EMOTIVE
Storytelling for cultural heritage

www.emotiveproject.eu

D6.2 Low-Cost Physical Artefact Reproduction

Paolo Cignoni[1], Luigi Malomo[1], Gianpaolo Palma[1], Fabio Ganovelli[1], Roberto Scopigno[1], Manuele Sabbadin[1]

Reviewed by:
Alejandra Barragán (DXT), Rémi Courtel (DXT), Julien Philip (INRIA)

Abstract
This document describes a new casting approach we have developed for the purpose of providing a fabrication technique for the in house reproduction of physical artefacts; the proposed approach can be cheaper than traditional production processes (including 3D printing), and it can easily scale to the production of hundreds of copies, quantities that can be expected to be distributed in museum applications. The approach is based on a traditional casting approach enhanced by automated design techniques that can make this technique practical and cost effective.
We have demonstrated the approach on a digitized artefact creating the mold for the fabrication of the replica of the 8000 years old *Female Figurine* that was found in Çatalhöyük.

Official Submission Date: 31/10/2017
Actual Submission Date: 15/11/2017
Dissemination Level: PU

Partner	Estimated Effort (in PMs)
[1]CNR	6

A 3-year EC Funded project
Start date: 1st November 2016
Funding: This project has received funding from the European Union's Horizon 2020 research and innovati on programme under grant agreement No 727188.

TABLE OF CONTENTS

1 Executive summary

Among EMOTIVE's aims, we have the objective of evaluating how new fabrication technologies can enhance visitor's experience, both on-site and off-site. In particular, we believe that the actual, physical reproductions of artefacts that visitors can freely handle are great facilitators to encourage emotional engagement. The planned digital extension of the museum experience online and on-site can benefit from the distribution of artefact reproductions. These reproductions act simultaneously as souvenirs and as prompts to continue the story and allow the creation of personalized/adaptive experiences.

Current technologies for producing a small number of physical artefacts are either costly (like using a servicing company) or slow (by using for example traditional 3D printing technologies). These constraints limit the usage or the free experimentation of engaging museum users through means such as physical copies or customization of site artefacts. Traditional gypsum or resin casting techniques are simple, cheap and safe enough to be used by low skilled personnel for the production of replicas of a single object. On the other hand, using casting techniques requires the design of the casting mold, e.g., the shape where you pour the casting liquid, task that is still a daunting problem to create general 3D shapes and that requires highly skilled technicians. We recently developed FlexMolds [Malomo et al. 2016], a new algorithmic approach that allows to automatically perform this mold design and we have engineered this approach in a set of innovative tools that allows a user with basic 3D knowledge to automatically generate the required molds.

This document will explain the technical aspects of the mold generation technique (Section 2) and will then describe (Section 3) how to use the provided tools. We will illustrate the explanation of the provided tools with an actual example: the mold generation for the creation of the cast replica of the 8000 years old Female Figurine that was found in Çatalhöyük.

2 Description of the mold generation technique

Mold casting is a widely used manufacturing process for rapidly producing shapes. It allows the efficient shaping of a wide range of materials, such as resins, and scales well with the number of required copies. However, reusable molds come with the requirement of being detachable from the actual molded part without physically destroying the mold or the molded part. This poses hard constraints on reproducible shapes. For rigid molds, the cavity of a mold piece must resemble a height field. This results in a trade-off between reproducible shape complexity and the number of required mold pieces. Each mold piece should be designed by keeping in mind the complete movement that is needed to separate it from the cast object. Eventually, shapes with high genus or very complex geometries may require to be split into multiple pieces, which can be individually cast and assembled afterwards. In general, automatic mold design is a very complex task since it must satisfy multiple non-local constraints.

A different strategy is silicone mold casting. An object is surrounded by silicone, and after curing, the silicone is cut out and detached from the original object, obtaining a flexible mold. This allows the creation of stunning replicas of objects with highly complex geometry.

However, so far this has been a labor-intensive manual process, requiring a positive copy of the shape, performing a silicone cast, and cutting the silicone mold. Furthermore, in practice, deciding and performing these manual cuts is challenging, especially in the presence of complex shapes and topologies.

Inspired by this approach, we propose a novel reproduction approach for highly detailed free-form shapes, based on flexible mold shells (FlexMolds). FlexMolds are made of a thin but still sufficiently shape-preserving shell of deformable material, which can be printed by using a commercial 3D printing device. Thanks to their flexibility, they provide more freedom during the removal process, allowing the reproduction of complex shapes, often even with just a single mold piece. This fabrication method inherits the main advantages of reusable mold casting manufacturing techniques: the mold shell can be reused for multiple copies; the fabrication process is fast and cheap compared to 3D printing; there is a wide range of manufacturing materials and colors available.

FlexMolds are designed with a sufficient number of cuts that allows the extraction of the internal object once the cast liquid is solidified, without damaging the mold. A proper cut layout should be designed not only to guarantee local detachment, but also to allow the mold to completely depart from the cast object. This may include complex, global, large-scale deformations that allow the flexible mold to adapt to geometric constraints during the extraction. To guarantee the existence of a feasible extraction sequence we simulated the entire extraction process. Specifically, we are interested in ensuring the existence of a feasible extraction sequence that keeps the deformation within a certain range.

Ideally, we would like to use as few and as short cuts as possible. However, optimizing cut design is not a well-posed problem, i.e., the extraction sequence does not change continuously with respect to the cut.

2.1 Background on mold design techniques

In recent years, the computer graphics community has contributed with many shape-processing approaches and design tools for computing physically realizable objects. While a comprehensive overview can be found in [Umetani et al. 2015; Liu et al. 2014] and, the following sections will focus on setting our work in context with closely related work.

Mold design Molding is a widely used manufacturing process for rapidly producing shapes. While ancient molds, for example, for creating clay figurines or spear heads, are proof that the technique has existed for thousands of years, mold design is still a highly challenging and important topic in engineering. Expendable mold casting allows the reproduction of shapes of similar complexity as 3D printing with a wide range of materials beyond readily available 3D printing materials [Singh 2009] and is used, for example, for wax casting of digitally designed jewelry [Wannarumon 2011]. In our work we focus on non-expendable mold casting, i.e., molds that can be reused for multiple production cycles. An introduction to manufacturing processes can be found in [Temple Black et al. 2013].

Reusable molds are usually assumed to be made out of a rigid material, such as metal, which restricts the complexity of shapes that can be manufactured as the shape needs to be removable from the mold without damaging them. Therefore, the parting direction and parting surfaces of molds play an important role. Chakraborty and Reddy [Chakraborty and Venkata Reddy 2009] proposed a joint optimization to minimize the area of undercuts, the flatness of the parting surface, and minimized draw depth for determining the best parting direction, parting line, and surface for a two-piece molded component. Zhang et al. [Zhang et al. 2009] performed a geometric analysis of the shape to identify potential undercuts with their possible withdrawal directions; this analysis provides decision support information for designers choosing parting direction, parting lines, and surfaces. For more complex models, Lin and Quang [Lin and Quang 2014] proposed a heuristic to segment the surface of an object and compute feasible parting directions for automatically generating multi-piece molds, and [Hu et al. 2014] explored the decomposition of shapes into pyramidal pieces. To address the problem that complex geometries might require an extensive number of mold parts, Herholz et al. [Herholz et al. 2015] proposed a method for approximating free-form geometry with height fields in which they deform the shape slightly in order to meet the height field constraints needed for rigid casting. Thermoforming techniques may accurately reproduce shape and color [Schüller et al. 2016]. While these methods are suitable for large-scale production, their use is limited for the fabrication of simple geometric shapes. In our technique, by exploiting the properties of flexible molds, we follow a very different strategy that does not require any changes to the original geometry and is able to manage objects with complex geometry and topology.

As already introduced, silicone molds are a known technique art reproduction, made by surrounding an object with silicone and manually cutting, they have the potential to reproduce highly complex shapes. On the other hand, so far the design of such molds relies on a manual process and well-trained users.

Simulation and motion planning A variety of methods have been presented for simulating deformable objects and shells based, for example, on nonlinear finite elements [Sifakis and Barbic 2012], discrete elastic shells [Grinspun et al. 2003], mass-spring systems, co-rotated linear finite elements [Müller et al. 2008], or position-based dynamics [Müller et al. 2007; Bender et al. 2014]. The chosen method is usually the result of a careful balance between required accuracy, speed, and robustness.

Solving the inverse problem, i.e., what forces or displacements are required to achieve a desired goal position, is highly challenging. Methods were proposed for motion planning for robotic manipulation and the assembly planning of deformable objects, such as ropes and wires [Saha and Isto], sheets, and elastic volumes [Jiménez 2012]. These strategies are often defined for relatively simple polyhedral environments and do not extend directly to our case, in which we are facing the removal process of a tightly fitting elastic shell from an object with complex geometric features and topology.

While generally important, we consider the simulation of the material flow during the casting process out of the scope of our technique and restrict ourselves to low-viscous resins that do not require to address this problem.

Shape decomposition and approximation for fabrication In the broader context, numerous algorithms for shape decomposition and fabrication have been proposed in computer graphics. For example, recent work suggested strategies for partitioning models into smaller parts, relevant for packing large objects into 3D print volumes [Luo et al. 2012; Vanek et al. 2014; Chen et al. 2015; Yao et al. 2015].

To approximate the surface with a small number of planar polygonal primitives, Chen et al. [Chen et al. 2013] iteratively assigned mesh faces to planar segments. This results in a closed intersection-free mesh, which can be augmented with internal connectors and fabricated. Relaxing the constraint to developable patches, D-Charts [Julius et al. 2005] provide a quasi-developable mesh segmentation, while Tang et al. [Tang et al. 2016] and Solomon et al. [Solomon et al. 2012] provided interactive modeling systems for developable surfaces that can be used for model fabrication from sheets of material. Skouras et al. [Skouras et al. 2012] included the effects of air pressure to design inflatable structures.

Starting from a surface model, Hildebrand et al. [Hildebrand et al. 2012] and Schwartzburg and Pauly [Schwartzburg and Pauly 2013] computed interlocking planar elements that resemble an illustrative approximation of the desired shape. Cignoni et al. [Cignoni et al. 2014] extended this concept and placed the elements driven by a feature-aligned input cross-field [Ray et al. 2009; Bommes et al. 2009], a mathematical entity that catches the overall structure and curvature of the object. Inspired by this concept, we compute potential candidates for cuts along smooth polylines that emerge from a field-aligned patch layout decomposition of the initial mesh [Tarini et al. 2011; Campen et al. 2012; Myles et al. 2014; Razafindrazaka et al. 2015].

2.2 Designing FlexMolds

Given an object represented by a manifold, watertight, triangulated surface S, we target the problem of the automatic design of a flexible thin mold M that can be used for liquid casting. Regarding the more commonly faced problem of rigid casting, allowing the deformation of the mold makes it possible to remove more complex non-height-field shapes without requiring intricate fine-grained mold decomposition or the modification of the original object shape. Our flexible molds are usually very thin (in the order of 2-3 mm) and can be composed of a few pieces or even a single shell that can be unwrapped/detached from the cast object, exploiting the elasticity of the mold material.

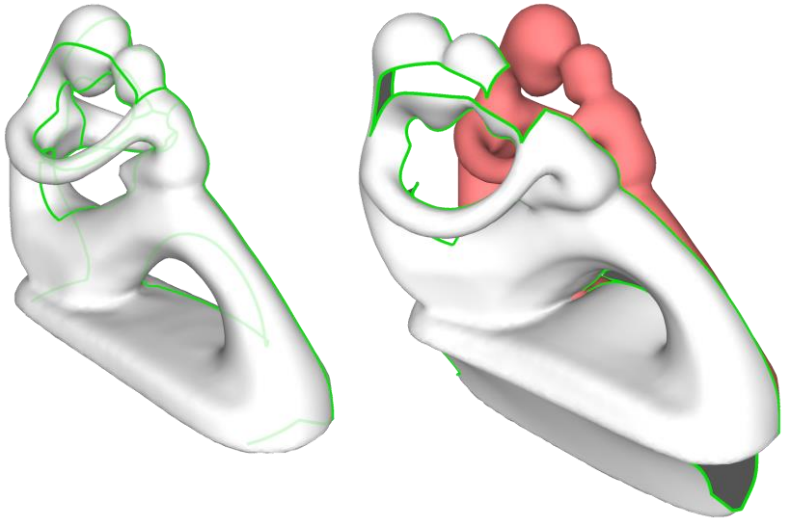


Figure 1: *A cut layout X (in green) characterizes a thin flexible mold M (white), determining how it can be opened and detached from the object surface S (in red).*

Under these assumptions, the main characterizing feature of a mold for a surface S is the cut layout X over S that determines how the mold M_X (which is the mold M, equipped with the cut X) decomposes and opens into one or more patches p_i. Figure 1 illustrates these concepts, showing a cut layout X (in green) of a flexible mold M (white) and shows how X rules how M can be opened and detached from the surface S of the cast object (in red). The other geometric characteristics of the mold (e.g., its thickness and the actual 3D profile of the cut) can be inferred starting from X and S and are discussed in the Section 2.3.

To understand if a cut layout X can generate a feasible mold M_X, we test if M_X can be successfully removed from the cast surface S without suffering high strains. This requires solving the following two problems:

Extraction movement Given a flexible mold M wrapped around an arbitrarily complex object S, we want to find a dynamic, temporally varying, force field $F_M(t)$ that, once applied to M, detaches it from S in a finite time and without breaking it, or in practice, without imposing excessive strain over the mold during this process. We postulate that the complete solution to this problem, given the similarity with the computational hardness of motion planning [Hwang et al. 1992], probably lies in the realm of intractable

problems, and we propose a practical heuristic for that (Section 2.2.5).

Strain evaluation Assuming we know how to drive the removal of a mold, we want to evaluate how this process affects the mold itself in terms of the strain suffered during the extraction: we have to ensure that for multiple uses of the same mold, it will not break when the object is extracted (Section 2.2.4).

These two interconnected problems are specific to the case of flexible molds and, to our knowledge, have never been faced before. In this paper, we define that a cut layout X is *feasible* if, solving the two problems above, we find that the mold M_X can be removed from $S$and during this process the strain is below a certain threshold.

One very important characteristic of flexible molds is that their thickness can be considered negligible in many parts of the feasibility evaluation process. For example, if a cut layout is composed of multiple pieces (as the one shown in Figure 2, left), we can assume that we can evaluate the feasibility of each piece independently, expecting that, given the small thickness of the mold and the flexibility of the material, the patches do not impede/obstruct each other.

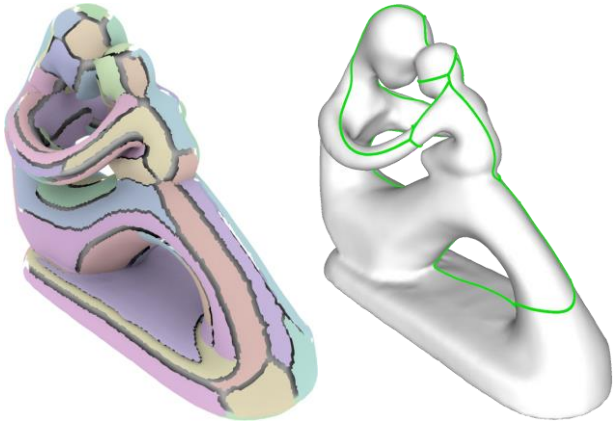


Figure 2: *Left, a cut layout obtained using the approach in [Cohen-Steiner et al. 2004]. Right, a cut layout that opens the mold to a disk is not sufficient to ensure its full removal because of the high deformation induced by the extraction process.*

2.2.1 Searching for Good Cut Layouts

As introduced in the previous section, the shape of the cut layout X is the main feature defining a mold.

To obtain a cut layout, we initially tried to decompose the mesh in multiple pieces by using a purely geometric approach that clusters portions of the surface with similar normals [Cohen-Steiner et al. 2004] (see Figure 2, left). Unfortunately, we observed that this approach, when applied to arbitrarily complex geometry, may decompose the mold into an overly high number of disconnected pieces, making the manual assembly of the mold practically infeasible. Conversely, introducing just the minimal amount of cuts to open the mesh to a topological disk [Dey 1994] may result in a configuration that does not allow for extraction due to excessive deformation (see Figure 2, right). This motivates the need for a method to adapt an initial cut layout, either by removing or introducing more cuts.

With our technique, we want to derive a cut that, while being sufficiently short and simple to generate a practically feasible mold, it also allows generating a mold that can be detached from the cast object without suffering excessive deformations.

One possible way to explore the space of possible cut layouts could be a top-down approach in which cuts are grown over the surface starting from an almost closed mold. This class of approaches has been successfully used in texture parametrization to minimize the distortion [Gu et al. 2002], but we found it unsuitable to bootstrap our optimization. In fact, with those approaches in the initial steps, when the cut is short, the resulting deformations and forces required to extract the mold can be arbitrarily high. This

can be a significant problem for two main reasons: first, large deformations require more accurate and costly simulations, making their use in the inner loop of an optimization process impractical; second, we experimentally found that when a high deformation is allowed, our heuristic extraction strategy may generate implausible removal movements.

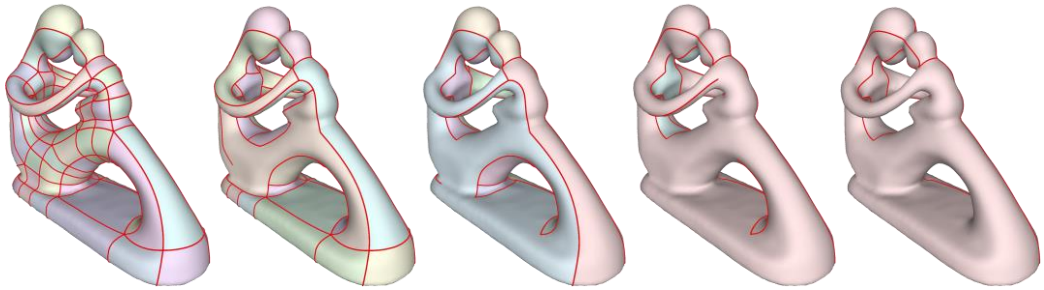


Figure 3: *The bottom-up greedy optimization process. Starting from a dense cut layout (left), generated by a patch decomposition, we iteratively perform operations that remove segments of the initial cut layout, choosing at each step the operation that requires the minimal deformation in the extraction process and stopping when this exceeds a given threshold.*

For these reasons, we opted for a bottom-up approach that stays in the region of feasible cut layouts. As illustrated in Figure 3, we start by partitioning the mesh into a set P of small, simple, disk-like patches p_i whose boundaries provide a dense cut layout X_0. For such a *dense* cut layout, it is easy to test feasibility; all patches are checked individually to know whether they are removable with relatively low deformation. Assuming that this initial condition is fulfilled (i.e., each patch in the initial patch layout is removable), we consider the initial cut layout induced by P as composed of a set of curved polylines χ_{ij} over S that correspond to the portion of boundary shared by two patches p_i, p_j. The main idea is that we incrementally make X sparse in a greedy way by iteratively removing one polyline χ_{ij} at a time. Intuitively, at the beginning of the process, this means merging the two patches p_i, p_j along a common boundary.

2.2.2 Greedy Optimization

The optimization process that generates a good cut is organized as a greedy loop where at each step we remove the best polyline χ from X. This incremental removal process is illustrated in Figure 3 where the curved polylines are the red segments representing the cuts over the surface. Sometime these segments separate different patches (usually at the beginning) while other times they represent cuts along a single patch (usually at the end of the process like in the rightmost part of the figure). A cut removal can involve one or two patches p_k of the mold M_X according to whether before the removal the polyline χ connects two different connected components of M_X or not. At each step of the process, we remove the polyline χ such that the resulting mold can still be extracted with minimal deformation.

We keep all the possible removal operations arranged in a heap where the score function is the maximum deformation found during the feasibility evaluation process described in Section 2.2.4. We continue this greedy process until there is no more feasible operation, or in other words, closing the current cut X by removing a polyline $\chi \in X$ would introduce too much strain or lead to a mold for which we are not able to find a successful extraction movement.

Even if we use a very efficient dynamic simulation system for the feasibility evaluation, it is necessary to keep the number of these tests as low as possible. For this reason, we use the two following lazy update heap strategies.

Mailboxing After each extraction operation that involves the patches p_i, p_j, we should re-evaluate the heap score for all the χ lying on the boundary of p_i, p_j. To avoid all these evaluations, we mark these entries as not updated in the heap. When we extract one of these outdated entries from the heap, we re-

evaluate it and eventually put it in the heap again. Such an approach is a reasonable heuristic because it is highly probable that such scores will increase. In most cases, two patches are more difficult to remove when merged than individually.

Early abort of evaluation The most time-consuming evaluations are the ones that require many steps and large deformations to extract the mold piece. For this reason, we abort the simulation when the deformation grows above a given threshold. This deformation threshold is dynamically increased during the whole greedy optimization loop and is set to be the 30^{th} percentile of the entries in the heap. In other words, we abort all the evaluations that involve deformations higher than the 30 best values in the heap. We mark these entries in the heap, and when they pop out from the heap we proceed as above, re-evaluating and putting them in the heap again. The rationale is to delay the slow evaluations only when the process is close to the end and they are really needed.

This approach guarantees that a feasible cut layout will always be generated. It is based on two prerequisites: we require an initial feasible patch decomposition and a method to evaluate whether a candidate cut layout X allows the resulting mold to be extracted. It should be noted that, while we cannot guarantee that the approach ends up with a single patch, all the experiments we performed generated a mold composed of a single piece. From a topological point of view, note that there is no need for having a mold that is disk-like; in fact, several of the presented results are molds with multiple holes.

2.2.3 Initial Patch Decomposition

Our method requires an initial patch decomposition in which each element of the generated mold can be easily detached from the object surface. There is a vast literature about obtaining patch decompositions through mesh segmentation and partitioning (see [Shamir 2008]), and many methods can be reasonable candidates for generating it. We evaluated different approaches for this task, and the results are discussed in Section 2.4. For our experiments, we chose the quad layout proposed in [Pietroni et al. 2016] that is generated starting from a cross-field over S mostly for two intuitive reasons. First, quad patches are simple enough to guarantee feasibility but sufficiently large to not slow down the greedy optimization process. Second, the edges of this class of quad layouts are aligned with a smoothed version of the surface curvature. This characteristic generates curvature-aligned cut layouts, a feature that, intuitively, strongly resembles the natural way of decomposing a mold; see, for example, the long lines in Figure 4 that align with the overall shape of the model. These lines constitute a reasonable set of cut candidates to open the mold without too much deformation. A more detailed comparison between the different decomposition strategies is shown in Section 2.4.

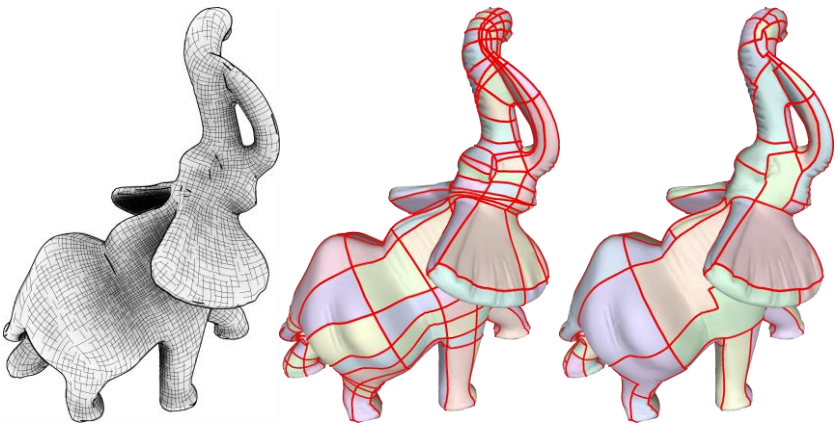


Figure 4: *An example of an input cross-field (left), the resulting quad layout (center), and the merged patch decomposition (right) used as the starting point for the greedy optimization process.*

2.2.4 Evaluation of a Patch Layout

Detaching the mold from an object induces deformations. A single, localized excessive stress in one moment of the extraction might already damage the mold. For this reason, the induced stress must be considered at the local level.

This requires a physically-based simulation of the mold and the derivation of the forces imposed by the user to extract the cast rigid object from the deformable mold surrounding it. While a full FEM simulation would be highly accurate, it comes with significant computational cost. For this reason, we opted for a dynamic simulation and a heuristic to compute a force field that drives the mold off the surface.

We based our dynamic simulation on the projective dynamics method [Bouaziz et al. 2014], which ensures a sufficient degree of accuracy and robustness at relatively low computational costs. We apply this approach on a thin shell model of our mold M_X. The contact between the mold M and the cast object S is simulated using dynamic plane constraints. We uniformly remesh each patch of the initial decomposition to ensure the quality of the triangulation and sufficient degrees of freedom for the physical simulation.

For each step of the extraction simulation, we evaluate the *current* deformation of each triangle. We then record the maximum deformation, defined as the principal stretch of the plane strain, i.e., the maximum singular value obtained from the polar decomposition of the in-plane deformation gradient. Our measure is based on the St. Venant maximum principal strain theory, which assumes that yield occurs when the maximum principal strain reaches the strain corresponding to the yield point during a simple tensile test. However, as we know the deformation field, other measures could be easily employed as well. The quality of a cut is quantified as being inversely proportional to the maximum deformation among all triangles during all the steps of the extraction process.

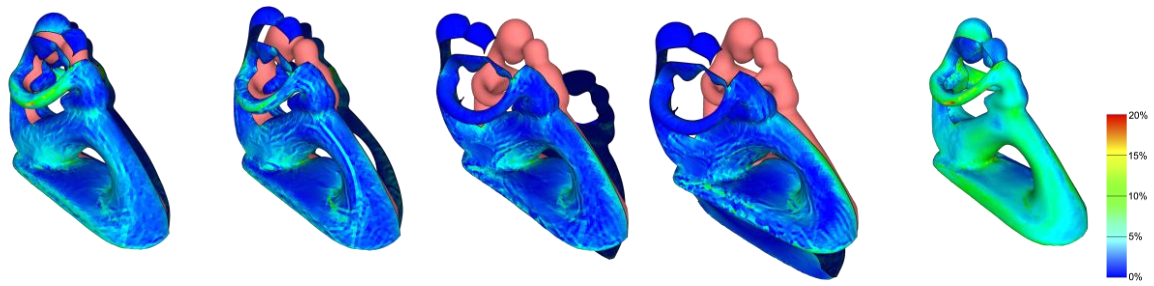


Figure 5: *The feasibility evaluation process, used in the greedy optimization to score candidate cut layouts X, attempts to extract the mold M_X and records the maximum deformation suffered during this process. Deformation is color coded according to the maximum feasible deformation threshold.*

Figure 5 shows some steps of one of these extraction simulations, where the color indicates, for the intermediate four steps on the left, the current deformation of each triangle. The last figure on the right shows the maximum deformation encountered by each triangle during the entire simulation; color coding of the deformation is done according to the maximum acceptable deformation.

2.2.5 Extraction Path

We use a dynamic system in which the extraction of a mold patch is simulated by applying a force field $F_M(t)$ to each vertex. The force field $F_M(t)$ deforms and drives the extraction from the cast model. It is defined as the sum of two fields: the *detaching forces* $F^D(S)$, a volumetric force field that pushes the mold piece away from the object surface, and the *moving forces* $F^M(P_k, t)$, an evolving field that depends on the actual shape of each mold piece P_k.

The **Detaching Force** field $F(S)$ is a volumetric static force field that is directed along the negative gradient of the distance field computed for the surface S (see Figure 6). Its main purpose is to push each piece away from the cast object. The magnitude of this force linearly decreases from a fixed maximum on the

surface to zero for all the points with a distance greater than one third of the surface bounding box diagonal. For simple cases, detaching forces may be sufficient to separate the mold from the inner object. Unfortunately, for complex objects the mold can still be stuck in a position that wraps the object, even if "locally detached" from the cast. To overcome this problem, we introduced moving forces.

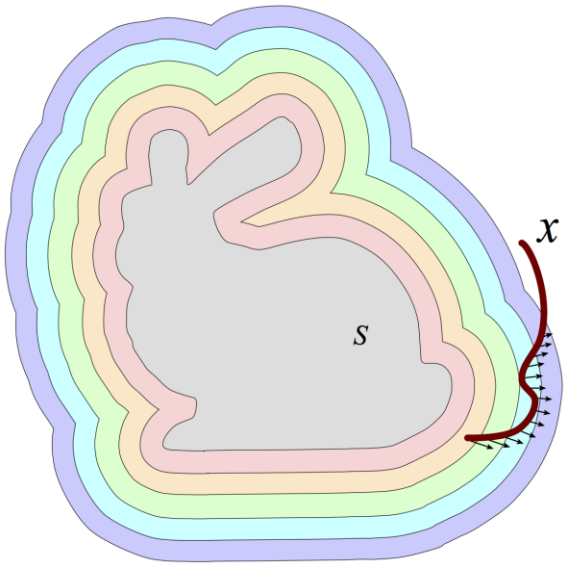


Figure 6: *An illustrative representation of detaching forces. With increasing distance from the surface the forces linearly decrease to zero.*

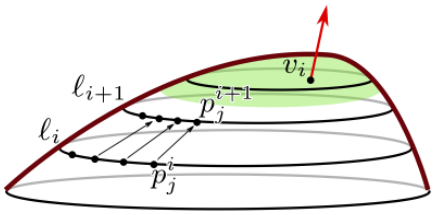


Figure 7: *Moving Forces: for a given mold piece, in the dynamic simulation, we use the transformation τ matching point pairs ${p^i}_j, {p^{i+1}}_j$ lying on the geodesic isolines l_i, l_{i+1} to determine, for each point in the inner (green) part of the patch, the direction of the moving force.*

Moving Forces are defined by a dynamic field that depends on the current shape and position of the mold: intuitively they grab the mold from the farthest region from the cuts, and pull it away, along a single direction, from the cast. For each step of the simulation, given the deformed patch P_k of the mold M at time t, we compute the isolines l_i at an increasing constant geodesic distance from the border (with a step of 1/20 of the bounding box). For each pair of consecutive lines l_i, l_{i+1}, we regularly sample points, and for each point ${p^i}_j$ on l_i, we find the corresponding point ${p^{i+1}}_j$ closest on l_{i+1} along the geodesic gradient direction. We find the rigid transformation ${\tau^t}_k$ that best matches all these point pairs (${p^i}_j \rightarrow {p^{i+1}}_j \ \forall i,j$), and we use it to compute the moving force direction as follows: we consider the set V_k of the 30th percentile vertices v_i farthest from the border and, for each vertex v_i, we use the displacement vector ${\tau^t}_k(v_i) - v_i$ as the moving force direction. The intensity of the force grows linearly with the geodesic distance from zero on the boundary of V_k up to a fixed maximum on the farthest point from the boundary. Figure 7 illustrates this concept and the resulting force direction used to generate the force $F^M(P_k, t)$ applied to one of the points of the set V_k of the inner vertices of the patch (shown in green).

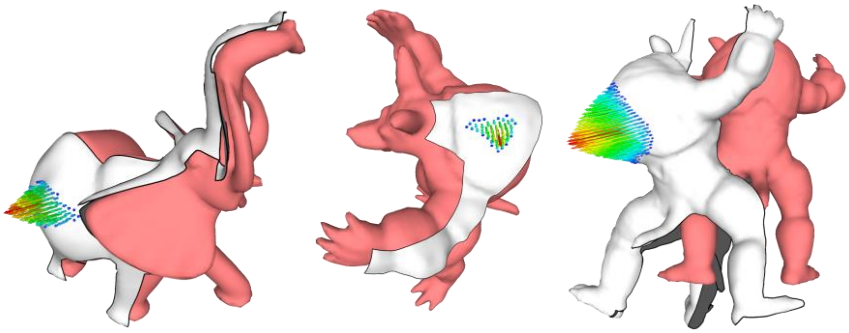


Figure 8: *Moving Forces: a few patches with the resulting moving force directions; each patch $P^t{}_k$ is shown, in its deformed state, at time t. Color and length of the arrow represent force intensity. The first two examples on the left show patches that only partially cover the model, while the one on the right covers the whole model.*

Figure 8 shows a few patches in their deformed state during the dynamic simulation and the moving force direction applied on their vertices. For the sake of efficiency, assuming the mold piece will suffer only limited deformations, we compute the isolines at rest shape and we track corresponding point pairs during the simulation; then, for each step of the simulation, we only have to update the transformation τ to compute the new forces.

Both detaching and moving forces are essential to derive the extraction sequence in the general case. While detaching force depends only on the shape of the cast object and drive the mold on a global perspective, moving forces depend on the current deformation of the mold and allows the mold to escape from locally intricate situations. We experimentally observed that omitting the detaching forces significantly reduces the convergence speed of the simulation, while removing the moving forces can prevent the mold extraction, even for simple cases.

The evaluation process for a mold piece P_k starts at its rest position. We then start the simulation by applying detaching and moving forces until the mold is completely extracted from the surface S (the cast object). This condition is tested by checking if the intersection between the mold and the cast object's oriented bounding box is zero. If the number of simulation steps exceeds a given threshold, we mark the current mold as non-feasible.

Our strategy is designed to guarantee that the computed extraction sequence can completely remove the cast object from its FlexMold. Our method is capable of leaving additional cuts to lower the deformation due to the extraction process or to allow the mold to bend and pass more easily through narrow passages during the extraction. To achieve this result, the simulation of the entire extraction process is required, including modeling the contact between the FlexMold and the cast object.

2.3 Generating Fabrication-ready FlexMolds

As result of our greedy optimization process, we have a cut layout X defined on the surface S of our starting model. However, to fabricate the final mold M_X, we need to extend it to a 3D solid. Hence, the initial surface is first inflated, then cuts are modeled as solid V-shaped prisms and finally subtracted from the volume by performing boolean operations on the volume.

2.3.1 Placing Air Vents

To ensure proper air escape when pouring the resin inside the mold shell, we have to accurately place small holes in the mold (see Figure 9). A bigger hole is used to pour the resin inside the mold.

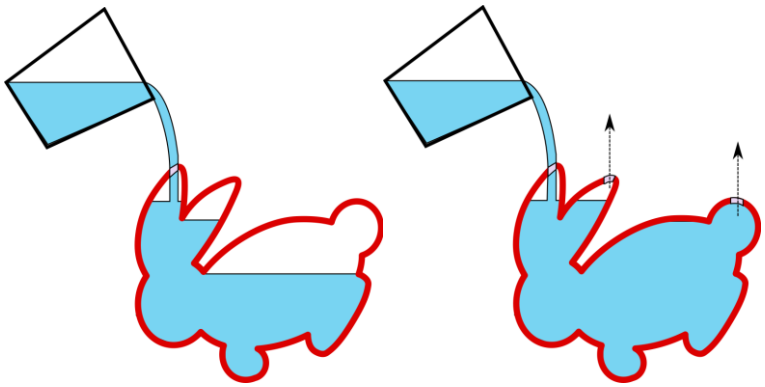


Figure 9: *Diagram illustrating the bubble trap problem.*

Theoretically, for a given mold orientation, every local maximum along the gravity direction should be equipped with a hole for letting the air escape. We select the best orientation by sampling directions on a sphere [Keinert et al. 2015] and pick the one that minimizes the number of required air vents. Due to geometrical noise and high-frequency details, this approach may still find a large number of maxima; however, some of the holes may be practically unnecessary. If we slightly shake the mold before the resin solidifies, the air may escape through nearby maxima. Formally, we can think of shaking being equivalent to varying the mold's vertical direction by an angle α. Following this intuition, we reduce the number of maxima by using a simple strategy.

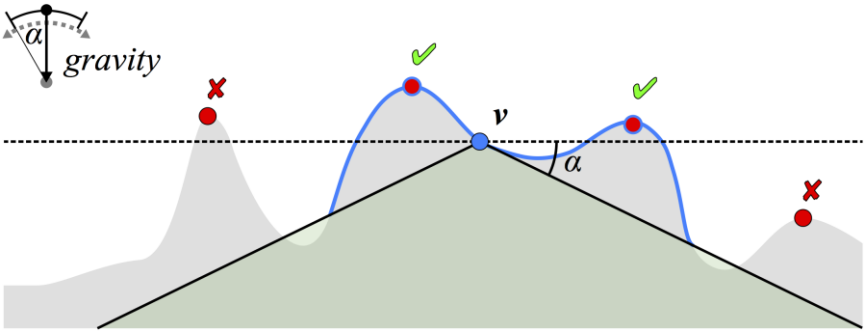


Figure 10: *Diagram illustrating the shaking cone of a vertex.*

For each vertex v, we define a shaking cone with an aperture equal to $\pi - 2\alpha$. We select the portion S_v of the mesh that is above the cone and connected to v and then associate to v all the maxima that are in S_v. Figure 10 shows a vertex v, the shaking cone, the region of S associated to v in blue, and the two maxima that can cover that vertex according to the shaking angle α. At the end of this procedure, each vertex is associated to one or more maxima. We reduce the holes by searching for the smallest set of maxima that cover all vertices. We follow a classical greedy heuristic for this minimum set cover problem. We iteratively add one maximum at a time, favoring the ones that cover the larger uncovered portion of the surface, until the entire mesh is covered. The result of this process on the dragon model is shown in Figure 11.

2.3.2 Using the Mold

During the casting process, the mold has to be properly sealed. Therefore, we print a sequence of small posts along each side of the cut, tie them together using a thin elastic rubber band (as shown in Figure 12), and seal the seams with silicone. While casting, the mold should be kept in the optimized orientation, otherwise the air vents may not work properly. For this purpose, we print a custom lower envelope support (see Figure 13). As a byproduct, this support helps to more evenly distribute the overall

deformation under the pressure of the cast liquid.

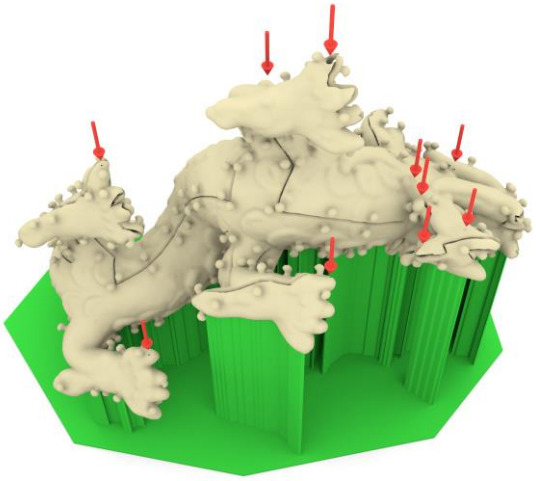


Figure 11: *The holes needed to cast the dragon model after the reduction process has been applied.*

(a) *(b)*

Figure 12: *(a) The mold is tied together before the resin is poured; (b) the result of the cast of the elephant model.*

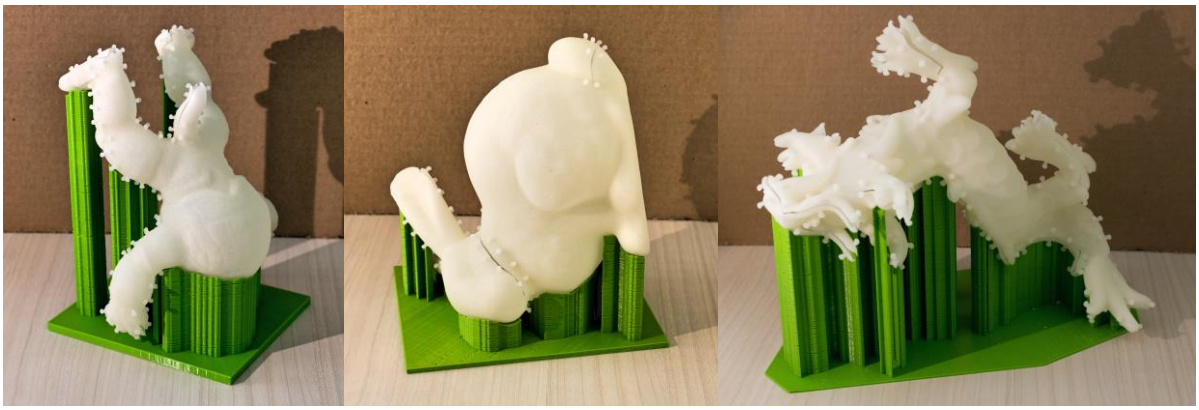


Figure 13: *To keep the models in their correct position, some simple 3D printed supports are used.*

2.4 Results

We evaluated our approach by generating cut layouts on a wide range of different shapes and initial partitionings. Then we actually 3D printed several of the most interesting FlexMolds and used them in practice to cast multiple copies of objects. In the following, we first validate some of our design choices and then demonstrate the effectiveness of our approach by showcasing several complex models that can be easily reproduced using our method.

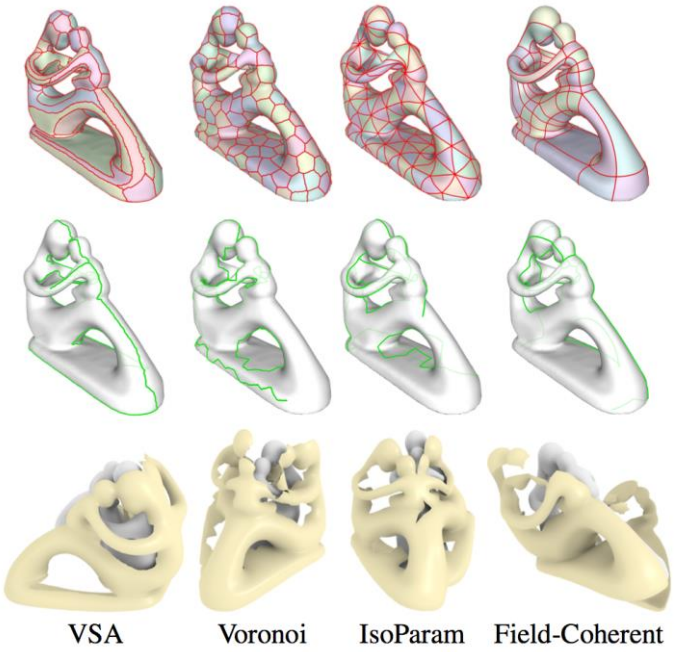


Figure 14: *Initial patch layout, the produced final cut, and a step of the extraction procedure for the approaches proposed in [Cohen-Steiner et al. 2004], [Lévy 2014], [Pietroni et al. 2010], and [Pietroni et al. 2016].*

Initial patch layout Our approach for generating a proper cut layout works with any reasonable initial patch layout. Figure 14 shows the result of our algorithm when applied to different initial partitionings. The first row shows the initial partitionings we obtained by using Variational Shape Approximation [Cohen-Steiner et al. 2004], a Voronoi-based mesh partitioning with Lloyd relaxation [Lévy 2014], the Almost Isometric patch layout of [Pietroni et al. 2010], and the field-coherent quad patch layout decomposition method proposed in [Pietroni et al. 2016]. The second row shows the final cut layout, while the final row shows a snapshot of the extraction sequence. To evaluate the impact of different initial partitioning on the final cut layout, we measured how the max stretch induced by the extraction grows with the shortening of the cut. We used the cut length as an intuitive measure of the quality of a layout. If two layouts allow the extraction of their corresponding molds with the same deformation and arguably a similar effort, the shorter one must be preferred for practical reasons (easier to be closed and sealed). Figure 15 plots the relation between cut length and maximum deformation for the extraction during the optimization process. It has to be noted that the relation is not always monotone. Sometimes a change of topology may force the detaching procedure to choose a significantly different extraction path, resulting in an abrupt reduction of stretch values.

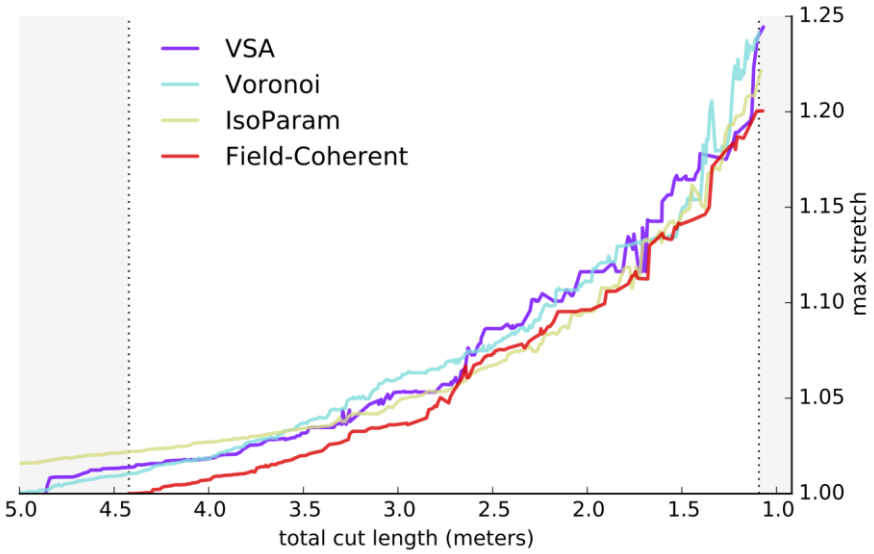


Figure 15: *Maximum stretch reached (y axis) with respect to the total cut length (x axis) for the cut optimization process, using different initial cut layouts.*

From our experiments, we found that, while any kind of reasonable initial partition works, quad-based, curvature-aligned coarse layouts [Pietroni et al. 2016] offer the best cut-length/deformation ratio most of the time. In the experiment, quad-based patch layout reduces the average deformation of VSA partitioning [Cohen-Steiner et al. 2004] by 5.1%, the Voronoi partitioning by 6%, and the Isoparametrization [Pietroni et al. 2010] by 3.4%. Finally curvature-aligned cuts tend to remain straight and more regular, which makes them easier to seal.

In order to speed up the optimization process, we perform a simple initial patch simplification step. We merge the patches where the average normal does not differ more than a given threshold ($\pi/16$), using a VSA-style approach (see Figure 4).

Fabricated examples We fabricated several models that are widely used in the Computer Graphics community to test our method. We used laser sintering to 3D print the flexible molds with thermoplastic polyurethane (TPU), a commercially available, flexible, and resistant material. The printing process we used requires a tolerance of 0.3 mm as a gap for the cuts; narrower gaps get fused. Our V-shaped cuts merged at the tip, but we were able to easily open them manually.

We cast replicas of several models with variable complexity (see Figures 12 and 16). All the molds are composed of a single connected component. As shown by the armadillo and dragon models, we successfully capture the high frequencies of the surface detail and we can extract the object without damaging the surface details or the mold. We also demonstrated our technique with topologically challenging examples, like the fertility model, that required several cuts. However, due to the flexibility of the mold shell, we were able to fabricate the replicas with a single-piece mold. In comparison, methods based on rigid molds would require either a highly complex multi-piece mold or alternatively several cast operations to create pieces of the object that would then be assembled in a post-processing step. Instead, we are able to achieve reproduction in one single cast operation with a one-piece mold. All the models were produced by a novice user who had never performed any castings, demonstrating the easy applicability of our approach. Tiny air bubbles trapped in the resin could have been removed by performing vacuum degassing before casting, a step that we ignored to keep the process as simple and lightweight as possible.

Figure 16: *Molds and casts obtained for the bimba, the bunny, the fertility, the armadillo and the dragon models.*

3 Reproducing the Female Figurine

The core of our system is an automatic method for generating feasible cut layouts for molds, building on a physics-based approach for simulating and evaluating the parting of the mold from the cast object. Our tests have shown that our method can handle geometrically difficult cases, such as models with high genus and intricate surface details. The flexible molds can be easily fabricated using 3D printing technology and have proven to be robust enough for multiple uses. Our system provides a convenient way of bringing virtual models to the real world as it empowers common users to cast objects with a wide range of materials and provides an attractive option, filling a gap between mass production and direct 3D printing.

This version of the mold generation system, described in the next sections, is internally distributed together with the releases of the whole authoring system.

In the following sections we describe in detail how flexible molds can be produced with our technique, explaining all the necessary steps and showing the tools involved using the Female Figurine artifact as an example. These instructions will serve as a guide for creating and using FlexMolds for any generic object.

Figure 17: *Two views of the Female Figurine model reconstructed using structure from motion technology.*

3.1 Model Processing

We provide four command line tools (***PatchGen, Unfolder, MoldGen, SupportGen*****)** that constitute the pipeline to produce a FlexMold starting from an input 3D model. Their role is described in the next sections (3.2, 3.3 and 3.4). The instructions on how to print and use a FlexMold for casting are detailed in sections 3.5, 3.6 and 3.7.

The first step to consider for the production of a FlexMold is the input model that we want to replicate. Any generic digital 3D model is suitable for this technique: from synthetic computer generated models to real objects digitized, independent from the technology used to acquire their geometry. Although this is in general true, for extremely complex shapes it may not be always possible to produce a mold that is usable from a practical point of view. In general, shapes with high genus and/or thin features cannot be easily reproduced even with traditional molding techniques and that is also the case of FlexMolds.

Apart from these considerations, the input 3D model must fulfil a set of additional requirements.

The input model must be a triangular mesh expressed in metric units. In particular the mesh should be scaled so that 1 unit = 1 meter. Also, considering the specific application, the size of the input model should be constrained: as a rule of thumb, each dimension of the mesh should be in the range of 5 cm to 18 cm. This is required for practical reasons because both big and small models are difficult to handle while performing the casting. Concerning big models, producing the relative FlexMold and the amount of casting material necessary have a cost proportional to the volume of the cast object. Therefore, increasing the size of the object has a considerable impact on the *cost-effectiveness* of the replica production. Moreover, in the FlexMolds technique (Section 2) the pressure of the liquid casting material during the production is not considered: we observed that for relatively small models the pressure induced on the inner surface of the mold can be neglected, but for models outside the advised size range the increased pressure can lead to significant deformation of the FlexMold and the produced cast.

To guarantee a correct processing, the input mesh must satisfy a set of geometric constraints. In particular it must be ensured that the mesh is:

- 2-manifold;
- orientable;
- watertight;
- free of self-intersections.

These properties are required to clearly define the volume enclosed by the mesh surface.

It is preferred to provide a 3D model that has a limited amount of triangles (i.e., less than 300K faces). This will increase the speed of some of the steps involved in the pipeline. Keep in mind that 3D printing technology employed for producing FlexMolds has a limited accuracy. In particular, consider using a mesh simplification strategy to reduce the triangle count trying to obtain a good compromise between number of triangles and reproduced details.

Moreover, it is recommended to avoid providing 3D models with bad triangulations, presenting folded and/or thin triangles, as this may prevent the correct execution of the tools involved.

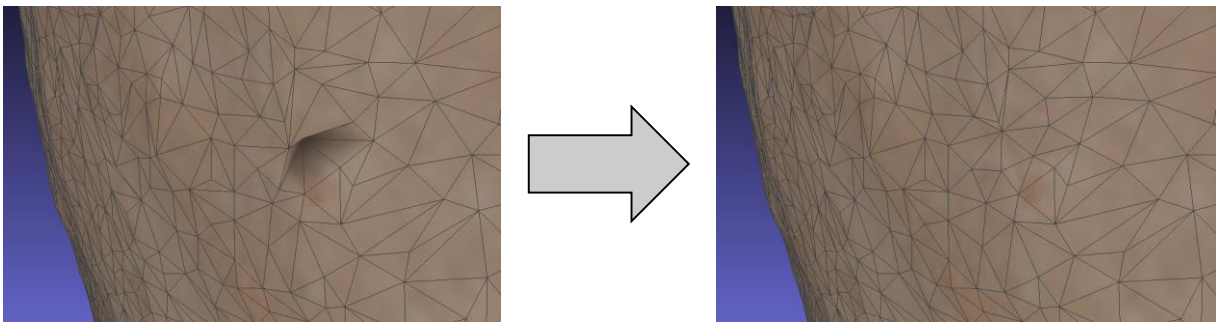


Figure 18: *An example of bad triangulation (left), and the triangulation after fixing (right).*

A mesh processing software like MeshLab [Cignoni et al. 2008] can be used to ensure these properties and to fix the model. Note that while processing a mesh there is no need to preserve color or other information stored with the model as the tools provided for the FlexMolds pipeline require the geometry only.

In the case of the Female Figurine, the input 3D model was created using structure from motion reconstruction from photos using Agisoft PhotoScan software (Figure 17) and consisted of 200K triangles. The size of the reconstructed model was in the acceptable range, specifically 10.5 x 16.4 x 8.2 cm. Furthermore, the model complied with all required geometric properties except from some bad triangulation issues that were manually fixed using MeshLab (Figure 18).

3.2 Patch Generation

Tool binary: **PatchGen**
Input: *original*
Output: *patches*

The output of the previous step represents the input to our method and, ultimately, the final shape of the model replicas that we are going to produce with the resulting flexible mold. As mentioned in the Section 2.3.3 our technique requires an initial patch decomposition of the model whose boundaries represent the set of candidate cuts for the mold. Launching the PatchGen tool on the original input we obtain a patch decomposition that can be then fed to our core algorithm to retrieve an optimized cut layout that will be used for producing the FlexMold.

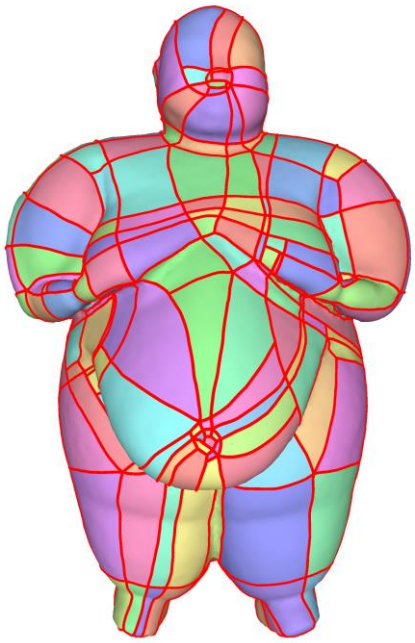


Figure 19: *A quadrilateral patch decomposition of the Female Figurine model.*

In our tool we used a simplified implementation of the technique described in [Pietroni et al. 2016] to produce an initial decomposition of the model surface into quadrilateral patches. In the case of the Female Figurine artifact we obtained a model made of 300 quadrilateral patches (Figure 19).

3.3 Cut and Mold Generation

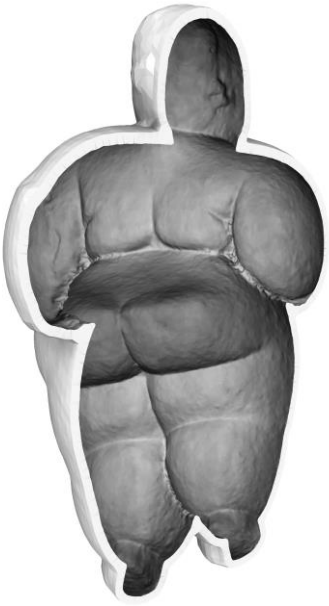
This is the most important part of the process and it is composed of two main steps:

- first, we have to find the opening cuts that will allow the release of the mold;
- then, using these cuts, we can define the actual 3D shape of the mold that will be fabricated.

These two steps correspond to two different binaries that have to be used in sequence: *Unfolder* and

MoldGen described in the next paragraphs.

Tool binary: **Unfolder**
Input: *patches*
Output: *cut_layout*



Starting from the quadrilateral patch of previous step, the core algorithm of our technique performs a simulation-driven optimization to find a cut configuration suitable for creating a FlexMold (Figure 20, left). The generation of a cut layout is a computationally expensive process and could take from minutes to a couple of hours, depending on the provided input. Please note that in the case of complex shapes, for which molding techniques are not effective, the algorithm will fail producing a cut layout that would be overly complicated and, from a practical point of view, not usable anyway.

Tool binary: **MoldGen**
Input: *cut_layout*, *original*
Output: *flexmold*

Starting from the obtained cut layout and the original model provided, this tool produces a solid shell model equipped with a *V-shaped* profile carved along the designated cut. A set of pegs protruding on the external shell of the mold is created to ease the sealing process (Figure 20, center). Also, our tool automatically equips the mold with a hole to allow the pouring of the liquid casting material and a set of smaller holes to allow the air to exit during the casting process.

For the specific Female Figurine FlexMold, only the pouring hole was generated because the air vents were not necessary for this specific shape (Figure 20, right).

The MoldGen tool, when it finishes processing, produces the FlexMold model but also outputs on the console the volume of the object wrapped by this mold. Take note of the volume as it will be necessary to measure out the casting material for the casting process (Section 3.7).

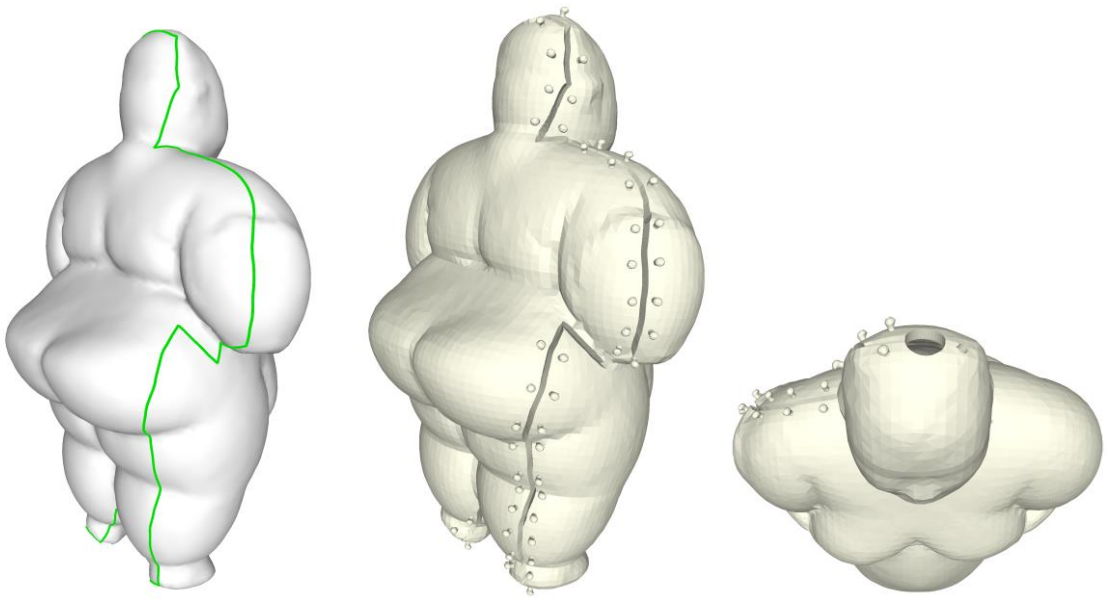


Figure 20: *The optimized cut layout (green) generated from the patch decomposition of the Female Figurine model (left); the FlexMold resulting from the provided cut layout (center); a particular of the pouring hole generated on the mold (right).*

3.4 Support Generation

Tool binary: **SupportGen**
Input: *flexmold*
Output: *support*

The resulting mold is produced with a preferential orientation that is automatically computed to enable the casting with the provided holes. To preserve its designated orientation during the casting process we provide a tool that, given the obtained FlexMold 3D model, generates a custom supporting structure model that can be 3D-printed using any 3D printing technology (Figure 21).

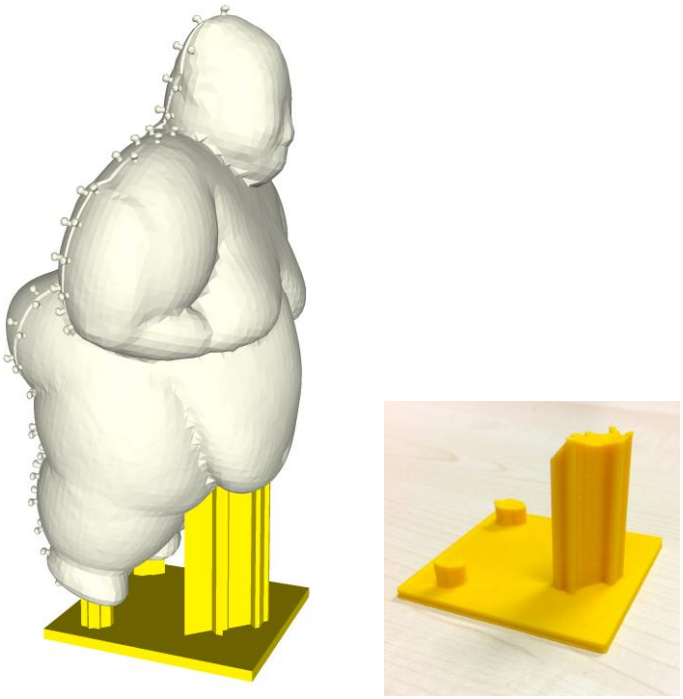


Figure 21: *The Female Figurine FlexMold on its supporting structure (left); the support 3D-printed with PLA material (right).*

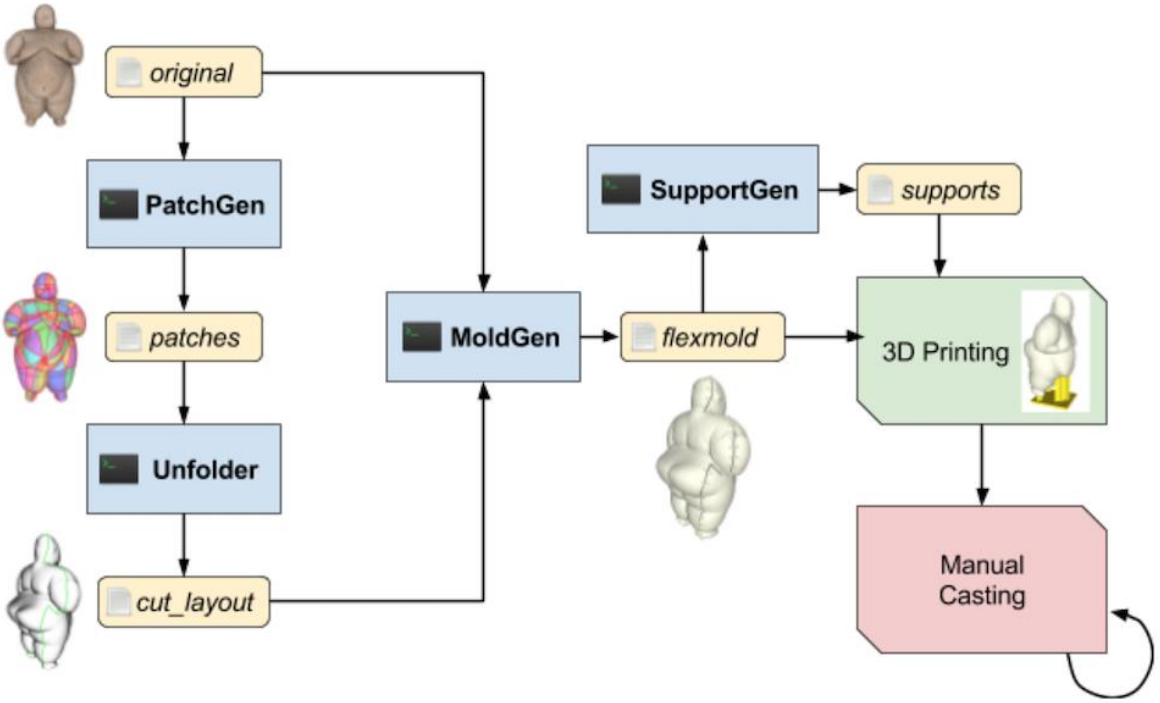


Figure 22: *Overview scheme of the mold production process and the tools involved.*

3.5 Printing mold and support

Once the FlexMold and the supporting structures model are generated they can be physically manufactured using 3D printing.

The FlexMold model is best created using Laser Sintering technology using a semi-rigid thermoplastic material, like TPU 92A. For cost-effectiveness, we advise to take advantage of online 3D printing services like Materialise OnSite[1], which offers to print objects with the suggested flexible material. Please note that either online services or 3D printing software can raise some warnings for the FlexMold printing, especially within the regions presenting the cuts. This is totally normal as no 3D printing technology is able to accurately reproduce some of the thin geometries present on mold. The warnings can be safely ignored as the FlexMold will be cut open exactly along the cut lines.

The support structure, as explained in the previous Section, can be produced with any 3D printing technology. Since it will serve as a support for the mold and its cast during the casting process, it must be able to withstand their weight. For this reason, the use of a very rigid and resilient material is advisable. For this purpose, standard ABS or PLA commonly used with FDM printers are good candidates.

The Female Figurine FlexMold was printed using TPU. The relative supporting structure was printed with an Ultimaker 2+ 3D printer using PLA material.

3.6 Casting setup

Notes: before starting preparing the mold for casting it is recommended to carefully read the Safety Data Sheets of each product that will be used in the process.

Once the 3D printed mold is obtained it must be prepared for casting. The first step consists in applying a coating of release agent on the inner surface of the molds. This procedure is mandatory if casting using urethane resins or other materials that may stick to the inner surface of the mold.

The next step consists of closing the cuts on the mold taking advantage of the protrusion present on the mold (on both sides of the cuts). If casting with gypsum, cuts can be closed with little rubber bands. If you

[1] https://onsite.materialise.com/

employ urethane resins it is recommended to use nylon or cotton threads. This is necessary since resins produce heat during the curing process and that may break thin rubber bands. For increased robustness it is advisable to close the cuts with a “cross stitch”-like pattern as shown in Figure 12a.

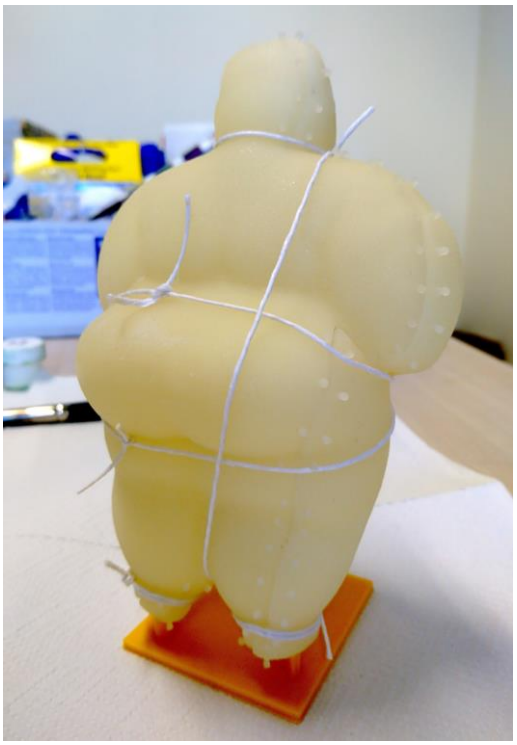


Figure 23: *The FlexMold closed by tying it with cotton threads on the 3D printed support.*

Once the mold is closed we must guarantee that every cut is perfectly sealed. Although many solutions can be used to perform this step, from direct experience we suggest to apply a layer of viscous silicone to seal the mold watertight along the cuts. The recommended material is *Smooth-On Dragon Skin® 10 VERY FAST*[2]. Please note that this step must be performed very carefully to prevent leaking of the casting material.

3.7 Casting

After the FlexMold is sealed it can be put on its 3D-printed supporting structure as shown in Figure 13. At this point it is necessary to fix thin straws on the air vents to allow the air to exit during the casting. Keep in mind that these straws must be taller than the whole model: in particular, their summit must be higher than the pouring hole as the casting material will flow inside them and will reach the same height everywhere. Additionally, to facilitate the pouring of the casting material, it is recommended to fix a funnel onto the pouring hole.

Attaching the funnel and the straws to the mold can be performed by using bicomponent silicone rubber, which can be mixed by hand and cures in a few minutes. As an alternative, simple play dough can be used as well.

At this point the setup is completed and the model is ready to accommodate a cast. Prepare your casting material following the instruction specific for each application. To obtain an accurate casting we recommend the use of urethane resins, like the Smooth-On Smooth-Cast 320[3]. An adequate amount of casting material must be prepared using the volume information obtained from the MoldGen tool (Section

[2] https://www.smooth-on.com/products/dragon-skin-10-very-fast/
[3] https://www.smooth-on.com/products/smooth-cast-320/

3.3) and, at this point, can be poured into the mold.

Once the mold is filled everywhere it must be tilt around to allow air bubbles eventually blocked inside the cast to exit through the air vents or the pouring hole.

Once the casting material is fully cured, the mold can be cut open again by removing the sealing material and the cast replica of the object can be extracted from the FlexMold which can then be reused to produce multiple replicas.

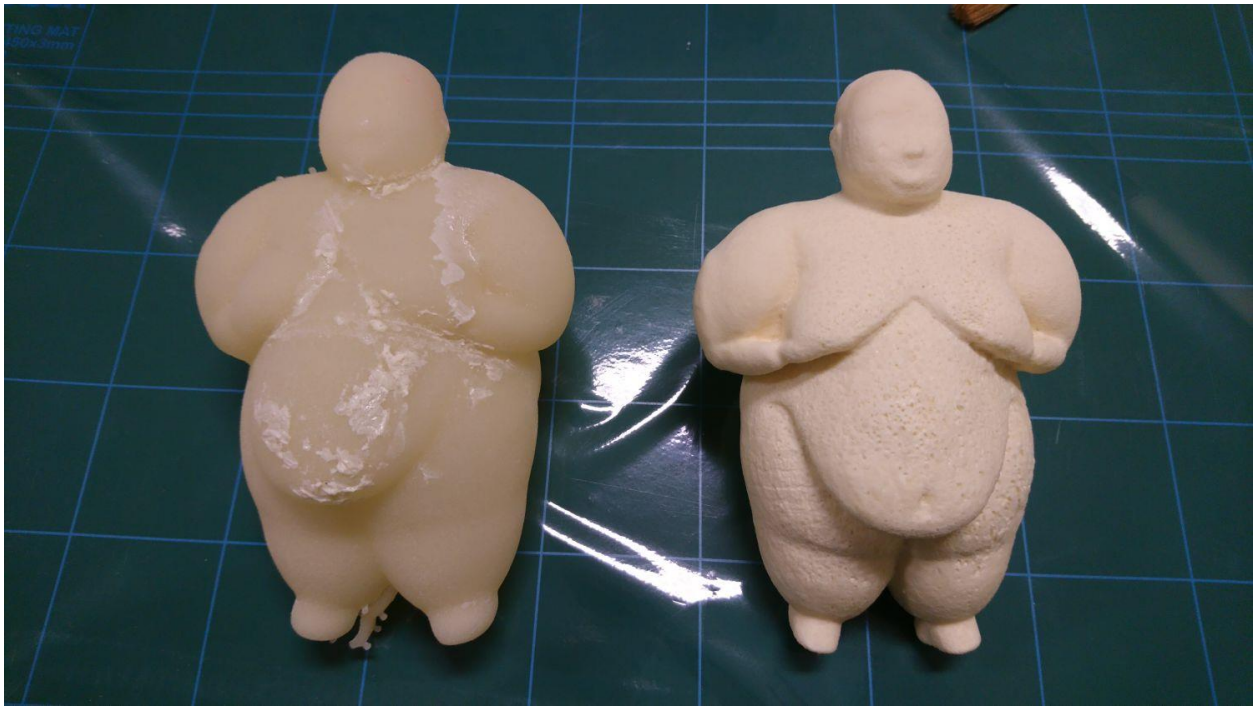


Figure 24: *The FlexMold after the casting and the resulting Female Figurine resin copy just after its extraction from the mold.*

4 Bibliography

Bender, J., Müller, M., Otaduy, M.A., Teschner, M., and Macklin, M. 2014. A Survey on Position-Based Simulation Methods in Computer Graphics. *Computer graphics forum: journal of the European Association for Computer Graphics 33*, 6, 228–251.

Bommes, D., Zimmer, H., and Kobbelt, L. 2009. Mixed-integer quadrangulation. *ACM transactions on graphics 28*, 3, 1.

Bouaziz, S., Martin, S., Liu, T., Kavan, L., and Pauly, M. 2014. Projective dynamics. *ACM transactions on graphics 33*, 4, 1–11.

Campen, M., Bommes, D., and Kobbelt, L. 2012. Dual loops meshing. *ACM transactions on graphics 31*, 4, 1–11.

Chakraborty, P. and Venkata Reddy, N. 2009. Automatic determination of parting directions, parting lines and surfaces for two-piece permanent molds. *Journal of Materials Processing Technology 209*, 5, 2464–2476.

Chen, D., Sitthi-amorn, P., Lan, J.T., and Matusik, W. 2013. Computing and Fabricating Multiplanar Models. *Computer graphics forum: journal of the European Association for Computer Graphics 32*, 2pt3, 305–315.

Chen, X., Zhang, H., Lin, J., et al. 2015. Dapper. *ACM transactions on graphics 34*, 6, 1–12.

Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G. 2008. MeshLab: an Open-Source Mesh Processing Tool. *Eurographics Italian Chapter Conference*, The Eurographics Association.

Cignoni, P., Pietroni, N., Malomo, L., and Scopigno, R. 2014. Field-aligned mesh joinery. *ACM transactions on graphics 33*, 1, 1–12.

Cohen-Steiner, D., Alliez, P., and Desbrun, M. 2004. Variational shape approximation. *ACM SIGGRAPH 2004 Papers on - SIGGRAPH '04*.

Dey, T.K. 1994. A new technique to compute polygonal schema for 2-manifolds with application to null-homotopy detection. *Proceedings of the tenth annual symposium on Computational geometry - SCG '94*.

Grinspun, E., Hirani, A.N., Desbrun, M., and Schröder, P. 2003. Discrete shells. *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, 62–67.

Gu, X., Gortler, S.J., and Hoppe, H. 2002. Geometry images. *Proceedings of the 29th annual conference on Computer graphics and interactive techniques - SIGGRAPH '02*.

Herholz, P., Matusik, W., and Alexa, M. 2015. Approximating Free-form Geometry with Height Fields for Manufacturing. *Computer graphics forum: journal of the European Association for Computer Graphics 34*, 2, 239–251.

Hildebrand, K., Bickel, B., and Alexa, M. 2012. crdbrd: Shape Fabrication by Sliding Planar Slices. *Computer graphics forum: journal of the European Association for Computer Graphics 31*, 2pt3, 583–592.

Hu, R., Li, H., Zhang, H., and Cohen-Or, D. 2014. Approximate pyramidal shape decomposition. *ACM*

transactions on graphics 33, 6, 1–12.

Hwang, Y.K., Ahuja, N., United States. Army Research Office, University of Illinois, Center, A.C.T., and Laboratory, N.C.E. 1992. *Gross Motion Planning: A Survey*. .

Jiménez, P. 2012. Survey on model-based manipulation planning of deformable objects. *Robotics and computer-integrated manufacturing 28*, 2, 154–163.

Julius, D., Kraevoy, V., and Sheffer, A. 2005. D-Charts: Quasi-Developable Mesh Segmentation. *Computer graphics forum: journal of the European Association for Computer Graphics 24*, 3, 581–590.

Keinert, B., Innmann, M., Sänger, M., and Stamminger, M. 2015. Spherical fibonacci mapping. *ACM transactions on graphics 34*, 6, 1–7.

Lévy, B. 2014. Restricted Voronoi Diagrams for (Re)-Meshing Surfaces and Volumes. In: *Curves and Surfaces*. .

Lin, A.C. and Quang, N.H. 2014. Automatic generation of mold-piece regions and parting curves for complex CAD models in multi-piece mold design. *Computer-aided design and applications 57*, 15–28.

Liu, L., Shamir, A., Wang, C., and Whitening, E. 2014. 3D printing oriented design. *SIGGRAPH Asia 2014 Courses on - SA '14*.

Luo, L., Baran, I., Rusinkiewicz, S., and Matusik, W. 2012. Chopper. *ACM transactions on graphics 31*, 6, 1.

Malomo, L., Pietroni, N., Bickel, B., and Cignoni, P. 2016. FlexMolds: automatic design of flexible shells for molding. *ACM transactions on graphics 35*, 6, 1–14.

Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. 2007. Position based dynamics. *Journal of visual communication and image representation 18*, 2, 109–118.

Müller, M., Stam, J., James, D., and Thürey, N. 2008. Real time physics. *ACM SIGGRAPH 2008 classes on - SIGGRAPH '08*.

Myles, A., Pietroni, N., and Zorin, D. 2014. Robust field-aligned global parametrization. *ACM transactions on graphics 33*, 4, 1–14.

Pietroni, N., Puppo, E., Marcias, G., Scopigno, R., and Cignoni, P. 2016. Tracing Field-Coherent Quad Layouts. *Computer graphics forum: journal of the European Association for Computer Graphics 35*, 7, 485–496.

Pietroni, N., Tarini, M., and Cignoni, P. 2010. Almost isometric mesh parameterization through abstract domains. *IEEE transactions on visualization and computer graphics 16*, 4, 621–635.

Ray, N., Vallet, B., Alonso, L., and Levy, B. 2009. Geometry-aware direction field processing. *ACM transactions on graphics 29*, 1, 1–11.

Razafindrazaka, F.H., Reitebuch, U., and Polthier, K. 2015. Perfect Matching Quad Layouts for Manifold Meshes. *Computer graphics forum: journal of the European Association for Computer Graphics 34*, 5, 219–228.

Saha, M. and Isto, P. Motion planning for robotic manipulation of deformable linear objects. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*

Schüller, C., Panozzo, D., Grundhöfer, A., Zimmer, H., Sorkine, E., and Sorkine-Hornung, O. 2016. Computational thermoforming. *ACM transactions on graphics 35*, 4, 1–9.

Schwartzburg, Y. and Pauly, M. 2013. Fabrication-aware Design with Intersecting Planar Pieces. *Computer graphics forum: journal of the European Association for Computer Graphics 32*, 2pt3, 317–326.

Shamir, A. 2008. A survey on Mesh Segmentation Techniques. *Computer graphics forum: journal of the European Association for Computer Graphics 27*, 6, 1539–1556.

Sifakis, E. and Barbic, J. 2012. FEM simulation of 3D deformable solids. *ACM SIGGRAPH 2012 Posters on - SIGGRAPH '12*.

Singh, R. 2009. Three Dimensional Printing for Casting Applications: A State of Art Review and Future Perspectives. *Advanced materials research 83-86*, 342–349.

Skouras, M., Thomaszewski, B., Bickel, B., and Gross, M. 2012. Computational Design of Rubber Balloons. *Computer graphics forum: journal of the European Association for Computer Graphics 31*, 2pt4, 835–844.

Solomon, J., Vouga, E., Wardetzky, M., and Grinspun, E. 2012. Flexible Developable Surfaces. *Computer graphics forum: journal of the European Association for Computer Graphics 31*, 5, 1567–1576.

Tang, C., Bo, P., Wallner, J., and Pottmann, H. 2016. Interactive Design of Developable Surfaces. *ACM transactions on graphics 35*, 2, 1–12.

Tarini, M., Puppo, E., Panozzo, D., Pietroni, N., and Cignoni, P. 2011. Simple quad domains for field aligned mesh parametrization. *ACM transactions on graphics 30*, 6, 1.

Temple Black, J., De Garmo, E.P., and Kohser, R.A. 2013. *DeGarmo's Materials and Processes in Manufacturing*. John Wiley & Sons Incorporated.

Umetani, N., Bickel, B., and Matusik, W. 2015. Computational tools for 3D printing. *ACM SIGGRAPH 2015 Courses on - SIGGRAPH '15*.

Vanek, J., Garcia Galicia, J.A., Benes, B., et al. 2014. PackMerger: A 3D Print Volume Optimizer. *Computer graphics forum: journal of the European Association for Computer Graphics 33*, 6, 322–332.

Wannarumon, S. 2011. Reviews of Computer-Aided Technologies for Jewelry Design and Casting. *Naresuan University Engineering Journal 6*, 1, 45–56.

Yao, M., Chen, Z., Luo, L., Wang, R., and Wang, H. 2015. Level-set-based partitioning and packing optimization of a printable model. *ACM transactions on graphics 34*, 6, 1–11.

Zhang, C., Zhou, X., and Li, C. 2009. Feature extraction from freeform molded parts for moldability analysis. *International Journal of Advanced Manufacturing Technology 48*, 1-4, 273–282.

D6.2: QUALITY CHECKLIST

Julien Philip
INRIA

Document Review Date: 31/10/20017
Document version reviewed: 0.1

CONFIDENTIAL DOCUMENT

Criteria		Verified (Y/N)
1) Conformity to Standards and Project templates		
	Use of EMOTIVE Deliverable template	Y
	Cover page information completed (Number, title, authors, organizations, dates, version number, dissemination level, abstract)	N
	Table of contents updated	Y
	List of Abbreviations updated	Y
	Executive summary completed	Y
	Deliverable file title properly structured (EMOTIVE_ID-DocumentName.Version.ext, e.g. EMOTIVE_*D3.1-User_Requirements_Scenarios–A.V0.1.docx*)	N
	Template fonts and styles followed	Y
	Deliverable title in Footer completed	Y
Comments		
2) Language review (typing mistakes, grammar, etc.)		
	Revised document with language corrections sent to Del. Leader?	Y
Comments		
3) Coherence with document / task objectives as declared in the DoA		Y
Comments		
4) Reliability of data		
	Information and sources well identified	Y
	Bibliography section properly structured (if applicable)	Y
Comments		

5) Validity of content In your opinion,	Y
are there any sections missing?	Y
does the document cover the topic successfully?	Y
is information presented in a structured and clear way?	Y
are conclusions presented sufficiently?	Y

Comments / Suggestions for revision	

6) Deliverable Accepted? (provided that suggested changes are implemented)	**Y**

If no, please state reasons:	

The following section should be filled in by the Deliverable Leader	
7) Implementation of revisions/modifications suggested	
Explanation for eventual rejections	

Please send the filled checklist to the Deliverable Leader and to the Project Coordinator.

D6.2: QUALITY CHECKLIST

Alejandra Barragán, Rémi Courtel
Diginext

Document Review Date: 06/11/2017
Document version reviewed: "Deliverable 6.2.TBR_INRIA.docx"

CONFIDENTIAL DOCUMENT

Criteria		Verified (Y/N)
1) Conformity to Standards and Project templates		
	Use of EMOTIVE Deliverable template	Y
	Cover page information completed (Number, title, authors, organizations, dates, version number, dissemination level, abstract)	Y
	Table of contents updated	Y
	List of Abbreviations updated	Y
	Executive summary completed	Y
	Deliverable file title properly structured (EMOTIVE_ID-DocumentName.Version.ext, e.g. EMOTIVE_*D3.1-User_Requirements_Scenarios–A.V0.1.docx*)	N
	Template fonts and styles followed	Y
	Deliverable title in Footer completed	Y
Comments		
2) Language review (typing mistakes, grammar, etc.)		
	Revised document with language corrections sent to Del. Leader?	Y
Comments		
3) Coherence with document / task objectives as declared in the DoA		Y
Comments		
4) Reliability of data		
	Information and sources well identified	Y
	Bibliography section properly structured (if applicable)	Y
Comments		

5) Validity of content In your opinion,	
are there any sections missing?	N
does the document cover the topic successfully?	Y
is information presented in a structured and clear way?	Y
are conclusions presented sufficiently?	Y

Comments / Suggestions for revision	

6) Deliverable Accepted? (provided that suggested changes are implemented)	**Y**

If no, please state reasons:	

The following section should be filled in by the Deliverable Leader	
7) Implementation of revisions/modifications suggested	
Explanation for eventual rejections	

Please send the filled checklist to the Deliverable Leader and to the Project Coordinator.