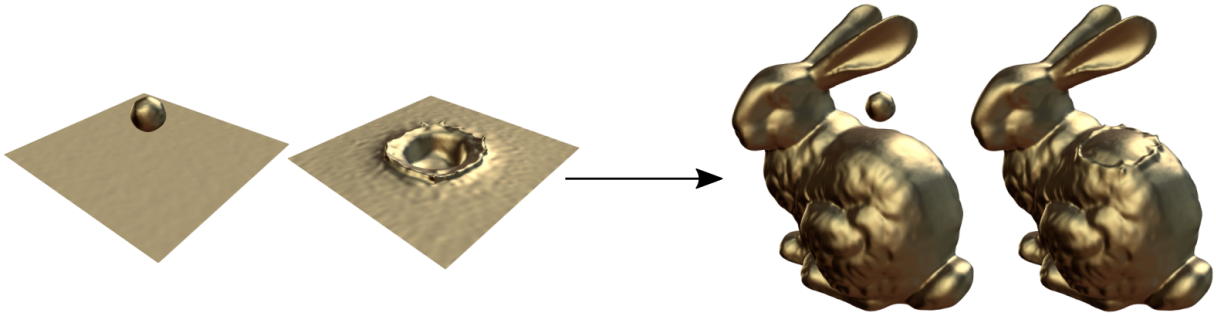


Figure 3.24: *An input animation (left) modeling a falling droplet is used to model an artificial scene of a droplet falling onto a static bunny and reacting as an underlying fluid (right).*

3.3.3 Discussion

We proposed a simple, yet efficient, system for direct interaction with fluid animation following a select-edit-paste approach. This approach is compatible with the view of a sculptor using additive material that can model, deform, and merge pieces to create a rich result, while applied to space-time material. As animations are manipulated at the mesh level surface, instead of the volume or simulation level, the user can blend various effects computed using different algorithms. At this state, our application only offers basic, but generic, editing tools. In this application the user is also fully responsible for the visual plausibility of the result as the system provides him a direct control which may allow arbitrary scenario, but doesn't ensure physical plausibility. Extension to specialize this system to plausible fluid animation could be done in integrating specific constraints to be satisfied when pasting a space-time-feature. For instance, constant volume could be locally enforced around the region where a feature is pasted. Storing, in some case, fluid velocity when the input is computed using simulation, may also be used to ensure conservation of momentum.

Although we illustrated our method with applications to liquid animations, we believe the generic approach of select-edit-paste of space-time content is a powerful approach that could be applied to several other applications such as smoke animation, explosions effects, but also deformable models such as cloth animation in adding animated details, or more generally animated scenery that would benefit from merging procedural effects with simulations such as natural open-air scenes. Interestingly, crowd animation is another example where sculpting approach has been studied [Jordao *et al.*, 2014]. In all these scenarios, specialized space-time

deformers may be developed in order to ensure that the resulting animation remains plausible, and leave several avenues of future works. Another, more generic, extension would be to propose an automatic topological relation extraction. At the current state, visual events are segmented and aggregated through time without taking into account their causality. However, different events may be related to each other as action and reaction response, as for instance, a falling droplet is at the origin of a resulting wave. Automatically detecting, and linking such causality could be a way of describing the topology of an animated scene. This causality could be stored as a space-time graph, and could have several prospective usages. For instance, allowing to select and interact with the event but also with its cause and result in order to automatically ensure consistency of the animation. Or, in a more prospective way, describing a coarse *fluid story* in filtering for instance the most significant events, and storing, compressing, or comparing animation based on their graph of events.

CHAPTER 4

Conclusion and perspectives

We presented several computationally efficient approaches allowing to automatize shape and animation creation in integrating high level knowledge to existing models in various applications domains. We showed that such integration can be used to automatically synthesize a visually plausible model, or preserve such plausibility through the deformation. This high level knowledge can also be used to improve the ways to control the 3D model, thus improving its interactivity under the gesture of the user. We summarize in table 4.1 the a priori, or plausibility criteria selected in the presented methods. We also assess the general computational cost and level of user control on associated the method.

Chap.	**A priori/Plausibility**	**Computational cost**	**High level control**
2.1	Developable surface	+ (1-5 s)	+ (Sketch based)
2.2.2	Constant spatial distribution	++ (real time)	+ (Spatial free form deformation)
2.2.3	Preservation of relation b/w sub-elements	- (high)	+/- (Automatic synthesis)
2.2.4	Preservation of hierarchical constraints	n/a	++ (Spatial free form deformation)
2.3	Spatio-temporal relations b/w organs	+ (interactive)	+/- (ontology query)
3.1	Articulation b/w rigid elements	+ (real time)	+/- (Blending operator)
3.2.1	Developable conical regions	+ (0.1s)	+/- (Direct handle control)
3.2.2	Tearing geometrically guided by hand positions	++ (real time)	+/- (Direct handle control)
3.2.3	Basic sounds types	++ (real time)	+/- (Sound data base)
3.3	Space-time features exhibits consistent high curvature	+ (interactive)	++ (Direct user selection and operations)

Table 4.1: *Summary of the characteristics of the presented models.*

In addition to developing and extending other types of knowledge for CG applications, I would like, as a longer perspective vision, allow to deepen the use, integration, and development of 3D model as a central part of other scientific disciplines such as medical applications, biology, mathematics, or more generally, physics sciences. Although 3D models are increasingly used within fundamental sciences, they are often used as external black box tools for dissemination visualization and final rendering. Indeed, the complexity of these 3D modeling tools don't allow them to take part as a whole in the core research process of these domains. Still, adapted 3D models can help to the comprehension of the studied phenomenon, and thus, be an inspiring source for the researchers themselves. In being visualized, but also, deformed and animated in an interactive way within this context through the use of adapted interface with respect to the domain, 3D models can help to improve our comprehensive vision. Reciprocally, this higher level of comprehension, will lead to evolution and refinement of the 3D model itself in adapting the adequate level of freedom and controls, thus bringing contributions to the computer graphics field as well.
To this end, I have already started multidisciplinary projects. In the medical domain, in addition to the project related to embryology modeling presented in this document, I also collaborated with Grant Gullberg from the Lawrence Berkeley National Laboratory on the modeling and visualization of fibrous and sheet structure of the heart in normal and pathological cases. And we plan, with another team, to collaborate in related projects for the brain imaging using a multi-resolution modeling approach. I am also part on another multidisciplinary project relating fundamental mathematics and computer graphics, with the Hevea project aiming at developing tools to model and visualize new paradoxical surfaces. The first result of the project concerned the first visual representation of a *smooth flat torus*, a C^1 surface isometric to the 2D unit square exhibiting a torical topology. Its visual representation, exhibiting both smoothness and fractal structure, as been largely reused in general public diffusion magazines such as Pour la Science[1], La Recherche[2], Science et Vie[3], showing the general interest for such applications.

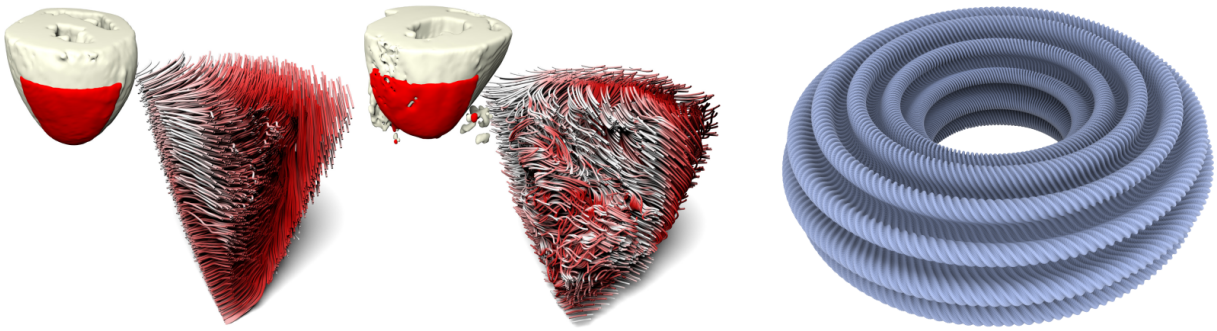


Figure 4.1: *Left: Comparing the fiber organisation on respectively, healthy, and pathological heart. Right: Rending of the flat torus object.*

In order to develop high level 3D models even further, I propose to develop the following two axes.
First, to develop new strategies to handle **large and complex level of details** that are necessary to synthesize and interact with more detailed shapes and animation. To this end, hybrid models are of main interest, in mixing procedural, simulation, and example based approaches.

[1]Les fractales lisses, un nouvel objet mathématique, n.245, Mars 2013.
[2]Le tore plat carré visualisé grâce à l'informatique, n.471, Janv. 2013; n. 467, Janv. 2012.
[3]Le tore plat n'a plus de secret, n.1138, Jul. 2012

These models will need to be enriched in relevant and intuitive way using for instance machine learning on distributions, advanced parameterization, or the use of sketches providing a large freedom to the user. Dedicated constraints related to each element and their interactions will need to be taken into account, and at the opposite of traditional simulation approaches, these models will need to be described at high levels, while being able to infer finer details in a consistent way. An interesting, and challenging area of research is the control of detailed animation. Sketching, copy and past, or sculpting approaches, have been specifically successful to model 3D shape, and extending these to animation may be very promising. Transfer methodology may also be of main interest to design animation. For instance, artists commonly sketch, or even mimic motion or specific rhythm they want to integrate within a virtual model. Controlling the animation would not directly be handled on the model, but could be defined through an intermediate prototype, either physical or virtual, or even sound based, from which we can extract, analyze, and transfer motion parameters that will need to be adapted to the targeted object.

Second, I propose to further integrate **perceptual notions within the interaction model**. Physically based models corresponds to local description which may be very accurate on the numerical side, but may not necessarily highlight high level comprehension of the associated phenomenon, nor allow their direct control. Moreover, some terms in the equation of the physical model may be particularly hard to solve, while not necessarily needed to the visual plausibility of the result. An example of this remark is the Navier-Stokes equation, guiding the motion of fluids, where the link between each term of the equation and phenomenon such as the size of waves, or vortices, is not clearly apparent, although mainly impacting the visual result. The objective is then to propose an alternative model, allowing to highlight relevant terms on the visual aspect, and, conversely, allowing to relax constraints with low visual impact. These alternative models should both integrate the geometry of the object, but also the expression of its associated motion in order to be able to control it. The resulting simulation will therefore not be necessarily physically accurate, but rather perceptually valid, and may be more quantitatively evaluated through perceptual studies. Note that one of the main constraints especially relevant for the visual plausibility is collision and self-collision handling. Developing models integrating efficiently and intrinsically collision avoidance, through for instance, the use of implicit representation, remains a way to pursue in order to apply deformation on complex models. Finally, interacting with complex phenomena will require the development of methods able to visualize and interact with abstractions. For instance, objects such as paper surfaces or plastics are linked to length or metric constraints, mechanical objects are depending on tensorial constraints, and more generally, shape and motions are expressed by style and rhythms. In order to preserve, or even interact, with such abstract constraints, dedicated and efficient visualization approaches, analysis on existing or acquired objects, as well as new interaction methods, will need to be developed.

Bibliography

[Adrien, 1991] J.-M. Adrien. The missing link: Modal synthesis. *Representation of Musical Signals*, pages 269–298, 1991.

[Alexa, 2002] Marc Alexa. Linear combination of transformations. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 21(3), 2002.

[Alhashim *et al.*, 2012] Ibraheem Alhashim, Hao Zhang, and Ligang Liu. Detail-Replicating Shape Stretching. *The Visual Computer*, 28(12), 2012.

[Aliaga *et al.*, 2016] Daniel Aliaga, Ike Demir, Bedrich Benes, and Michael Wand. Inverse procedural modeling of 3D models for virtual worlds. *ACM SIGGRAPH Course Notes*, 2016.

[Amar and Pomeau, 1997] M. Ben Amar and Yves Pomeau. Crumpled paper. *Proceedings of the Royal Society. Mathematical, Physical and Engineering Sciences.*, 1997.

[An *et al.*, 2012] Steven An, Doug James, and Steve Marschner. Motion-driven Concatenative Synthesis of Cloth Sounds. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 31(4), 2012.

[Angelidis and Singh, 2007] Alexis Angelidis and Karan Singh. Skinodynamic skinning using volume-preserving deformations. *Symposium on Computer Animation*, 2007.

[Bae *et al.*, 2008] Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. ILoveSketch: As-natural-as-possible sketching system for creating 3D curve models. *ACM Symposium on User Interface and Technology*, 2008.

[Barroso *et al.*, 2013] Santiago Barroso, Gonzalo Besuievsky, and Gustavo Patow. Visual Copy & Paste for Procedurally Modeled Buildings by Ruleset Rewriting. *Computers and Graphics*, 37(4), 2013.

[Barrow and Tenenbaum, 1981] H. Barrow and J. Tenenbaum. Interpreting line drawings as three-dimensional surfaces. *Artificial Intelligence*, 17, 1981.

[Bergou *et al.*, 2007] Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. Tracks: Toward directable thin shells. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 26(3), 2007.

[Berner *et al.*, 2011] Alexander Berner, Michael Wand, Niloy Mitra, Daniel Mewes, and Hans-Peter Seidel. Shape Analysis with Subspace Symmetries. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 2011.

[Bo and Wang, 2007] Pengbo Bo and Wenping Wang. Geodesic-controlled developable surfaces for modeling paper bending. *Computer Graphics Forum*, 26(3), 2007.

[Bo *et al.*, 2016] Pengbo Bo, Johannes Wallner, and Helmut Pottmann. Interactive Design of Developable Surfaces. *ACM Transactions on Graphics*, 35(2), 2016.

[Bokeloh *et al.*, 2011] Martin Bokeloh, Michael Wand, Vladlen Koltrun, and Hans-Peter Seidel. Pattern-aware shape deformation using sliding dockers. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 30(6), 2011.

[Botsch and Kobbelt, 2004] Mario Botsch and Leif Kobbelt. An Intuitive Framework for Real-Time Freeform Modeling. *ACM Transations on Graphics, Proc. ACM SIGGRAPH*, 23(3), 2004.

[Brune *et al.*, 1999] R. Brune, J. Bard, C. Dubreuil, E. Guest, W. Hill, M. Kaufman, M. Stark, D. Davidson, and R. Baldock. A Three-Dimensional Model of the Mouse at Embryonic Day 9. *Developmental Biology*, 216, 1999.

[Burgoon *et al.*, 2006] Rob Burgoon, Zoë J. Wood, and Eitan Grinspun. Discrete Shells Origami. *Proceedings of Computers and their Applications*, 2006.

[Cani, 1993] Marie-Paule Cani. An implicit formulation for precise contact modeling between flexible solids. *ACM SIGGRAPH*, 1993.

[Carmo, 1976] Manfredo P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976.

[Cordier *et al.*, 2013] Frederic Cordier, Hyewon Seo, Mahmoud Melkemi, and Nickolas Sapidis. Inferring mirror symmetric 3D shapes from sketches. *Computer-Aided Design*, 45, 2013.

[Decaudin *et al.*, 2006] P. Decaudin, D. Julius, J. Wither, L. Boissieux, A. Sheffer, and M.-P. Cani. Virtual Garments: A Fully Geometric Approach for Clothing Design. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 25(3), 2006.

[Dekkers and Kobbelt, 2014] Ellen Dekkers and Leif Kobbelt. Geometry Seam Carving. *Computer-Aided Design*, 46, 2014.

[Demaine and Demaine, 2007] Erik Demaine and Martin Demaine. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graph and Combinatorics*, 2007.

[Desbrun and Cani, 1996] Mathieu Desbrun and Marie-Paule Cani. Smoothed Particles: A new paradigm for animating highly deformable bodies. *Workshop on Computer animation and simulation*, 1996.

[Fondevilla *et al.*, 2017] Amélie Fondevilla, Adrien Bousseau, Damien Rohmer, Stefanie Hahmann, and Marie-Paule Cani. Patterns from Photograph: Reverse-Engineering Developable Products. *Accepted to Computer & Graphics, Proc. Shape Modeling International*, 2017.

[Frey, 2004] William H. Frey. Modeling buckled developable surfaces by triangulation. *Computer Aided Design*, 36(4), 2004.

[Funkhouser *et al.*, 2004] Thomas Funkhouser, Michael Kazhdam, Philip Shilane, Patrick Min, William Kiefer, Ayellet Tal, Szymon Rusinkiewicz, and David Dobkin. Modeling by Example. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 23(3), 2004.

[Gal *et al.*, 2009] Ran Gal, Olga Sorkine, Niloy Mitra, and Daniel Cohen-Or. iWIRES: An Analyze-and-Edit Approach to Shape Manipulation. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 28(3), 2009.

[Gao *et al.*, 2014] Ming Gao, Nathan Mitchell, and Eftychios Sifakis. Steklov-Poincaré Skinning. *Symposium on Computer Animation*, 2014.

[Garcia *et al.*, 2013] Francisco González Garcia, Teresa Paradinas, Narcis Coll, and Gustavo Patow. *Cages: A multilevel, multi-cage-based system for mesh deformation. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 32(3), 2013.

[Gingold *et al.*, 2004a] Yotam Gingold, Adrian Second, Jefferson Y. Han, Eitan Grinspun, and Denis Zorin. A Discrete Model for Inelastic Deformation of Thin Shells. Technical report, Courant Institute of Mathematical Sciences, New York University, 2004.

[Gingold *et al.*, 2004b] Yotam Gingold, Adrian Secord, Jefferson Han, Eitan Grinspun, and Denis Zorin. A Discrete Model for Inelastic Deformation of Thin Shells. *Technical Report*, 2004.

[Gourmel *et al.*, 2013] Olivier Gourmel, Loic Barthe, Marie-Paule Cani, Brian Wyvill, Adrien Bernhardt, Mathias Paulin, and Herbert Grasberger. A Gradient-Based Implicit Blend. *ACM Transactions on Graphics*, 32(2), 2013.

[Gray *et al.*, 2006] Alfred Gray, Elsa Abbena, and Simon Salamon. *Modern Differential Geometry of Curves and Surfaces with mathematica.* Chapman and Hall/CRC, 2006.

[Grinspun *et al.*, 2003] Eitan Grinspun, Anil Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. *Symposium on Computer Animation*, 2003.

[Gruber, 1993] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2), 1993.

[Hinsinger *et al.*, 2002] Damien Hinsinger, Fabrice Neyret, and Marie-Paule Cani. Interactive Animation of Ocean Waves. *Symposium on Computer Animation*, 2002.

[Horvath, 2015] Christopher Horvath. Empirical directional wave spectra for computer graphics. *Symposium on Digital Production*, 2015.

[Hwang and Yoon, 2015] Hae-Do Hwang and Seung-Hyun Yoon. Constructing developable surfaces by wrapping cones and cylinders. *Computer-Aided Design*, 58, 2015.

[Jordao *et al.*, 2014] Kevin Jordao, Julien Pettré, Marc Christie, and Marie-Paule Cani. Crowd Sculpting: A space-time sculpting method for populating virtual environments. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 33(2), 2014.

[Jung *et al.*, 2015] Amaury Jung, Stefanie Hahmann, Damien Rohmer, Antoine Begault, Laurence Boissieux, and Marie-Paule Cani. Sketching Folds: Developable Surfaces from Non-Planar Silhouettes. *ACM Transactions on Graphics*, 34(5), 2015.

[Kang *et al.*, 2009] Young-Min Kang, Heng-Guang Zhang, and Hwan-Gue Cho. Plausible Virtual paper for Real-time Applications. *CASA Short Paper*, 2009.

[Kavan and Sorkine, 2012] Ladislav Kavan and Olga Sorkine. Elasticity-inspired deformers for character articulation. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH Asia*, 31(6), 2012.

[Kavan and Zára, 2005] Ladislav Kavan and Jirí Zára. Spherical Blend Skinning. *Symposium on Interactive 3D Graphics and Games*, 2005.

[Kavan *et al.*, 2007] Ladislav Kavan, Steven Collins, Jirí Zára, and Carol O'Sullivan. Skinning with dual quaternions. *Symposium on Interactive 3D Graphics and Games*, 2007.

[Keogh and Pazzani, 2001] Eamonn Keogh and Michael Pazzani. Derivative dynamic time warping. *SIAM Conference on Data Mining*, 2001.

[Kergosien *et al.*, 1994] Yannick L. Kergosien, Hironobu Gotoda, and Tosiyasu L. Kunii. Bending and creasing virtual paper. *IEEE Computer Graphics and Applications*, 14(1), 1994.

[Kim *et al.*, 2006] Yootai Kim, Raghu Machiraju, and David Thompson. Path-based Control of Smoke Simulations. *Symposium on Computer Animation*, 2006.

[Kraevoy *et al.*, 2008] Vladislav Kraevoy, Alla Sheffer, Ariel Shamir, and Daniel Cohen-Or. Non-homogeneous resizing of complex models. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 27(5), 2008.

[Kreavoy *et al.*, 2007] Vladislav Kreavoy, Dan Julius, and Alla Sheffer. Model Composition from Interchangeable Components. *Computer Graphics and Applications, Proc. Pacific Graphics*, 2007.

[Krull, 1994] Fred N. Krull. The Origin of Computer Graphics within General Motors. *IEEE Annals of the History of Computing*, 16(3), 1994.

[Kry *et al.*, 2002] Paul Kry, Doug James, and Dinesh Pai. EigenSkin: Real Time Large Deformation Character Skinning in Hardware . *Symposium on Computer Animation*, 2002.

[Lander, 1999] Jeff Lander. Over my dead, polygonal body. *Game Developer Magazine*, pages 17–22, 1999.

[Lejemble *et al.*, 2015] Thibault Lejemble, Amélie Fondevilla, Nicolas Durin, Thibault Blanc-Beyne, Camille Schreck, Pierre-Luc Manteaux, Paul Kry, and Marie-Paule Cani. Interactive procedural simulation of paper tearing with sound. *Motion in Games*, 2015.

[Lewis *et al.*, 2000] J. P. Lewis, Matt Cordner, and Nickson Fong. Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. *ACM SIGGRAPH*, 2000.

[Lipşa *et al.*, 2012] Dan Lipşa, Robert Laramee, Simon Cox, Jonathan Roberts, Rick Walker, Michelle Borkin, and Hanspeter Pfister. Visualization for the physical sciences. *Computer Graphics Forum*, 31(8), 2012.

[Lipp *et al.*, 2011] Markus Lipp, Daniel Scherzer, Peter Wonka, and Michael Wimmer. Interactive Modeling of City Layouts using Layers of Procedural Content. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 30(2), 2011.

[Lipson and Shpitalni, 1996] H. Lipson and M. Shpitalni. Optimization-based reconstruction of a 33D object from a single freehand line drawing. *Computer Aided Design*, 28(8), 1996.

[Liu *et al.*, 2015] Han Liu, Ulysse Vimont, Michael Wand, Marie-Paule Cani, Stefanie Hahmann, Damien Rohmer, and Niloy J. Mitra. Replaceable Substructures for Efficient Part-Based Modeling. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 34(2), 2015.

[Longay *et al.*, 2012] Steven Longay, Adam Runions, Frédéric Boudon, and Przemyslaw Prusinkiewics. TreeSketch: Interactive Procedural Modeling of Trees on a Tablet. *Sketch-Based Interfaces and Modeling*, 2012.

[Ma *et al.*, 2014] Minhua Ma, Lakhmi Jain, and Paul Anderson. *Virtual, Augmented Reality and Serious Games for Healthcare*. Springer, 2014.

[Macêdo *et al.*, 2011] I. Macêdo, J. Gois, and L. Velho. Hermite Radial Basis Functions Implicits. *Computer Graphics Forum*, 30(1), 2011.

[Madill and Mould, 2013] Jamie Madill and David Mould. Target Particle Control of Smoke Simulation. *Graphics Interface*, pages 125–132, 2013.

[Magnenat-Thalmann *et al.*, 1988] Nadia Magnenat-Thalmann, Richard Laperrière, and Daniel Thalmann. Joint-dependent local deformations for hand animation and object grasping. *Graphics Interface*, 1988.

[Manteaux *et al.*, 2016] Pierre-Luc Manteaux, Ulysse Vimont, Chris Wojtan, Damien Rohmer, and Marie-Paule Cani. Space-Time Sculpting of Liquid Animation. *Motion In Games*, 2016.

[McAdams *et al.*, 2011] Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. Efficient elasticity for character skinning with contact and collisions. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 30(4), 2011.

[Mitra *et al.*, 2010] Niloy Mitra, Alex Bronstein, and Michael Bronstrein. Intrinsic Regularity Detection in 3D Geometry. *ECCV*, 2010.

[Mitra *et al.*, 2013] Niloy Mitra, Michael Wand, Hao Zhang, Daniel Cohen-Or, Vladimir Kim, and Qi-Xing Huang. Structure-aware shape processing. *ACM SIGGRAPH Asia Course Notes*, 2013.

[Mohr and Gleicher, 2003] Alex Mohr and Michael Gleicher. Building efficient, accurate character skins from example. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 22(3), 2003.

[Mori and Igarashi, 2007] Y. Mori and T. Igarashi. Plushie: an interactive design system for plush toys. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 26(3), 2007.

[Müller *et al.*, 2003] Matthias Müller, David Charypar, and Markus Gross. Particle-Based Fluid Simulation for Interactive Applications. *Symposium on Computer Animation*, 2003.

[Narain *et al.*, 2013] Rahul Narain, Tobias Pfaff, and James O'Brien. Folding and Crumpling Adaptive Sheets. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 32(4), 2013.

[Nielsen and Bridson, 2011] Michael Nielsen and Robert Bridson. Guide Shapes for High Resolution Naturalistic Liquid Simulation. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 30(4), 2011.

[Nielsen *et al.*, 2009] Michael Nielsen, Brian Christensen, Nafees Bin Zafar, Doug Roble, and Ken Museth. Guiding of Smoke Animations Through Variational Coupling of Simulations at Different Resolutions. *Symposium on Computer Animation*, 2009.

[O'Brien and Hodgins, 1999] James O'Brien and Jessica Hodgins. Graphical modeling and animation of brittle fracture. *ACM SIGGRAPH*, 1999.

[O'Brien *et al.*, 2001] J. O'Brien, C. Shen, and G. Essl. Synthesizing sounds from physically based motion. *Proc. ACM SIGGRAPH*, 2001.

[O'Keffe, 1994] R. O'Keffe. Modeling the tearing of paper. *American Journal of Physics*, 62(4), 1994.

[Opstal *et al.*, 2014] Baptiste Van Opstal, Lucas Janin, Ken Museth, and Mihai Aldén. Large Scale Simulation and Surfacing of Water and Ice Effects in Dragons 2. *ACM SIGGRAPH Talk*, 2014.

[Owada *et al.*, 2006] Shigeru Owada, Frank Nielsen, and Takeo Igarashi. Copy-Paste Synthesis of 3D Geometry with Repetitive Patterns. *LNCS, Smart Graphics*, 4073, 2006.

[Palombi *et al.*, 2014] Olivier Palombi, Frederico Ulliana, Valentin Favier, Jean-Claude Léon, and Marie-Christine Rousset. My Corporis Fabrica: an ontology-based tool for reasoning and querying on complex anatomical models. *Journal of Biomedical Semantics*, 5(20), 2014.

[Pan *et al.*, 2013] Zherong Pan, Jin Huang, Yiying Tong, Changxi Zheng, and Hujun Bao. Interactive localized liquid motion editing. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH Asia*, 32(6), 2013.

[Patkar *et al.*, 2015] Saket Patkar, Ning Jin, and Ronald Fedkiw. A new sharp-crease bending element for folding and wrinkling surfaces and volumes. *Symposium on Computer Animation*, 2015.

[Pérez and Suàrez, 2007] Francisco Pérez and José Suàrez. Quasi-developable BB-spline surfaces in ship hull design. *Computer-Aided Design*, 39(10), 2007.

[Peternell, 2004] M. Peternell. Developable surface fitting to point clouds. *Computer Aided Geometric Design*, 2004.

[Pfaff *et al.*, 2014] T. Pfaff, R. Narain, J. De Joya, and J. O'Brien. Adaptive tearing and cracking of thin sheets. *ACM Transactions on Graphics, Proc. SIGGRAPH*, 33(4), 2014.

[Popa *et al.*, 2009] Tiberiu Popa, Quingnan Zhou, Derek Bradley, Vladislav Kraevoy, Hongbo Fu, Alla Sheffer, and Wolfgang Heidrich. Wrinkling captured garments using space-time data-driven deformation. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 28(2), 2009.

[Pottmann *et al.*, 2007] Helmut Pottmann, Andreas Asperl, Michael Hofer, and Axel Kilian. *Architectural Geometry*. Bentley Institute Press, 2007.

[Rabattu *et al.*, 2015] Pierre-Yves Rabattu, Benoit Massé, Federico Ulliana, Marie-Christine Rousset, Damien Rohmer, Jean-Claude Léon, and Olivier Palombi. My Corporis Fabrica Embryo: An ontology-based 3D spatio-temporal modeling of human embryo development. *Journal of Biomedical Semantics*, 6(36), 2015.

[Raveendran *et al.*, 2012] Karthik Raveendran, Nils Thuerey, Chris Wojtan, and Greg Turk. Controlling liquids using meshes. *Symposium on Computer Animation*, 2012.

[Reish *et al.*, 2016] Jon Reish, Stephen Marshall, Magnus Wrenninge, Tolga Göktekin, Michael Hall, Michael O'Brien, Jason Johnston, Jordan Rempel, and Andy Lin. Simulating Rivers in The Good Dinosaur. *ACM SIGGRAPH Talk*, 2016.

[Ricci, 1973] A. Ricci. A constructive geometry for computer graphics. *Computer Journal*, 1973.

[Rivers *et al.*, 2010] Alec Rivers, Frédo Durand, and Takeo Igarashi. 3D Modeling with Silhouettes. *ACM Transactions On Graphics, Proc. ACM SIGGRAPH*, 29(4), 2010.

[Robson *et al.*, 2011] C. Robson, R. maharika, A. Sheffer, and N. Carr. Context-aware garment modeling from sketches. *Computer & Graphics, Proc. Shape Modeling International*, 35, 2011.

[Rohmer *et al.*, 2009] Damien Rohmer, Stefanie Hahmann, and Marie-Paule Cani. Exact volume preserving skinning with shape control. *Symposium on Computer Animation*, 2009.

[Rohmer *et al.*, 2015] Damien Rohmer, Stefanie Hahmann, and Marie-Paule Cani. Real-Time Continuous Self Replicating Details for Shape Deformation. *Computer & Graphics, Proc. Shape Modeling International*, 51, 2015.

[Roman, 2013] Benoît Roman. Fracture path in brittle thin sheets: a unifying review on tearing. *International Journal of Fracture*, 182(2), 2013.

[Rose *et al.*, 2007] Kenneth Rose, Alla Sheffer, Jamie Wither, Marie-Paule Cani, and Boris Thibert. Developable surfaces from arbitrary sketched boundaries. *Symposium on Geometry Processing*, 2007.

[Rosse and Jr, 2003] Cornelius Rosse and José L. Mejino Jr. A reference ontology for biomedical informatics: the Foundational Model of Anatomy. *Journal of Biomedical Informatics*, 36, 2003.

[Ruthenbeck and Reynolds, 2015] G. Ruthenbeck and K. Reynolds. Virtual reality for medical training: the state-of-the-art. *Journal of Simulation*, 9(1), 2015.

[Schpok *et al.*, 2005] Joshya Schpok, William Dwyer, and David S. Ebert. Modeling and Animating Gases with Simulation Features. *Symposium on Computer Animation*, 2005.

[Schreck *et al.*, 2015] Camille Schreck, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani, Shuo Jin, Charlie C.L. Wang, and Jean-Francis Bloch. Non-Smooth Developable Geometry for Interactively Animating Paper Crumpling. *ACM Transactions on Graphics*, 35(1), 2015.

[Schreck *et al.*, 2016] Camille Schreck, Damien Rohmer, Doug James, Stefanie Hahmann, and Marie-Paule Cani. Real-time sound synthesis for paper material based on geometric analysis. *Symposium on Computer Animation*, 2016.

[Schreck *et al.*, 2017] Camille Schreck, Damien Rohmer, and Stefanie Hahmann. Interactive Paper Tearing. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 36(2), 2017.

[Sheffer *et al.*, 2005] Alla Sheffer, Bruno Lévy, Maxim Mogilnitsky, and Alexander Bogomyakov. ABF++ : Fast and Robust Angle Based Flattening. *ACM Transactions on Graphics*, 24(2), 2005.

[Shi and Yu, 2005] Lin Shi and Yizhou Yu. Taming liquids for rapidly changing targets. *Symposium on Computer Animation*, 2005.

[Sifakis *et al.*, 2007] Eftychios Sifakis, Kevin Der, and Ronald Fedkiw. Arbitrary Cutting of Deformable Tetrahedralized Objects. *Symposium on Computer Animation*, 2007.

[Sirin *et al.*, 2007] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A Practical OWL-DL Reasoner. *Journal of Web Semantics*, 5(2), 2007.

[Solomon *et al.*, 2012] Justin Solomon, Etienne Vouga, Max Wardetsky, and Eitan Grinspun. Flexible Developable Surfaces. *Symposium on Geometry Processing*, 2012.

[Sorkine and Alexa, 2007] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. *Symposium on Geometry Processing*, 2007.

[Stam, 1999] Jos Stam. Stable Fluids. *ACM SIGGRAPH*, 1999.

[Stanculescu *et al.*, 2011] Lucian Stanculescu, Raphaelle Chaine, and Marie-Paule Cani. Freestyle: Sculpting meshes with self-adaptive topology. *Computer and Graphics, Proc. Shape Modeling International*, 2011.

[Stiny and Gips, 1972] George Stiny and James Gips. Shape Grammars and the Generative Specification of Painting and Sculpture. *Information Processing*, 1972.

[Sumner and Popovic, 2014] Robert W. Sumner and Jovan Popovic. Deformation Transfer for Triangle Meshes. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 23(3), 2014.

[Tang and Chen, 2009] K. Tang and M. Chen. Quasi-Developable Mesh Surface Interpolation via Mesh Deformation. *IEEE Transactions on Visualization and Computer Graphics*, 2009.

[Tang and Wang, 2005] K. Tang and C. Wang. Modeling Developable Folds on a Strip. *Journal of Computing and Information Science*, 5, 2005.

[Taubin, 1995] Gabriel Taubin. A Signal Processing Approach To Fair Surface Design. *ACM SIGGRAPH*, pages 351–358, 1995.

[Tessendorf, 2004] Jerry Tessendorf. Simulating Ocean Surface. *ACM SIGGRAPH Course Notes*, 2004.

[Treuille *et al.*, 2003] Adrien Treuille, Antoine McNamara, Zoran Popović, and Jos Stam. Keyframe control of smoke simulations. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 22(3), 2003.

[Turquin *et al.*, 2007] E. Turquin, J. Wither, L. Boissieux, M.-P. Cani, and J. Hughes. A sketch-based interface for clothing virtual characters. *IEEE Computer Graphics & Applications*, 27, 2007.

[Umetani *et al.*, 2012] Nobuyuki Umetani, Takeo Igarashi, and Niloy Mitra. Guided Exploration of Physically Valid Shapes for Furniture Design. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 31(4), 2012.

[Vaillant *et al.*, 2013] Rodolphe Vaillant, Loic Barthe, Gael Guennebaud, Marie-Paule Cani, Damien Rohmer, Brian Wyvill, Olivier Gourmel, and Mathias Paulin. Implicit Skinning: Real-Time Skin Deformation with Contact Modeling. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH*, 32(4), 2013.

[Vaillant *et al.*, 2014] Rodolphe Vaillant, Gaël Guennebaud, Loïc Barthe, Brian Wyvill, and Marie-Paule Cani. Robust Iso-Surface Tracking for Interactive Character Skinning. *ACM Transactions on Graphics, Proc. ACM SIGGRAPH Asia*, 33(6), 2014.

[van den Doel *et al.*, 2001] Kees van den Doel, Paul G. Kry, and Dinesh K. Pai. FoleyAutomatic: Physically-based Sound Effects for Interactive Simulation and Animation. *Proc. ACM SIGGRAPH*, 2001.

[Vimont *et al.*, 2017] Ulysse Vimont, Damien Rohmer, and Marie-Paule Cani. Deformation Grammars: Hierarchical Constraints Preservation under Deformation. *Computer Graphics Forum*, 2017.

[von Funck *et al.*, 2008] Wolfram von Funck, Holger Theisel, and Hans-Peter Seidel. Volume-preserving Mesh Skinning. *Vision, Modeling and Visualization*, 2008.

[Wang and Tang, 2004] Charlie C. Wang and Kai Tang. Achieving developability of a polygonal surface by minimum deformation: a study of global and local optimization approaches. *The Visual Computer*, 20(8), 2004.

[Wang, 2008] Charlie Wang. Towards Flattenable Mesh Surfaces. *Computer Aided Design*, 40, 2008.

[Xu *et al.*, 2014] Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. True2form: 3d curve networks from 2d sketches via selective regularization. *ACM Transactions on Graphics, Proc. SIGGRAPH*, 33(4), 2014.

[Yuan *et al.*, 2011] Zhi Yuan, Fan Chen, and Ye Zhao. Pattern-guided smoke animation with lagrangian coherent structure. *ACM Transactions on Graphics, ACM SIGGRAPH Asia*, 30(6), 2011.

[Zheng *et al.*, 2011] Youyi Zheng, Hongbo Fu, Daniel Cohen-Or, Oscar Kin-Chung Au, and Chiew-Lan Tai. Component-wise controllers for structure-preserving shape manipulation. *Computer Graphics Forum, Proc. EUROGRAPHICS*, 30(2), 2011.

[Zhu and Bridson, 2005] Yongning Zhu and Robert Bridson. Animating sand as a fluid. *ACM Transactions on Graphics, Proc. of ACM SIGGRAPH*, 24(3), 2005.