Feature Driven Rule Based Framework for Automatic Modeling of Organic Shapes in the Design of Personalized Medical Prosthetics

Sajjad Baloch[1], Konrad Sickel[2], Vojtech Bubnik[3], Rupen Melkisetoglu[1], Sergei Azernikov[1], Andreas Reh[4], Artem Boltyenkov[4], and Tong Fang[1]

[1] Siemens Corporate Research, Princeton, NJ, USA
[2] Friedrich-Alexander University, Erlangen, Germany
[3] Siemens Hearing Aid Instruments, Piscataway, NJ, USA
[4] Siemens Audiologische Technik GmbH, Erlangen, Germany

Abstract. We propose a novel framework for the personalized design of organic shapes that are constrained to exhibit conformity with the underlying anatomy. Such constrained design is significant for several applications such as the design of implants and prosthetics, which need to be adapted to the anatomy of a patient. In such applications, vaguely defined work instructions are usually employed by expert designers to carry out a sequence of surface modification operations using interactive CAD tools. Our approach involves the abstraction of the work instructions and the expert knowledge into feature dependent machine interpretable rules in a *Knowledge Base*. Robustly identified *canonical* set of anatomical features are then employed to determine concrete surface shaping operations by a *Smart Shape Modeler*. These operations are eventually performed sequentially to adapt a surface to a target shape. The versatility of our approach lies in a priori defining an entire design workflow through a scripting language, thereby yielding a high degree of automation that is completely flexible and customizable via scriptable rules. Consequently, it eliminates tedious manual intervention and offers desirable precision and reproducibility. We validate this framework with a practical application – automatic modeling of shells in hearing aid (HA) manufacturing (HAM).

1 Introduction

Computer aided modeling of organic shapes, such as implants and prosthetics, constitutes an important element of digital manufacturing [5]. Typically, it has two major requirements. First, the designed surfaces should be able to hold essential components such as electronics. Second, they should demonstrate a tolerable degree of conformity with the underlying anatomy. The latter is particularly important to ensure that they comfortably fit to the anatomy of a patient. Due to biological variability, the design of such shapes has traditionally been carried out manually, where expert designers start with a surface representation of the underlying anatomy as an *anatomical template*, which may be acquired through 3D laser or CT scans [5]. The template is then modified by a sequence of manual operations, which are vaguely described in application specific work instructions.

H. Liao et al. (Eds.): MIAR 2010, LNCS 6326, pp. 128–138, 2010.

To this end, the designers rely on certain biomarkers on a template to ensure the conformity between the anatomy and the resulting shape.

The problem is highly significant in various areas, such as HAM, orthopedics, orthotics, and orthodontics. HAM [8], for instance, requires tedious amount of manual involvement. In-the-ear HAs are generally custom made to ensure a comfortable fit to the ear(s) of a patient. A wide range of HA configurations exist to cater for various levels of hearing losses and styles (Fig. 5). HA designers, therefore, have to rely on their experience along with lengthy work instructions for carrying out surface modifications [8] via interactive CAD tools (iCT) [1].

The amount of variation across individuals hinders complete automation. Recently, there have been some advances in this area. In the context of HAs, [6] employed active shape models for describing the shape of a human ear. [7] used Markov random fields for highlighting gender differences. Unal et al. [9] proposed PCA based learning for the design of HA shells. Major limitations of their approach included minimal interpretability for the HA designers, and lack of explicit constraints for component placement. Our approach addresses these limitations by transforming the iCTs to fully automatic tools. In general, iCTs allow a user to specify, for instance, a surface cutting plane by drawing a line in a CAD software, and then to apply the cutting. The design process is simplified if the plane placement and the cut is done automatically. Other examples include automatically painting a region on a surface for local deformation, or positioning additional components at a specific location relative to the surface as in Fig. 1 for HAs.

The proposed framework automates modeling workflows by translating human readable work instructions, and expert knowledge to machine interpretable rules. First, a set of abstract rules is defined in a *knowledge base*, and implemented via a specially designed scripting language. The rules generalize the definition of various selections, e.g., plane definition, and the execution of surface editing operations like cutting. The concrete definition of the operations or selections is determined by a *smart shape modeler* that combines rules with associated anatomical and geometrical features found by an automatic *feature detector*. Once a concrete operation is determined, CAD tools are invoked for actual editing.

Since the rules are implemented via scripts, various surface modeling operations are easily configured for a given application, yielding a high degree of customization and flexibility. The framework automates a process that otherwise exhibits large amount of uncertainty stemming from surface variability. To the best of our knowledge, feature based rules for automated modification of organic surfaces via abstraction of surface modification operations has not been previously proposed. We highlight the effectiveness of our framework by replicating the entire workflow pipeline for the modeling of in-the-ear HAs and substantiate it with experimental results. A fully functional implementation of this framework has already been rolled into the production floor of a major HA manufacturer.

2 Motivation

CAD software allows a user to perform surface shaping operations to adapt them to a final optimal shape. A typical example is provided in Fig. 1, where a sequence

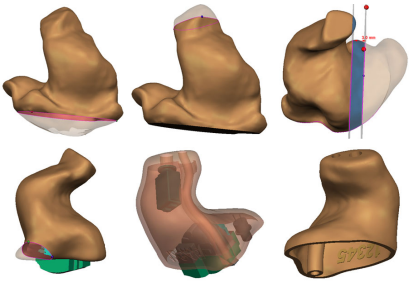

Fig. 1. Modeling operations. Top row: Canal tapering, and helix modification. Bottom row: Vent and electronics placement. The last item corresponds to the finished shell with labeling.

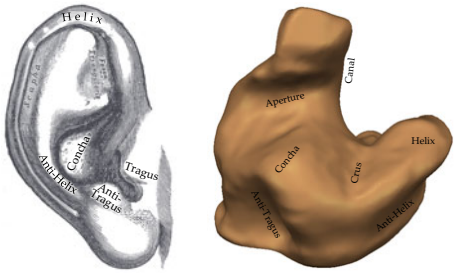

Fig. 2. The human ear anatomy (courtesy [4]) and its 3D reconstruction

of operations is carried out to deform the surface of a 3D outer ear impression to the surface of a finished HA shell. Irrespective of the application, and the nature of an operation, it is typically applied at a particular region or location on a surface. Due to the inherent variability in organic surfaces, it has not been traditionally possible to automate the surface shaping process. A user has to manual specify the operations of interest. Here, we aim for modeling automation by eliminating manual processing to yield consistency, reproducibility, and quality of the designed surfaces. This leads to rapid manufacturing [5], with reduced number of recalls. Although our approach was motivated by HAM, it is fairly general and is applicable to similar applications.

3 Problem Formulation

The design of a surface is defined as a 2-tuple $(\mathcal{M}_s, \mathcal{M}_t)$ and an associated sequence of surface operators $T^{\mathcal{F},\mathcal{O}} = (T_1^{\mathcal{F},\mathcal{O}_1}, \ldots, T_n^{\mathcal{F},\mathcal{O}_n})$, where $\mathcal{M}_s$ and $\mathcal{M}_t$ respectively denote the source and the target shapes:

$$\mathcal{M}_t = (T_n^{\mathcal{F},\mathcal{O}_1} \circ \ldots \circ T_1^{\mathcal{F},\mathcal{O}_n})(\mathcal{M}_s). \tag{1}$$

A given surface is modified when acted upon by a surface operator $\mathcal{M}_i = T_i^{\mathcal{F},\mathcal{O}_i}\mathcal{M}_{i-1}$. Each operator, $T_i^{\mathcal{F},\mathcal{O}_i} : \mathcal{M}_{i-1} \rightarrow \mathcal{M}_i$, is parameterized by a set of anatomical landmarks $\mathcal{F} = \{f_j, j = 1, \ldots, m\}$, and options $\mathcal{O} = \{\mathcal{O}_1, \ldots, \mathcal{O}_n\}$. For simplicity, we drop the explicit dependence of the operator on the *options* for later discussion. Unless specified otherwise, the dependence is always implied.

For a given source, and a pre-specified target shape, $\mathcal{F}$, and $T^{\mathcal{F}}$ are typically defined by process engineers through work instructions, which are executed by trained technicians. Examples of operators include but are not limited to cutting,

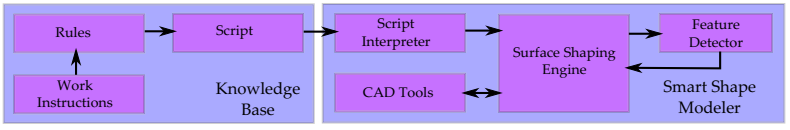

Fig. 3. Smart modeling framework for automation via feature driven rules

rounding, tapering, shrinking, scooping, and even more complex boolean and diffeormorphic operations.

For a given $\mathcal{M}_s$ and pre-specified target $\mathcal{M}_t$, problem under consideration is to estimate the landmarks $\mathcal{F}$ robustly, which will then uniquely identify a subsequence of operations $T^{\mathcal{F}}(R)$ according to certain rules $\mathcal{R} = \{r_k, k = 1, \ldots, K\}$. For complete automation, $\mathcal{F}$ should be consistent for all possible configurations of $\mathcal{M}_s$ and $\mathcal{M}_t$, and rules should be comprehensive to uniquely determine $T^{\mathcal{F}}(\mathcal{R})$. For notational simplicity, we have not indicated the dependence of $\mathcal{F}$ on the surface $\mathcal{M}_{i-1}$, but in general $\mathcal{F}$ does not constitute static landmarks, and have to be determined dynamically.

4 Workflow Automation Framework

Surface shaping steps are generally defined via work instructions. We model this process through a *smart modeling framework* (Fig. 3) that combines this information with the surface landmarks to perform surface shaping operations. Major components include a *knowledge base* (KB), and a *smart shape modeler* (SSM).

4.1 Knowledge Base

The role of the KB is to digitally represent a workflow through a set of rules $\mathcal{R}$ derived from the work instructions. The rules are application specific and encompass all realizations of the class of surfaces. They describe (1) how to perform various steps, and (2) which features to utilize in each step. A step, in turn, consists of one or more surface modification operations $T^{\mathcal{F}}(\mathcal{R})$, and combines features $\mathcal{F}$ for carrying them out. For instance, for cutting a surface, the KB must specify a rule $r \in \mathcal{R}$ that defines where to perform a cut consistently for a wide range of surfaces. This rule should be able to compute the cutting plane, as well as the type of the cut. Although surfaces may exhibit variability, their class membership ensures that some canonical set of features $\mathcal{F}$ is always identifiable, and is sufficient to define such a plane. The existence of the required features drives the entire framework. The KB implements the rules in a scripting language, allowing complete flexibility for various shell models and target shapes.

4.2 Smart Shape Modeler

The function of the SSM is to design a surface (or shape) via CAD tools, which are invoked automatically as directed by the KB. To this end, a *surface shaping engine*

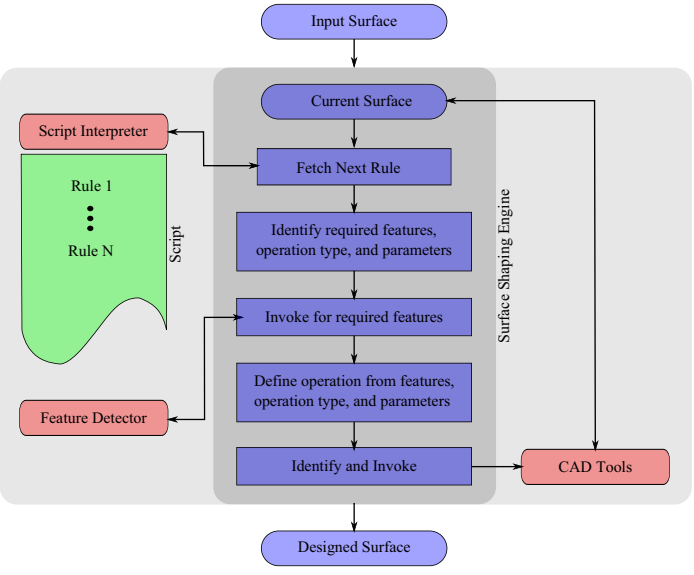

Fig. 4. Interaction of SSE with various components

(SSE) interacts with a *script interpreter* (SI) to sequentially feed it with rules from a digitized workflow. Each rule r is translated to an operation $T_i^F(r)$ that is defined by features $F \subset \mathcal{F}$. Sequential execution of the script rules is ensured through the SI, which parses the rules, and maintains the state of the current rule in a digitized workflow.

As shown in Fig. 4, the role of the SSE is to route the script commands to various components. Once a surface is specified, the SSE resets the current state of the surface, and requests the SI for the next rule. From the rule, it identifies the set of required features, operation type, and the associated parameters (if any). The *feature detector* (FD) is then invoked to detect required features, which are used to uniquely define the actual surface modification operation $T_i^F(r)$. This information leads to a selection of the desired *CAD tool* (CT) with correct parameters to perform T_i on the surface. For instance, for local scooping or smoothing, the KB informs the SSE about (1) the operation (scooping or smoothing); (2) the scooping or smoothing parameters; and (3) the identifier for the corresponding region of interest. Based on this identifier, the FD provides the SSE the area to be scooped or smoothed. The process is repeated until the SI is exhausted of all applicable rules, and the current surface is outputted as the designed shape.

The advantage of this modular approach is that the workflow modeler and the SSE constitute a fully automated smart modeling system that is not specific to a

particular application. One may readily switch between applications by modifying the scriptable rules $\mathcal{R}$ and the associated feature set $\mathcal{F}$. In addition, the smart modeling system is highly configurable via scripting.

5 Application to Hearing Aid Modeling

In this section, we consider the modeling of HA shells [8]. Ear impressions are first acquired by an audiologist by inserting a mold through an injector deep in the ear canal. The mold is allowed to settle, before taking out the impression and carrying out a 3D digital scan. The reconstructed surface then represents the shape of the interior of the ear. During the modeling process, an operator manually shapes an HA and embeds the electronics (Fig. 1). Some major HA configurations are shown in Fig. 5. Despite similar in appearance, ear impressions are unique and exhibit large amount of variability across individuals. The missing data as well as the excess material pose serious challenges to complete automation. Design inaccuracies potentially result in undesirable fit or performance.

Fig. 5. Various types of HA shells: (Left Most) In-the-Ear (ITE); (Right Most) Completely-In-the-Canal (CIC)

5.1 Feature Detection

A *canonical* set of anatomical features[1] [2] is incorporated in the FD of Fig. 3. It captures comprehensive information about an ear for defining a set of generalized rules. It include points, planes, areas, and curves, and capture the bumps, bends, convexities, concavities, ridges, valleys and the intersections of ridges as illustrated in Figs. 2 and 6. Expert designers already take some of these features into account while manually designing an HA shell. For instance, canal tapering may be done at the second bend plane (Fig. 6(a)), where part of the surface above this plane is removed. Inter-tragal notch (ITN) and crus-side ridge (CSR) (Fig. 6(b)) are used for the placement of vents. Crus area (Fig. 6(c)) is scooped, bumps are rounded, and helix region is tapered for comfort. Feedback seal area is scooped for a firm grip.

We have developed algorithms [2] for the robust detection of these features. The validation of these algorithms was carried out on a dataset of 198 impressions. They were tweaked until a detection rate of 95% and a detection quality in excess of 80% was achieved. Detection quality, in turn, was computed by comparing the detected results with expert manual annotations. In this paper, we assume that these features have already been robustly identified.

[1] These features are consistent across individuals.

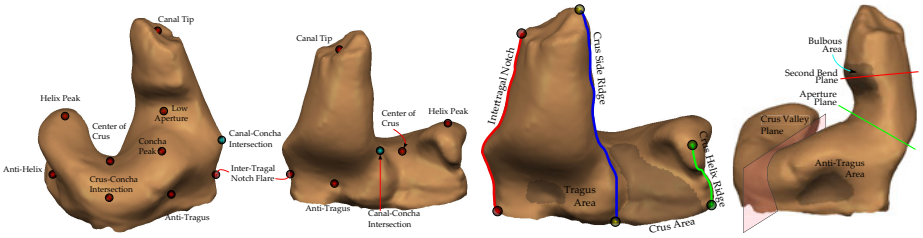

Fig. 6. Anatomical Features: (a) Points; (b) Curves and areas; (c) Planes and areas

5.2 Scripting Language

For realizing the rules in SI, a scripting language with context free grammar is designed using Bison and Flex [3]. The language supports standard data types, in addition to geometric primitives such as points, planes, and matrices, which allow easy manipulation of surface meshes. It includes control structures, such as `if-then-else`, and `for` and `while` loops. Specialized functions include interfaces to CAD tool APIs, FD, and primitive surface manipulators. The reason for developing a customized language is to allow simple handling of the scripts, and its easy integration with a CAD application, while keeping the latter flexible.

5.3 Scriptable Rules

We now briefly describe the role of features in driving rules for performing a typical HA modeling operation. In general, different HA configurations employ different combinations of features, and these combinations are selected through `if-then-else` statements in the script. Examples of rules include `CutCanalAtSeco- ndBend`, `SmoothAtXYZAreaFeature`, and `CutShellWithPlaneDefinedByXYZPoint Features`.

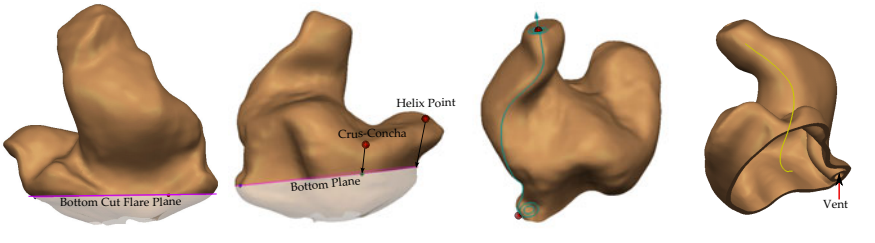

Fig. 7. Automatic placement of the vent: (a) Set up flare plane from anti-helix, tragus, and anti-tragus; (b) Set up the bottom plane by shifting helix peak and crus-concha intersection down and tragus towards the notch; (c) Set up the vent by using the ITN top and bottom points and vent thickness and diameter parameters; (d) Commit vent placement via CAD tool.

Now we describe the usage of the framework, the scripting rules, and the features via the vent placement step. As mentioned earlier, a vent is a tubular attachment to the canal of an HA shell. In the simplest configuration, it is placed along one of the canal ridges because of the acoustic principles and the size constraints. The side selection (ITN or CSR), the starting and ending points, and the wall thickness of a vent may easily be specified through scripting rules. Since the vent runs from the tip of the canal to the bottom of a shell, the script should first define the tip and the bottom planes. The shell bottom may be defined in terms of three point features – helix peak, crus-concha intersection, and tragus.

```
// Move helix peak down by HelixShift
ShiftedHelixPeak = HelixPeak - HelixShift * BottomCutFlarePlane.Normal
// Move tragus
ShiftedTragus = Tragus - TragusShift * BottomCutFlarePlane.Normal
// Move down by ConchaShift
ShiftedCCIPoint = CrusConchaIntersection - ConchaShift * BottomCutFlarePlane.Normal
SanityCheck(ShiftedHelixPeak, ShiftedTragus, ShiftedCCIPoint)
BottomPlane = Plane(ShiftedHelixPeak, ShiftedTragus, ShiftedCCIPoint)
if Below(BottomPlane, Tip) then
    BottomPlane = - BottomPlane
end
```

Once these features are detected, a CAD cutting tool should be invoked on the surface to get rid of the excess material:

```
CutSurface(BottomPlane, Tip)
```

The starting and ending points of the vent are required for its placement. Based on the vent side information, a user may opt to place the vent along ITN, or CSR. The starting and ending points in either case are different, and are found through features, such as the ITN top and bottom points, or CSR top and bottom points. The thickness of the vent wall is taken into account to ensure that the end points fall at the right place on the surface. Eventually, a vent placement CAD tool is invoked to commit the operation:

```
if IsNotchSideVent then
    VentTop = InterTragalNotchTop
    VentDirection = CrusRidgeTop - InterTragalNotchTop
    VentBottom = InterTragalNotchBottom

else
    VentTop = CrusRidgeTop
    VentDirection = InterTragalNotchTop - CrusRidgeTop
    VentBottom = CrusRidgeBottom
end
VentOffset = VentWallThickness + VentDiameter / 2
Normalize(VentDirection)
VentTop = VentTop + VentDirection * VentOffset
PlaceVent(VentTop, TipPlane.Normal, VentBottom, BottomPlane.Normal)
```

KB should contain these instructions in the form of a script, which are sufficient for a primitive vent placement operation. When initialized with a given surface, SSE queries the rule from SI, and executes each instruction. Features are detected on the fly, and the surface modification operation is determined at the run-time. The result is given in Fig. 7. Without loss of generality, these rules correspond to a simplified version of the actual vent placement.

6 Experiments

In this section, we model HAs from 3D ear impressions shown in Fig. 8(a). Results for automatic modeling are given in Fig. 8(b), which indicate excellent canal modification, vent placement, receiver positioning, and labeling even with missing and highly noisy data as in the bottom row.

6.1 Validation

Qualitative Evaluation of Individual Operations. First level of validation was qualitatively carried out by 6 different experts on a dataset of 39 samples. Each expert was asked to rate the outcome of each surface shaping operation according to a *quality matrix* with $\{0 = \text{Unusable}, 1 = \text{Little Modification Required}, 2 = \text{Acceptable}, 3 = \text{Perfect}\}$.

The average of the quality matrix for some common operations as rated by the experts is given in Table 1. Overall the results are very promising. Excellent

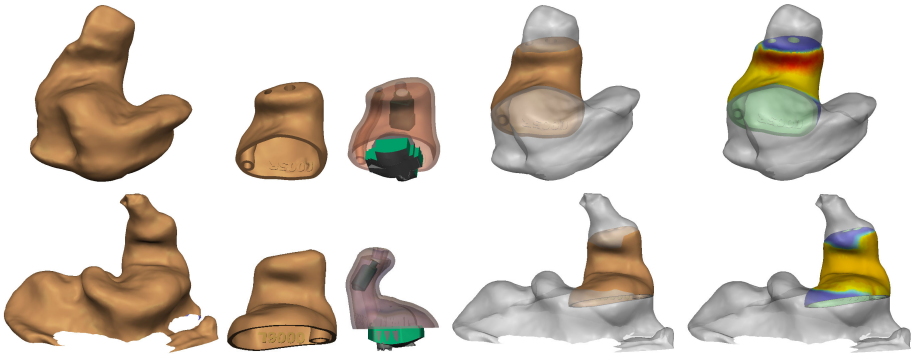

Fig. 8. Automatic detailing: (a) 3D impression; (b) Finished HA shell; (c) Shell with associated electronics; (d) Finished HA shell superimposed on the 3D impression shown with transparency; (e) A 3D error map.

Table 1. Quality matrix for common operations (only for highlighting the appropriateness of the corresponding operations; their description is beyond the scope of the paper)

Operation	Mean ± (Std.Dev.)	Operation	Mean ± (Std.Dev.)
Waxgaurd cut	2.57 (0.86)	ITC crus cut	2.45 (0.78)
Optional vent cuts	2.60 (0.76)	ITE cymba rounding	2.43 (0.77)
Receiver Placement	3.00 (0.00)	Canal Tip cut	1.74 (0.98)
ITE anti-helix filling	2.25 (1.10)	Labeling	1.86 (1.26)
ITC measured cut	2.10 (1.01)	ITE Crus scooping	1.07 (1.14)
Vent placement	2.04 (0.93)	Canal thickening	1.72 (1.13)
Excess material cut	2.21 (0.86)	CIC Measured Cut	1.14 (0.81)
Receiver Hole Placement	1.97 (1.06)	Total	2.08 (0.89)

performance may be noticed for some operations. Although for the others, the experts were not so satisfied, they were still towards the higher end, with the exception of ITE crus scooping, and CIC measured cut. In future, we plan to utilize this matrix as a baseline for fine turning the feature based rules.

Quantitative Evaluation of the Designed Shape. As the second validation, we identified the agreement between a source surface $\mathcal{M}_s$ and the estimated target (designed) surface $\hat{\mathcal{M}}_t$, to quantitatively highlight the fitness of an HA. An error map D was defined on $\hat{\mathcal{M}}_t$ as:

$$D(\mathbf{u}) := \mathcal{S}(\hat{\mathcal{M}}_t(\mathbf{u}), \mathcal{M}_s)||\hat{\mathcal{M}}_t(\mathbf{u}) - \mathcal{M}_s(h(\mathbf{u}))||, \tag{2}$$

where $\mathbf{u}$ parameterizes $\hat{\mathcal{M}}_t$, h represents the closest point mapping from $\hat{\mathcal{M}}_t$ to $\mathcal{M}_s$, and $\mathcal{S}$ denotes a sign function which assigns a positive value if the $\hat{\mathcal{M}}_t(\mathbf{u})$ is outside $\mathcal{M}_s$. The positive values in D indicate where $\hat{\mathcal{M}}_t$ protrudes outside $\mathcal{M}_s$. It should be noted that the source surface represents the inner surface of the ear, and therefore, the positive values means uncomfortable fits. For visualization, we normalize error maps to $[-1, +1]$ prior to applying a colormap with dark blue representing -1, and dark red representing $+1$.

Results given in Fig. 8(e) indicate a high level of agreement with the original 3D impression and the finished product. Red values on the canal correspond to the so-called *feedback seal*. The feedback seal is a selection of a narrow band of triangles on a canal, which is intentionally scooped outwards to ensure a firm grip. It is pointed out that this operation was also performed entirely automatically. Other than the red values corresponding to the feedback seal, mostly yellow and blue values in the error map show that the HA shell stays inside the impression, which amounts to a comfortable fit.

7 Conclusions

This paper has presented a powerful framework for the automation of surface modeling workflows. It translates human readable work instructions to machine interpretable rules. Consequently, interactive CAD operations get simulated by automatic surface shaping operations dictated by the rules. Rules are specified through a special scripting language, and are based on the features detected on a surface. Flexibility of this approach lies in the diversity of rules, which allows handling various configurations in digital manufacturing. The modular nature of the framework makes it fairly general and applicable to a wide range of applications, such as modeling of HAs, dentures, braces, orthotic devices, and orthopedic joint replacements. Results indicate the effectiveness of our approach. With some intervention, it yields industry standard results. At times, a user likes to modify the proposed result, rendering the currently devised rules semi-automatic. In future, we plan to fine tune the rules for full automation, which will allow batching of the work orders.

References

1. 3Shape A/S: ShellDesigner, DentalDesigner, http://www.3shape.com/
2. Baloch, S., Melkisetoglu, R., et al.: Automatic detection of anatomical features on 3D ear impressions for canonical representation. In: MICCAI 2010 (2010)
3. Donnelly, C., et al.: The Bison Manual, Using the YACC Compatible Parser Generator (2003)
4. Henry, G.: Gray's Anatomy: Descriptive and Surgical (1858)
5. Masters, M., et al.: Rapid manufacturing in the hearing industry. In: Hopkinson, N., Hague, R., Dickens, P. (eds.) Rapid Manufacturing: An Industrial Revolution for the Digital Age (2006)
6. Paulsen, R.R., et al.: Building and testing a statistical shape model of the human ear canal. In: Dohi, T., Kikinis, R. (eds.) MICCAI 2002. LNCS, vol. 2489, pp. 373–380. Springer, Heidelberg (2002)
7. Paulsen, R.R., et al.: Shape modelling using Markov random field restoration of point correspondences. In: Taylor, C.J., Noble, J.A. (eds.) IPMI 2003. LNCS, vol. 2732, pp. 1–12. Springer, Heidelberg (2003)
8. Slabaugh, G., Fang, T., et al.: 3D shape modeling for hearing aid design. IEEE Signal Processing Magazine 5(25), 98–102 (2008)
9. Unal, G., et al.: Customized design of hearing aids using statistical shape learning. In: Metaxas, D., Axel, L., Fichtinger, G., Székely, G. (eds.) MICCAI 2008, Part I. LNCS, vol. 5241, pp. 518–526. Springer, Heidelberg (2008)