Abstract Surface Modeling for concurrent Form Finding and Class A Surfacing in Computer-Aided Design

Sebastian Misztal
University of Applied Sciences and Arts
Ricklinger Stadtweg 120
30459, Hanover, Germany
sebastian.misztal@hs-hannover.de

Ingo Ginkel
University of Applied Sciences and Arts
Ricklinger Stadtweg 120
30459, Hanover, Germany
ingo.ginkel@hs-hannover.de

ABSTRACT

As the missing link between designers and engineers, we introduce a new abstract modeling approach for computer-aided design systems. In contrast to existing solutions, our strategy is less geometry driven and less based on low-level aspects like control points or mesh elements. Instead we operate on the idea of a hierarchical modular concept with abstract components like categories, parts and the features relations. The whole surface structure of the model is composed of abstract areas represented by meshes. This allows designers without engineering background to concentrate intuitively on the form finding process as they easily model abstract components and automatically generate high quality CAD-freeform equivalents suitable for computer-aided manufacturing. During the phase of construction, we focus on the designers intent and guide him through this process to enrich the model with semantic information. The goal is to describe the models structure, such that the automatically generated freeform surfaces not only meet correct geometry but also mirror the internal configuration of the model. Compared to common practice where design changes and updates lead to the reconstruction of the whole model, our system accomplishes these kinds of alterations automatically, based on the hierarchical model configuration, derived from the designers intent. So our approach of an abstract modeler is much faster, closes the gap between creative design changes and technical model construction and captures this in one contemporary system and workflow.

Keywords

Computer-Aided Design (CAD), Computer-Aided Manufacturing (CAM), Geometric Modeling, Interaction Techniques, 3D Modeling, Class A Surfacing, Shape Design, Object Representations

1 INTRODUCTION

Designing and manufacturing a product incorporates various tasks to be performed by professionals with partly very divergent backgrounds. These can be roughly classified into designers with creative minds on the one hand and engineers with technical expertise on the other hand.

Designers, who are responsible for form finding, often refuse technologies like splines and NURBS. So they usually rely on 2D-sketches, real clay models or virtual mesh models of the object they are creating.

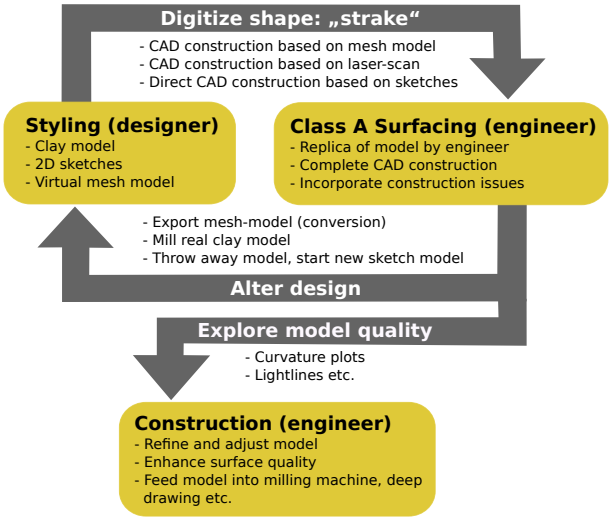

Figure 1: *Common design / CAD workflow*

Engineers have to transfer the model into a manufacturable model in a further step. Based on a real clay model or sketches, the whole technical CAD-model has to be created. In the case of a virtual mesh model one has to state that meshes can not provide a suitable mathematical surface quality, thus a CAD-replica of the shape is also necessary.

So for all of the usual design approaches, a hand-crafted recreation of a whole model into a freeform surface representation is everyday business and of course very time-consuming. Even more serious is the problem when a designer wants to alter the shape of a model afterwards and the engineer has to rebuild the whole freeform model again. Reverse eingineering in different process steps became mandatory [SK05]. The media disruption between a model created by a designer and the one formed by an engineer and its impact to the overall workflow is depicted in figure 1.

The task to overcome the media disruption is either to teach an engineer a creative mind or to provide designers a suitable modeling tool for designing and producing a manufacturable model simultaneously. The goal of our approach is the latter one, providing the designer an intuitively usable tool which guides him through the modeling process and tries to capture the design intent. Since neither meshes nor spline representations are suitable for a designer to simultaneously shape the desired object and create a manufacturable model, a combined or hybrid approach is necessary.

The idea is to build a system which allows modeling on a less detailed but hierarchical abstraction level. The model will both cover a mesh model and a spline representation but neither of them will be modified directly in the sense of low level polygonal transformations, control point movement, degree selection or parametrization changes. This kind of construction is too much a low level approach and distracts the designer from his main task, namely to find the overall shape of the object.

Instead we provide the user with an abstract surface model consisting of abstract surface areas, hierarchical dependencies and enrich this model with semantic information which will be queried from the user or can implicitly be derived from the context (i.e. the used construction tool, moment and region where and when it is applied). Topological operations will not be performed on meshes or spline surfaces, but on an abstract level containing adjacency between abstract areas freed from low level aspects like mesh consistency or curvature continuous transitions.

Based on this abstract model, a mesh surface for preview purposes will automatically be generated delivering a fast feedback. In contrast to a usual mesh model, additional sematic information, design intents etc. are still stored in the abstract model. After some iterations

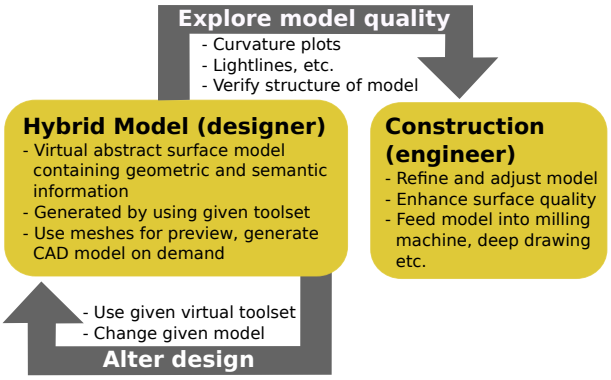

Figure 2: *Intended design / CAD workflow*

of design changes and form finding, this information can be used to automatically create a class A surface model which can be used as input for the construction phase.

Eliminating the media disruption between the design- and the manufacturing-model (i.e. designing and constructing in a joint hybrid model) the overall workflow can be simplified substantially, as shown in figure 2.

2 MODELING CONCEPTS AND TECHNOLOGIES

Modeling objects using virtual construction tools basically allows three conceptually different approaches. The first one are mesh-modelers, providing the generation of simple geometric shapes. Another solution are CAD-construction tools using freeform surfaces for class A surfacing. The third concept are solid modelers, focussing on solid primitives, building constructive elements. In any of these configurations a designer and an engineer still produce the before mentioned media disruption.

Most CAD/CAM tools focussing on the media disruption between design and construction phase try to simplify and automate the conversion between the generally used representations. This also holds for 3D sketching approaches like in [ZHH96] or [Diehl04]. These systems are trying to guide the designer through a model finding process, but still require a conversion of the design- (i.e. the sketches) into a construction-model. Using a converter is of course much faster than hand-crafting the model again for construction, but converters often demand adaptions after every conversion. Additionally with every conversion, valuable information get lost. Converting from a design-representation to a construction model, design intents and specific shapes can be sacrificed for technical restrictions. Converting a construction model into a design model often causes a loss of structural information.

To recover these informations various startegies are pursued, i.a. shown in [HPR00]. Reverse engineering approaches traverse the history-tree or -graph and thus all

recorded modeling operations and interpret their constellation to extract valuable meta information from it. But this requires an expressive tree- or graph-structure, holding the desired intent. Hint-based solutions analyse geometrical characteristics which also only qualify for capturing geometrical features. Knowing the pure features without any context is insufficient knowledge to create an overall view of the designers purposes. Moreover, relations are often not cosidered.

Concepts like [LLM10] decompose the whole modeling arrangement into smaller sub-parts to find symmetries which makes it easier to detect features and the design intent. Conceivable is also to split the history tree into little pieces too. The limitations of history-trees is the fact that they map a chronological sequence. In contrast, our idea is independent from the order of the operations by just using the structural and hierarchical information.

Establishing a modeling methodology like [BRC14] with advanced rules, predetermining the sequence of types of operations during the modeling process, lead to a clearer history tree and a better mapping of relations between elements and thus to better conditions for mentioned intent recognition approaches. But this methodology still requires a strong contribution of the designer and is less technology driven like the strategy of [AQRPHC12]. This attempt demands the designer to make annotations, describing their design intent which is helpful in further modeling steps especially for other designers, working with an unfamiliar model. Unfortunately these annotations are interpreted by human beeings and not by a machine.

Machine-interpretable technologies to capture the design intent are given in [KMY06] by reasoning which is ontology-based. Spatial relationships can be stored in a relational model. But the focus here lies on assembly design which is difficult to adapt to classical product design issues.

During the evolution of CAD systems, different generations and paradigms emerged, dealing with the capturing of design intents in the broadest sense, i.a. presented in [TDM10]. Parametric technologies and feature- as well as history-based systems in solid modelers are the fundamental idea, capable to describe the model precisely enough to map geometrical relations and dependencies. One of the major drawbacks is the steep learning curve. Direct or also explicit modeling tools have the advantage of beeing very simple, easy to learn and fast. But the absence of a construction history and missing relations require reverse engineering to capture features and design intents.

So today vendors tend to incorporate both approaches to hybrid solutions in its literal sense. That means that these concepts are not completely combined. Instead they often operate synchronous where the designer can switch between them like in *Siemens NX and SolidEdge*. *PTC Creo Parametric* also allows the adding of constarints to the model during direct modeling. But our goal is to create high quality freeform surfaces straight from the explicit modeling object. Other professional software like *Autodesk Fusion 360* provide tools like *snapping* which allows the user to create the freeform surfaces directly on the mesh but still being a time-consuming low-level attempt. Declared converters like *Kontenpunkt PointMaster* produce correct freeform surfaces but neither capture the designers intent nor provide an internal model structure that is suitable for production without further handwork. Solid modelers like *SolidWorks* provide very nice internal structures of the model, containing defined relations between parts. They also provide precise shapes of primitives based on implicit representations which are previewed by meshes. Their weak spot is the restriction to fairly simple surfaces based on the solids. So creating a real freeform surface with a solid modeling tool is very challenging.

To construct our system, we borrow ideas from each of these approaches. We use meshes for preview, parametric surfaces for construction models and an internal structure of abstract regions that are aligned and interact very similar to a solid modeling concept. This tool will allow arbitrary surface shapes, explicitly modeled internal dependencies and hierarchies built by a designer with limited technical skill or interest. The goal of our approach is to guide and help the designer to intuitively find the form or shape of an object. Simultaneously we need to establish the preconditions that allow an automatic retrieval of construction data, namely a high quality CAD model covering both the shape and internal structure (part groups, hierarchy etc.).

3 ABSTRACT MODELER

Our abstract modeler is an extending modeling paradigm which is theoretically transferable to various modeling systems. We integrated our technical implementation into the CAD-system *Rhinoceros 5* as a plug-in. We used the existing surface representations and tools as low-level objects and operations respectively and created our own high-level objects and operations upon them. In this section we introduce the basic concepts of our paradigm and describe the structure of our solution by an exemplarily workflow.

3.1 Modeling Elements

Meshes are consistent structures where the features are characterized by an absence of mesh elements or by a specific geometry or constellation of vertices. That means that features in a common mesh are not just integrated into the whole construct, they are melded into the mesh. Freeform surfaces expose their features in a

similar manner. They are defined by the mathematical representation of the surfaces and finally a geometrical declaration. Features in both technologies basically do not differ from the rest of their object, as the border between a feature and the rest of the model is fluent. An automatized way to identify a feature would be very expensive as they have to be retrieved afterwards by using complex algorithms. And there is no guarantee that all features and the hard boundaries between a feature and the rest of the model will be found.

Nevertheless our system is based on the idea of a modular concept with separable elements with a clear distinction between them. The following properties state our requirements to these elements:

- Each feature or part of the model is an individual, distinct and nameable model-element.
- Each model-element can be addressed and selected.
- A model-element has a designated state, carrying meta information.
- The meta information of a model-element precisely describes its own geometry and its relation and connectivity to other model-elements.
- A model-element is a module, which can be exchanged and integrated into another model.
- The consistent surface of the model is formed through the connections of a set of model-elements.
- Operations can either be performed on an element individually or by a related set of elements using their meta information and geometry.

As mentioned before, common surface elements like meshes and freeform surfaces are not the fundamental objects in our attempt to describe a model. The elements we use are an abstract generalization not only storing geometrical information. The visual shape of a model is composed of the two model-elements *additive* and *area*.

Definition 1 (Area) *An* area *is a distinct, bounded area, lying directly on the geometrical surface of a model. Together connected, areas form the outer hull of a model.*

Areas are the basis element comparable to faces of meshes or freeform surfaces in typical CAD-models but with fundamental differences. They do not form the whole model. Instead they shape the coarse form of it and function as a carrier for other elements. In practical application, areas not only have the task to structure a model but also to organize other elements of the model. To describe the whole model, there is a further element called additive.

Definition 2 (Additive) *An* additive *is a combination of arbitrary geometrical elements. As a closed unit, additives are modules with the ability to be attached to areas. An additive itself can again be composed of a modeling structure with areas and additives.*

Additives represent a self-contained feature. In practical application, whenever a designer wants to create a part of the model which can be named and modeled on its own, detached from the rest, he would create a new additive. As areas only describe the coarse parts of the model, additives are used when fine sections have to be created.

Definition 3 (Layer) *Each area can be layered. A* layer *is a specific area with its attached additives in a system with under- and overlaying other areas with their additives. A layer has a level, synchronous to its time of creation.*

Basically a layer is not more than a specific decorated area instance which can be exchanged by other layers. In practical application, several layers are used when the designer wants to experiment with different shapes and additive constellations. As each layer has a level, according its chronological creation, the designer can travel the progress time by switching the level of the layer. The unique characteristic here is the fact that individual parts of the model can be layered and traversed. Thus it is possible to assemble and modify a model with parts of several progress steps.

Definition 4 (Base Object) *The* base object *is a compound of not overlapping areas, on not necessary equal leveled layers, forming a gapless (and solid) surface structure. Each not overlapping (and solid) constellation of areas forms a valid base object, representing one instance of the models surface.*

Definition 5 (Model) *A* model *is the entirety (set) of all model-elements (additives and areas), layers, relations and operations, allocated to this model.*

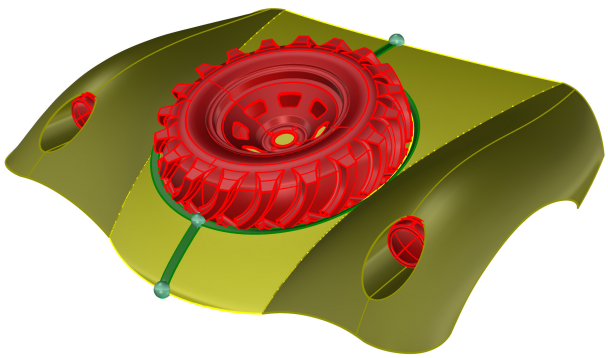

Figure 3: *Engine hood of a vehicle with attachments.*

Figure 3 shows a part of a vehicle with different model-elements. The surface of the model is constructed of areas (colored in yellow), all together forming the base object. Different shades of yellow indicate different layers. The spare tire and the headlights are designed as additives (colored in red and complete models themselves) attached to the engine-hood-area. The connectivity of the spare tire to the engine hood is highlighted in green. The position of the tire can be adjusted by using control points (also in green). Moving these control points alters the areas geometry as the hole for the tire is moved too. Control points at the border of the middle area adjust the size of this area which in turn has influence on the tire depending on the relation between them. Varieties of this impact are illustrated in figure 4.

3.2 Meta Information

As all model-elements are defined as separable objects, relations are the connecting element between them, creating the consistency of a model.

Definition 6 (Relation) *A* relation *is a meta information, describing the state between two model-elements with the following parameters:*

- *The connectivity, including the type (loose, fixed, fluent, static etc.) and adjacency.*
- *The dimension, including the sizing information.*
- *Distances, including absolute and relative lengths.*
- *Repetition, including the mirroring of elements using determined distances.*
- *The rank, including priorities of elements and relations.*

In practical application, relations come into play when the model or single parts of it are modified. These relations describe the behavior of linked elements and the modified object itself.

Definition 7 (Operation) *An* operation *is an action directly applied on one or more model-elements, areas and/or additives. Operations are all kinds of transformations on an element, not changing its relations. Relations of the considered and related elements are triggered through an operation. Operations aggregate the type of transformation, its location and the applied tool with parameters.*

Unlike relations, an operation is not a determined and fixed state. Operations are sequential and retraceable actions where the order matters. In practical application, operations are used to shape an element.

Definition 8 (Mode) *A* mode *is a state, chosen by the user, defining the impact of an operation on the model by redefining its relations.*

The idea is to attain different effects with the same operation just by switching the mode. Theoretically our systems allows an unlimited number of modes, some pre-defined by the system, further modes can be created by the user himself. We introduce two fundamental modes, also used in our example. The *geometry-mode* is a state where geometrical operations on a selected element have pure and isolated geometrical impact only on this particular element. The designer is performing plain low-level geometrical modifications on separated parts of the model without further impact on other parts. This is the kind of behavior which a designer would expect as he knows it from familiar modeling tools. Here the geometry-mode is mostly used for shaping the outer form of the model or isolated features. The *semantic-mode* is a state where geometrical operations on an element not only modifies its geometry but also all parts of the model which are semantically connected to it. As discussed before, adjacent elements of the model can be mutually related. These relations are used to determine the effect of an alteration.

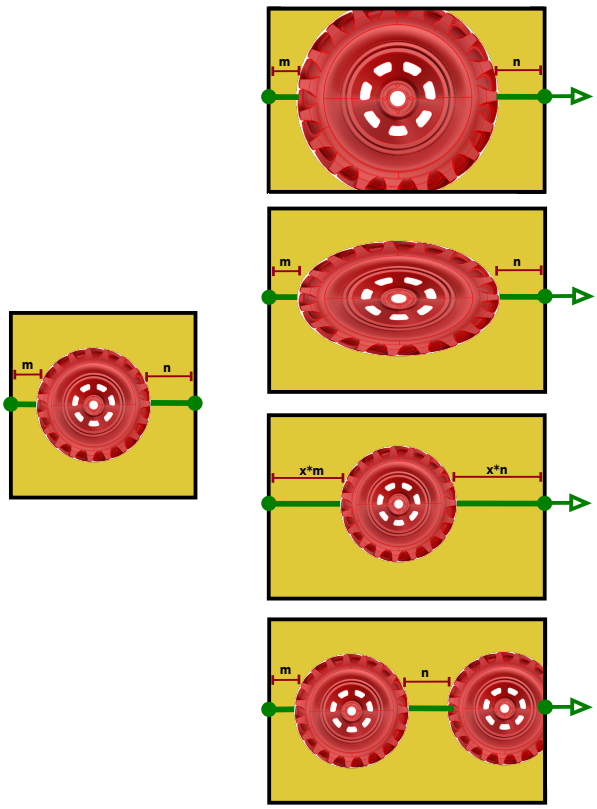

Figure 4: *Tire as an additive, attached to an area.* ***Left:*** *Initial state.* ***Right:*** *Various impacts on the tire after resizing the area depending on their relations.*

Figure 4 illustrates a tire on an area, similar to the example in figure 3. The tire is attached through relations between the border of the area and the border of the tire with defined distances m and n. When the area is enlarged (right column), the impact on the attached tire

depends on the predefined relations between them, particularly the parameters of the connections. The resizing of the area results in a resizing of the tire by satisfying the absolute distances *m* and *n* and its shape (*first image*), in a deformation of the tire by satisfying the absolute distances *m* and *n* (*second image*), in a translation of the tire by satisfying the relative distances *m* and *n* and its shape (*third image*) or in a mirroring of the tire by satisfying the absolute distances *m* and *n* and its shape (*fourth image*).

3.3 Mesh Display

Our surface representation is an abstract construct which needs to be displayed in some way. So to render our abstract surfaces we use polygonal meshes created by tessellation techniques from Delaunay [DeBerg08] and extended methods from Chew [Chew87] and Shewchuk [Shewchuk05] called Constrained Delaunay. This kind of tessellation creates triangles whose interior angles all tend to have the same size as they vary just a minimum user defined value from $\frac{\pi}{3}$. The result are homogenous looking meshes which is not only an advantage during rendering. The structure of the faces of these meshes also leave a more valuable impression for the designer.

Besides the pure display and tessellation, our system also uses subdivision techniques from Catmull-Clark [Catmull98] and Loop [Loop87] and furthermore extended methods from Ginkel [GU06]. Curved surfaces are created by subdividing the mesh representation of the areas to be shaped. Therefor the abstract area is decorated with an subdivision operation. These operations are editable and exchangeable. Other subdivision methods can be performed on the area by changing parameters.

Figure 5 exemplarily shows the creation of a transition between areas using Catmull-Clark subdivision techniques. The designer can expand this blended area through four control points at the corners and alter its roundness through another control point in the middel. The border is highlighted in green color. Ordinary meshes or freeform surfaces would provide a much larger number of vertices or control points respectively. We use just five. The techniqual details are hidden from the designer to reduce complexity.

3.4 Micro Modeling Workflow

To create a model with elements satisfying the before mentioned characteristics, we imply this coarse modeling process in three steps:

1. First of all the designer has to be aware of what exactly he wants to do in each modeling step. Therefore our system guides the designer by showing possible workflow sequences and suitable previews of all operations.

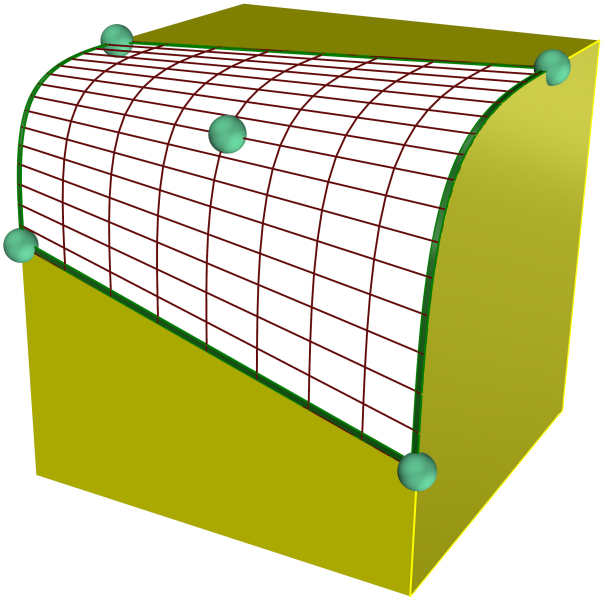

Figure 5: *Blended transition area.*

2. After the designer disclosed his decision, each action must then be declared by him. Not by performing low-level operations but instead by triggering abstract high-level operations which are capable to classify the designers action and to record the necessary meta information. These high-level operations can achieve the same geometrical result as familiar low-level operations. But they have a different structure and procedure where the designer instructs the system to do the operation and does not do it by his own. These operations are meaningful and summarize a set of single actions.

3. Because we do not want the designer to state all necessary meta information by himself, which would end in a long querying sequence, the system responses high-level actions with appropriate default solutions. Afterwards the designer can adjust these solutions by altering the meta information interactively.

During the whole process a lot of meta information is collected but not all of it is entirely provided by the user. A lot of it arises implicitly by choosing a tool, by applying an operation or by appropriate default settings. Another speed-up comes from the interchangeability and automation of our workflow, when the designer decorates new or empty parts of the model with already applied operations and previously stored categories.

In our interpretation of a modern CAD-system the role of the designer changes. Unlike common systems which are often geometry driven, our concept does not follow the *What You See Is What You Get* approach entirely. By restricting low-level attempts on the surfaces geometry and emphasizing the use of abstract

operations, the designer becomes more an instructor. We picture the working procedure of him more like drawing a construction plan and less in forming each single shape and feature manually. Besides the creative form finding process, we mainly want the designer to enrich the model with semantic information which distinguishes us from other modeling systems. In the following we will show how this can work in an exemplary workflow.

3.5 Exemplary Workflow

In this section we describe the concept behind our idea through a typical workflow by means of the example of a washing machine. Going through all modeling steps, we introduce all kinds of elements, operations and our fundamental layer system. The model was kept deliberately simple and could also be created by a constructive solid geometry (CSG) system. But our methods are conceived with the aim to construct freeform surfaces equivalent to the model.

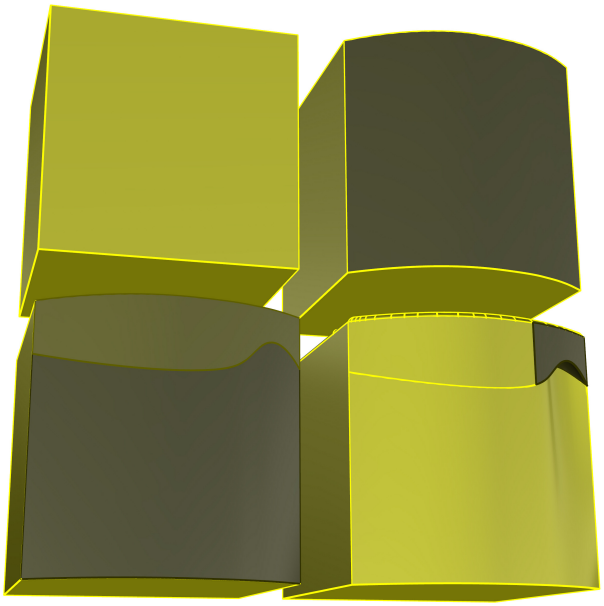

Figure 6: ***Top-left:*** *Base object;* ***Top-right:*** *Bended front area;* ***Bottom-left:*** *Front-area divided into panel-area (overhead) and door-area (below);* ***Bottom-right:*** *Panel-area divided into panel- and detergent-area. Deformation on door-area for the detergent dispenser and deformation on the top-area where the transition between the top-area and the rest was blended.*

Modeling Structure

The structure of our model is designed as a modular system with the main focus on interchangeability. On top of that system and at the very beginning of the workflow, there is the *base object*, the initial instance to work with, representing the coarse shape of the desired model. In our example the base object is a cube (figure 6). Each of its sides represent a specific user defined part on the surface, the *areas*. An area can be shaped like shown in our example where the designer bended the front area of the washing machine. From then on a bending operation is assigned to the front-area. But the main aim of areas is to organize the elements upon it. Therefor the designer can divide an area – like the bended front-area – into two separated areas. Each of them can then be modified individually like the door-area which was dented for the detergent dispenser.

Attaching Additives

Figure 7: *Attached additives colored in red.*

Furthermore an area can host supplementary features, the *additives*. Such an additive can be an entire model itself with a base object and areas or a plain geometrical object, depending on its complexity. In our example (figure 7) the panel-area is decorated with additives representing a display, some buttons, a rotary control and a coating for the detergent dispenser. Other additives are attached to the door-area for the door and the doorhandle.

Relations

The areas are connected through relations by default. All attached additives have at least one relation to its underlying area or other additives, describing its connectivity and behavior. In figure 8 the relation between the rotary control and the panel-area is displayed. This additive is connected to its area with a static connection, which means that in case of a deformation of the area, the distance between the additive and the areas border stays constant. The buttons on the panel-area are connected to the areas border and with each other. Control points adjust distances and the location of attachment.

Layer-System

Apart from their geometrical form, and their function as a host for additives, areas are predestinated to organize these additives by grouping them. But areas can also be layered. Like shown in figure 6 the door- and the panel-area were layered upon the front-area which is still available and not deleted. Moreover these layered areas are logically connected. The designer can flick through these layers and modify a certain one which has direct influence on the other layered areas automatically. This gives him the ability of saving a snapshot of a concrete part of the model for testing purposes or

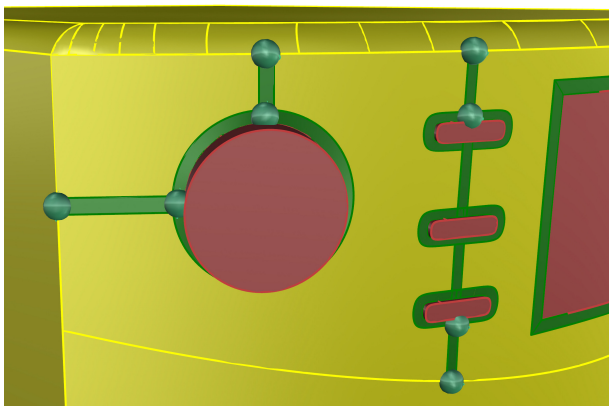

Figure 8: *Relations between additives and their area with control points.*

other design playthings. So different layered areas can be shaped differently and decorated with different additives, fully modular. As these areas can easily be exchanged, the designer can try out various constellations of the model in little amount of time.

This is another crucial distinction to other modeling systems. Going back in time in our solution means changing a layer and not changing the memory state. Thus a modification in the past has automatic influence in the future because of the relations between the layers. Layers give the designer the opportunity to just time travel selected parts (e.g. the front of the washing machine) and leave the rest of the model untouched. So various states of operations from various points in time can easily be combined.

Modeling Modes

The impact of an operation also depends on the chosen modeling-mode. In figure 9 the designer decided to modify the shape of the panel- and the door-area. On the left picture we see the old model. The right picture shows that he wanted the border between these two areas less curvy. So he modified the geometry of the border curve which is the coupler between both areas. Because this modification is conducted in semantic-mode, not only the geometry of this particular element was changed, also all relations to this curve are involved, including the adjacent areas and their attached additives. This means that obviously both areas were altered. But also the door was relocated, still fitting the same distance to the border and still retaining its size. Same goes for the buttons. The display was resized as the users intent defined it has to satisfy a fixed distance to the borders of the panel-area.

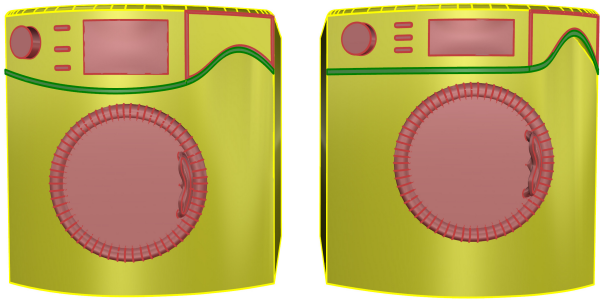

Figure 9: *Semantic-mode: Altering the shape between two areas.* ***Left:*** *Old shape with a curvy coupler between the areas.* ***Right:*** *New shape with a less curvy coupler.*

Figure 10 shows a doorhandle which was modified in geometry-mode (left: old state, right: new state). This operation has no impact an other elements and the alteration on this additive was only applied on its geometry. Here we can see both modes in contrast and how the same operation in different modes varies in their impact. In figure 9 the same operation in another mode also changed the whole arrangement of the additives. Whereas an operation in semantic-mode incorporates all gathered meta information, the geometry-mode only considers the pure geometry.

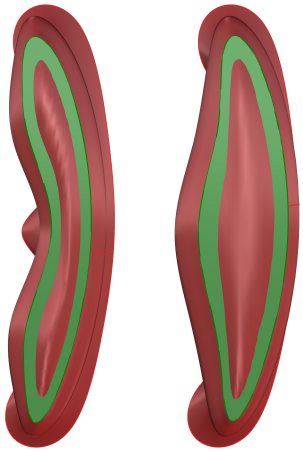

Figure 10: *Geometry-mode: Geometrical modification on the doorhandle.*

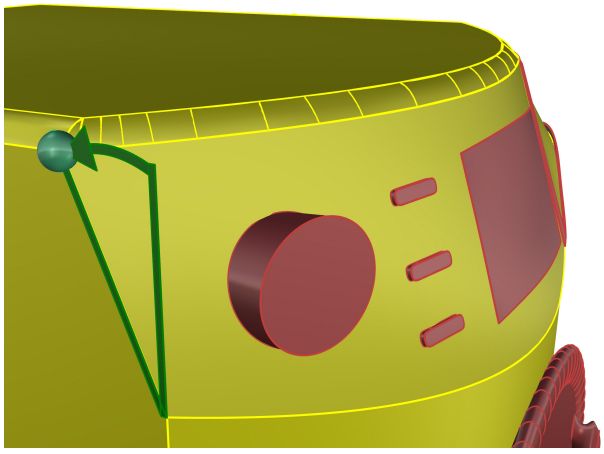

Figure 11: *Semantic-mode: Tilting the panel area.*

Another use case is shown in figure 11 where the panel-area was tilted and the attached additives on that area were automatically tilted too. The rotary control was automatically extruded, fitting its basis plane from before the modification.

4 SUMMARY AND FUTURE WORK

We have presented a modeling approach that enables a designer to construct a 3D model by building shapes and internal structures on an abstract level. Previews are accomplished by meshes and subdivision techniques while derivability of a CAD model based on the hierarchical structure of the abstract model is still guaranteed. The designer is equipped with construction tools and guided through the process of creating an object. For now we have focussed on the creation of the model starting from a base model. The presented tools were mainly meant to add features, both in geometric and in an abstract sense.

In the future we are going to extend the functionality by tools that modify the shape of a surface area. First to mention there is deformation. Neither control-point moving in a spline sense nor mesh deformation techniques in the sense of smoothing operators will be suitable to be used by a designer. Again we need a more abstract and non-technical view to the task. A surface deformation tool must be imaginable in a designer-context and could for example be interpreted as adding or removing clay from a certain surface region or to apply pressure to a rubber surface. Depending on which level our hierarchical model is attached to the surface area, we need to impose characteristics and restrictions to the allowed operations and of course to transfer the abstract modeling operator into a low level mesh and/or spline equivalent. Again geometric deformation and structural aspects will have to be executed simultaneously on the abstract model, the mesh and the spline surface-model.

5 REFERENCES

[AQRPHC12] Gerardo Alducin-Quintero, Alejandro Rojo, Francisco Plata, Arturo Hernandez and Manuel Contero, *3D Model Annotation as a Tool for Improving Design Intent Communication: A Case Study on its Impact in the Engineering Change Process*, Proceedings ASME 45011, Volume 2: 32nd Computers and Information in Engineering Conference, 2012

[BRC14] Yannick Bodeina, Bertrand Rose and Emmanuel Caillaud, *Explicit reference modeling methodology in parametric CAD system*, Computers in Industry 65 (1), 2014,

[Chew87] L. Paul Chew, *Constrained Delaunay Triangulations*, Proceedings of the Third Annual Symposium on Computational Geometry, 1987

[Catmull98] E. Catmull and J. Clark, *Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes*, Seminal Graphics, 1998

[DeBerg08] Mark de Berg, Otfried Cheong, Marc van Kreveld and Mark Overmars, *Computational Geometry - Algorithms and Applications, Third Edition*, Springer, 2008

[Diehl04] H. Diehl, F. Mueller and U. Lindemann, *From raw 3D-Sketches to exact CAD product models - Concept for an Assistant-system*, Proceedings of the First Eurographics Conference on Sketch-Based Interfaces and Modeling, 2004

[GU06] I. Ginkel and G. Umlauf, *Loop subdivision with curvature control*, Eurographics Symposium on Geometry Processing, 2006

[HPR00] JungHyun Han, M. Pratt and William C. Regli, *Manufacturing Feature Recognition from Solid Models: A Status Report*, IEEE Trans. Robotics and Automation 16 (6), 2000

[KMY06] Kyoung-Yun Kim, David G. Manley and Hyungjeong Yang, *Ontology-based Assembly Design and Information Sharing for Collaborative Product Development*, Computer Aided Design 38 (12), 2006

[LLM10] Ming Li, Frank Curd Langbein and Ralph Robert Martin, *Detecting Design Intent in Approximate CAD Models Using Symmetry*, Computer-Aided Design 42 (3), 2010

[Loop87] C.T. Loop, *Smooth Subdivision Surfaces Based on Triangles*, Master Thesis, University of Utah, 1987

[MHKO03] Duhwan Mun, Soonhung Han, Junhwan Kim and Youchon Oh, *A set of standard modeling commands for the history-based parametric approach*, Computer-Aided Design 35 (13), 2003

[Shewchuk05] Jonathan Richard Shewchuk, *General-Dimensional Constrained Delaunay and Constrained Regular Triangulations, I: Combinatorial Properties*, Discrete and Computational Geometry, 2005

[SK05] M. Sokovic and J. Kopac, *RE (reverse engineering) as necessary phase by rapid product development*, Journal of Material Processing Technology, 2005

[TDM10] Stefano Tornincasa and Francesco Di Monaco, *The Future and the Evolution of CAD*, 14th International Research/Expert Conference - Trends in the Development of Machinery and Associated Technology, 2010

[ZHH96] Robert C. Zeleznik, Kenneth P. Herndon and John F. Hughes, *SKETCH: An Interface for Sketching 3D Scenes*, ACM SIGGRAPH Courses, 2006