Snake-Based Segmentation of Teeth from Virtual Dental Casts

Thomas Kronfeld, David Brunner and Guido Brunnett

Chemnitz University of Technology, {tkro, brunner, brunnett}@cs.tu-chemnitz.de

ABSTRACT

During the past years several innovative and technological developments have been made in oral surgery. Today, digitized dental casts are common for simulation and planning of orthodontic interventions. In order to work with these models, knowing the exact position of the teeth is of high importance. In this paper, we present a new method for tooth segmentation with minimal user interaction. At the beginning, an initial estimate for the separating curve between the teeth and the gum is computed and optimized by use of an active contour. The second step calculates the dental arch and the interstices between the teeth. In order to detect each tooth surface exactly, we finally position a snake around the cusp of each tooth.

Keywords: active contours, snakes, tooth segmentation, mesh segmentation.
DOI: 10.3722/cadaps.2010.221-233

1 INTRODUCTION

In orthodontics surgery, dental plaster casts of the patient's denture are made to simulate and plan interventions (inlays, bridges, malocclusions). Traditionally, clinicians analyze these plaster casts manually. In the past two decades, computerized systems became an important factor in dental surgery due to the improvement of three-dimensional scanning devices (for example [5, 6, 15]).

In this paper, three-dimensional dental models are represented as triangular meshes. As well as in the traditionally case, the first step is the accurate separation of the teeth from the digitized dental model. After the exact position of each tooth is determined, one is able to measure orthodontic features and simulate orthodontic procedures such as tooth rearrangement.

Tooth segmentation, in general, is a difficult task, since teeth occur in different shapes and their arrangements vary substantially from one individual to another. Furthermore, when dealing with models with severe malocclusion, interstices may be missing. A lot of automatic methods have already been presented (for a recent survey see [21]). None of these methods have been designed to specifically deal with teeth and using them directly for tooth segmentation will produce bad results. Algorithms which were specifically designed to deal with dental models provide better results. Most of them, however, require time-consuming and error-prone user interaction. The method proposed by Kondo et al. [12] is the only highly automated tooth segmentation method we know. At first, the user selects four reference points on the surface of the mesh. After that, the algorithm computes two range images: the plan-view and the panoramic range image. The plan-view range image is used to determine the dental arch, which is used as a reference to calculate the panoramic range image. Both images are processed separately and composed into one indicator function, which is used to detect the interstices between teeth. Unfortunately, the method produces poor results, when processing models with severe malocclusions. The approach of Li et al. [13] uses a modified watershed algorithm for manual segmentation. At first, the user selects some points on the surface of each tooth. Starting at the selected markers, the method performs a region growing method until it reaches vertices with negative mean curvature. As stated by them and many other researchers, the boundary between tooth and gum is determined by vertices with negative mean curvature values. For example, Zhao et al. [28] describe a method based on the calculation of mean curvature, directly. Initially, the user specifies a certain threshold as an upper curvature bound. Vertices with smaller mean curvature are grouped into connected regions. Then they apply morphological operations to avoid multiple branching and extract the skeleton of each region to obtain thinner boundary lines. In order to achieve a complete segmentation, the user has to remove noisy lines and to connect open boundary lines manually. A similar method was suggested by Tian-ran et al. [23]. Sinthanayothin and Tharanont [22] propose two different manual segmentation methods. The first one is based on the selection of landmarks that are located at the transition between tooth and gum. These landmarks are then interpolated using a cubic spline to complete the boundary. Tooth segmentation using cutting planes is the second described method. Here, the user needs to position cutting planes between the teeth, so that each tooth is enclosed by two planes.

Beside the methods mentioned above there are some commercial products [1, 3, 4] but the details of their algorithms have not been published. The workflow is as follows: the dentist has to send the dental plaster cast to the company. After a few days the dentist receives a data file which contains the digitized and segmented dental cast. Now, one can rearrange each tooth or fit in inlays, but without the possibility to change the segmentation.

In this paper, we follow a completely different approach. Our method for tooth segmentation is based on active contour models (also known as snakes). The active contour model was originally developed by Kass et al. [11] to detect salient features in two-dimensional images. The aim of their research was to find connected boundaries based on low-level information such as the gradient magnitude of an image. They postulated that any object boundary could be described mathematically using a parameterized curve. The curve is initially placed outside the object and evolves under the influence of internal und external energies until it aligns with features of interest within an image. The internal energy derives directly from the curve and serves as a smoothness constraint, while the external energy derives from the image and forces the curve to lock on salient image features. The overall evolution process is obtained by energy minimization.

Milroy et al. [14] extends the original snake model to three-dimensional surface for segmentation purposes in reverse engineering. In their approach, the snake evolves directly on the surface. The user specifies a small closed contour within the region to be segmented, which then grows under the influence of an internal pressure force until it reaches vertices with local curvature maxima. Each vertex of a snake is moved to one of two adjacent positions on the surface of the mesh, perpendicular to the snake curve. The energy model, however, requires a locally continuous surface in the neighborhood of each vertex. For that purpose, the region surrounding each vertex is approximated by a second order surface. A simplified model for triangle meshes was proposed by Jung et al. [9]. Here, snake elements are located on adjacent vertices. During an iteration, each vertex of the snake moves to one of its neighboring vertices on the mesh, in order to reduce its energy as much as possible. The evolution is carried out according to the greedy principle, which was first introduced for active contours by Williams and Shah [25].

The rest of this paper is organized as follows. In section 2 we review the mathematical background of the proposed method. At first we calculate possible tooth boundaries (termed: features) and improve them by suppressing noise (section 2.1). After that, the cutting plane between the teeth and the gum is estimated (section 2.2) and converted into one or more snakes. The energy functional as well as the evolution equation of these snakes are outlined in section 2.3. Our newly defined external energy functional is presented in section 2.4. After the snake evolution converged, we calculate the exact position of interstices between neighboring teeth by using the *edge tangent flow* (section 2.5). As a last step, each tooth is separated individually. Therefore, we initialize a snake at the crown of a tooth, which then grows until it reaches the transition between tooth and gum. A complete overview of our approach, as well as a discussion on implementation details is presented in section 3 (see also figure 1). Results are demonstrated in section 4, and conclusions are drawn in section 5.

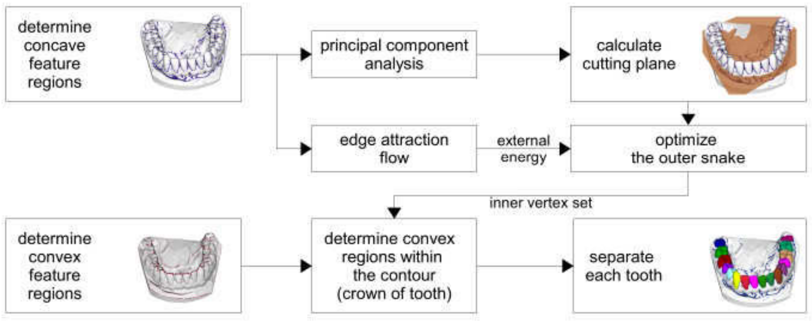

determine concave feature regions
principal component analysis
calculate cutting plane
edge attraction flow
external energy
optimize the outer snake
inner vertex set
determine convex feature regions
determine convex regions within the contour (crown of tooth)
separate each tooth

Fig. 1: Snake-based segmentation process.

2 MATHEMATICAL BACKGROUND

The three-dimensional dental model is represented using a triangular mesh $\mathcal{M} = (\mathcal{V}, \mathcal{T})$, which consists of a set of vertices $\mathcal{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_{n_\mathcal{V}})$ and a set of indexed triangles $\mathcal{T} = (t_1, \ldots, t_{n_\mathcal{T}})$ where each vertex $\mathbf{v} \in \mathcal{V}$ represents a unique position in $\mathbb{E}^3$ and each triangle $t \in \mathcal{T}$ is defined by three vertices $t = \{(\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k) \mid \mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k \in \mathcal{V}; \mathbf{v}_i \neq \mathbf{v}_j \neq \mathbf{v}_k\}$. The set of edges $\mathcal{E}$ is given by $\mathcal{E} = \{(\mathbf{v}_i, \mathbf{v}_j) \mid \mathbf{v}_i, \mathbf{v}_j \in \mathcal{V}; 0 \leq i, j \leq n_\mathcal{V}\}$ where $\mathbf{v}_i \neq \mathbf{v}_j$ belong to the same triangle $t \in \mathcal{T}$. At each vertex $\mathbf{v} \in \mathcal{V}$ a normal vector $\mathbf{n}(\mathbf{v})$ is defined with unit length $\|\mathbf{n}(\mathbf{v})\| = 1$. A vertex $\mathbf{w} \in \mathcal{V}$ is called neighbor of a vertex $\mathbf{v} \in \mathcal{V}$ iff the edge $(\mathbf{v}, \mathbf{w}) \in \mathcal{E}$ exists. The set of neighbors of a given vertex $\mathbf{v} \in \mathcal{V}$ is defined by $\mathcal{N}(\mathbf{v})$.

2.1 Feature Region

In order to detect regions of interest on the mesh by using snakes, one has to assign an attribute to each vertex of the mesh. In general, surface features are defined using curvature information, since curvature intrinsically describes the local shape of a surface.

Consider a continuous, oriented, and unbounded surface S and a point $\mathbf{v}$ which lies on S along with its unit normal vector $\mathbf{n}(\mathbf{v})$. One can construct a plane $P(\mathbf{v})$ that contains $\mathbf{v}$ and $\mathbf{n}(\mathbf{v})$. The intersection of this plane with S results in a curve c on the surface. This curve has an associated normal curvature $\kappa_\mathrm{n}(\mathbf{v})$ at the vertex $\mathbf{v}$. The curvature value does not however specify the surface curvature of S at $\mathbf{v}$, since the constructed plane is not unique. If we rotate the plane $P(\mathbf{v})$ around the unit normal vector $\mathbf{n}(\mathbf{v})$, we receive a new contour along with its own curvature value. Among all these curves, there is one with minimal curvature $\kappa_1(\mathbf{v})$ and one with maximal curvature $\kappa_2(\mathbf{v})$. These two curvatures are also known as the principal curvatures of S at point $\mathbf{v}$.

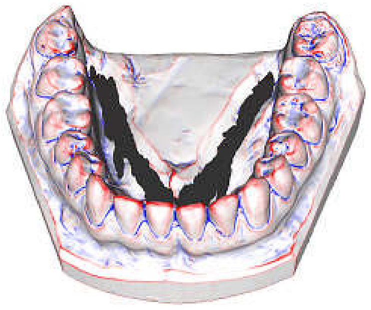
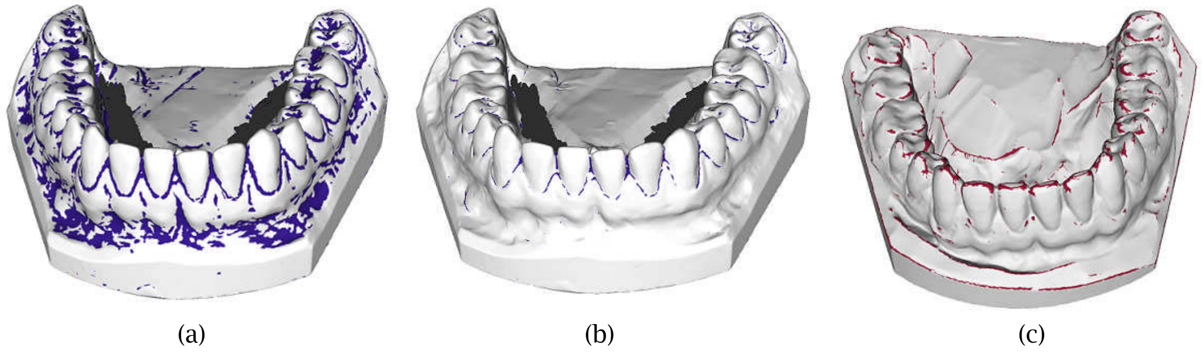


Fig. 2: Color-coded mean curvature values for each vertex $\mathbf{v} \in \mathcal{V}$. Positive curvature values are shown in red while negative curvature values are colored in blue.

In order to deal with piecewise smooth surfaces such as triangle meshes, we have to estimate discrete curvature values. We calculate these discrete curvature values according to the method proposed by Rusinkiewicz [20]. For each vertex $\mathbf{v} \in \mathcal{V}$ the principal curvature values are given by $\kappa_1(\mathbf{v})$ and $\kappa_2(\mathbf{v})$. Based on these values, we compute the mean curvature $\kappa_H(\mathbf{v}) = \frac{\kappa_1(\mathbf{v}) + \kappa_2(\mathbf{v})}{2}$ for each vertex $\mathbf{v} \in \mathcal{V}$. The values $\kappa_H(\mathbf{v})$ are then mapped to the range $[-1,1]$ (see [24] for implementation details). Figure 2 shows the distribution of mean curvature on the mesh.



(a) (b) (c)

Fig. 3: Results for the vertex set $\mathcal{D}$ (blue) according to the threshold $\varepsilon = -0.1$ (a) and $\varepsilon = -0.5$ (b). The color-coded vertices belonging to the vertex set $\mathcal{P}$ (red) with $\eta = 0.8$ are shown in (c).

State of the art segmentation methods mostly define part boundaries according to the minima rule [7]. This rule states that boundaries between parts could be found along lines of negative minima curvature. Figure 2 shows an example of the mean curvature distribution and in figure 3(a) vertices with $\kappa_H \leq -0.1$ are highlighted. These examples show that vertices with negative curvature are located at the transition between tooth and gum, so we could apply the minima rule as well. Unfortunately, there are many noisy regions due to the digitization and the natural shape of the gum. Therefore, we have to reduce the number of considered vertices. At first, a user defined threshold ε is used as an upper bound. Based on ε the feature vertex set

$$\mathcal{D} = \{\mathbf{v} \in \mathcal{V} \mid \kappa_H(\mathbf{v}) \leq \varepsilon\} \tag{0.1}$$

is generated.

Vertices belonging to $\mathcal{D}$ are grouped into one or more separate regions. Two vertices $\mathbf{u}, \mathbf{w} \in \mathcal{D}$ belong to the same region, if they are directly connected by an edge $(\mathbf{u}, \mathbf{w}) \in \mathcal{E}$ or if there exists a path $(\mathbf{u} = \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{k-1}, \mathbf{v}_k = \mathbf{w})$ between $\mathbf{u}$ and $\mathbf{w}$ whose vertices belong to the set $\mathbf{v}_i \in \mathcal{D}$ (with $0 \leq i \leq k$).

The choice of a threshold value ε is complicated. Using the value $\varepsilon \leq -0.5$ results in too few vertices in $\mathcal{D}$ to obtain closed regions at the tooth boundaries (Figure 3(b)). On the other hand, when using a threshold of $\varepsilon = -0.1$ or above, many undesired regions remain (figure 3(a)). Our experiments showed that $\varepsilon = -0.3$ is a good choice. After that, most noisy regions have been filtered, but there are still small feature regions left. Thus regions with less than $0.01 \cdot |\mathcal{D}|$ vertices are deleted. Finally we close small holes between and within the remaining regions, by applying morphological operations [19]. In the following sections, we assume that $\mathcal{D}$ specifies the enhanced vertex set.

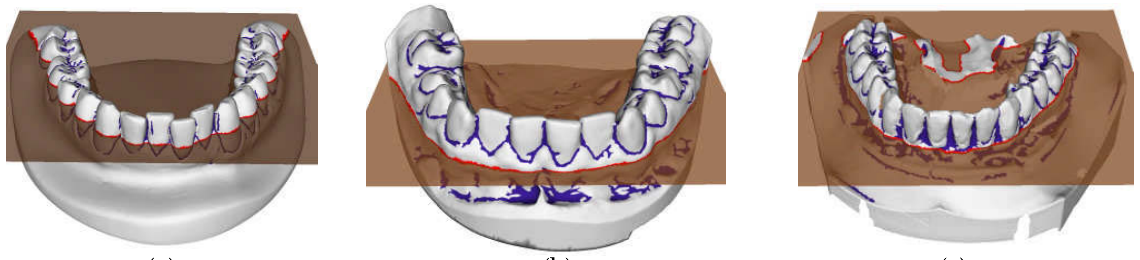
Since these feature regions along with their curvature values are part of the external snake energy, we define a new curvature function for each vertex $\mathbf{v} \in \mathcal{V}$:

$$\kappa(\mathbf{v}) = \begin{cases} \kappa_H(\mathbf{v}) & \text{if } \mathbf{v} \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases} \tag{0.2}$$

The final segmentation result is obtained by starting an active contour at the surface of each individual tooth crown. Vertices belonging to the crown are characterized by strong positive mean curvature. So we construct a new vertex set:

$$\mathcal{P} = \{\mathbf{v} \in \mathcal{V} : \kappa_H(\mathbf{v}) > \eta\} \tag{0.3}$$

where η is a user defined threshold. Experiments showed that a value of $\eta = 0.8$ is sufficient. An example of this set is shown in Figure 3(c). Unfortunately, we cannot use these vertices directly to segment the teeth, because there are many vertices within $\mathcal{P}$ that are located at the surface of the gum and at the base of the dental model. In that way, we will obtain a severe over segmentation, which cannot be reduced without expert knowledge. Therefore, we have to thin out the set $\mathcal{P}$ until only vertices at tooth crowns remain.



(a) (b) (c)

Fig. 4: Best fitting plane resulting from the principal component analysis of the vertex set $\mathcal{D}$.

2.2 Principal Component Analysis

Principal component analysis is a common tool in statistical analysis. The central idea is to reduce the dimensionality of a set of samples, while retaining as much as possible of the variation present in the set. This is achieved by transforming the original set to a new set of variables, which are called components. One could sort the components in a way that the first few of them retain most of the variation present in all of the original variables. Components that are derived in this way are called principal components. The earliest description of this technique was given by Pearson [17] and Hotelling [8]. Pearson [17] developed a method for finding a line or plane that best fit a set of points in p-dimensional space. The term best fit, however, is mathematically imprecise. Consider a set of points in three-dimensional space and their perpendicular distance to a set of planes. The plane which minimizes the sum of distances is called best fitting plane. Pearson [17] also showed that this plane contains the centroid of the point-set.

In our case, we are looking for a first estimate for the boundary between teeth and gum. Vertices belonging to the vertex set $\mathcal{D}$ are located in the vicinity of the searched boundary, as shown above.

Therefore we use principal component analysis to calculate the plane that best fits to all vertices in $\mathcal{D}$. The calculation is based on eigenvalue decomposition of the covariance matrix, which is given by

$$\mathbf{C} = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{v}\in\mathcal{D}}(\mathbf{v}-\bar{\mathbf{m}})\otimes(\mathbf{v}-\bar{\mathbf{m}})^T \tag{0.4}$$

where

$$\bar{\mathbf{m}} = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{v}\in\mathcal{D}}\mathbf{v} \tag{0.5}$$

is the barycentric coordinates of the vertex set $\mathcal{D}$. The square matrix $\mathbf{C}$ is symmetric and real-valued. Thus, the eigenvalue decomposition, calculated by the Jacobi eigenvalue algorithm [18] results in three real-valued eigenvalues. The eigenvector corresponding to the largest (first) eigenvalue points in direction of the maximal variance in the data-set. The plane we search for is spanned by the eigenvectors, corresponding to the first and second eigenvalue and contains the barycenter $\bar{\mathbf{m}}$ (figure 4).

2.3 Snake Model

The active contour model was first proposed by Kass et al. [11] for representing image contours. Their basic snake model is an energy minimizing curve, which evolves under the influence of several forces. The contour is represented as a parametric curve $\mathbf{c}(s) = (x(s), y(s))$ where $s \in [0,1]$ and the energy functional to be minimized is written as

$$E^*_{\text{snake}} = \int_0^1 E_{\text{snake}}(\mathbf{c}(s))\mathrm{d}s = \int_0^1 \Big(E_{\text{int}}(\mathbf{c}(s)) + E_{\text{image}}(\mathbf{c}(s)) + E_{\text{con}}(\mathbf{c}(s))\Big)\mathrm{d}s. \tag{0.6}$$

E_{int} represents the internal energy of the contour, E_{image} refers to the image forces and E_{con} is the external constraint force. The internal energy

$$E_{\text{int}} = \frac{\alpha(s)\left\|\mathbf{c}_s(s)\right\|^2 + \beta(s)\left\|\mathbf{c}_{ss}(s)\right\|^2}{2} \tag{0.7}$$

consists of the first and second-order derivative of the parametric curve, where $\alpha(s)$ and $\beta(s)$ are user defined weights. The first term will have large values at gaps in the curve, while the second term increases when the curve bends rapidly. The final location of the snake corresponds to local minima of the energy functional. The functional optimization is solved using calculus of variation, which involves deriving a variational integral and solving the corresponding Euler-Lagrange partial differential equation iteratively. Therefore image forces and constraint forces need to be differentiable to guarantee convergence. Also the computed contour coordinates are real numbers, allowing the points to fall between the discrete pixel coordinates of the underlying image.

In order to overcome these drawbacks, as well as to speed up computation, Williams and Shah [25] proposed an evolution method based upon the greedy principle. First of all, they use a sampled version of the contour $\mathbf{c} = (\mathbf{c}_0, \ldots, \mathbf{c}_n)$ where each sample $\mathbf{c}_i$ (where $0 \le i \le n$) is related to a discrete pixel coordinate. During an iteration, the energy function is computed at $\mathbf{c}_i$ and each of its eight neighbors. Therefore, the energy functional to be minimized is written as:

$$E^*_{snake} = \sum_{i=0}^{n} E_{snake}(\mathbf{c}_i) = \sum_{i=0}^{n}\Big(E_{int}(\mathbf{c}_i) + E_{image}(\mathbf{c}_i) + E_{con}(\mathbf{c}_i)\Big) \tag{0.8}$$

and the internal energy is approximated using finite differences.

Milroy et al. [14] extend the snake evolution to three-dimensional wireframe models. Their evolution method is based on the greedy principle as well. During each iteration each vertex of the snake is moved to one of two nearby positions on the surface, perpendicular to the snake contour. The distance of these points is specified by the user. Therefore, the region surrounding each vertex is approximated by a second-order surface.

Jung et al. [9] present a simpler evolution scheme to ensure that the snake does not leave the surface. The snake elements are always located at vertices of the mesh. In each iteration, the snake elements move to one of their neighboring vertices.

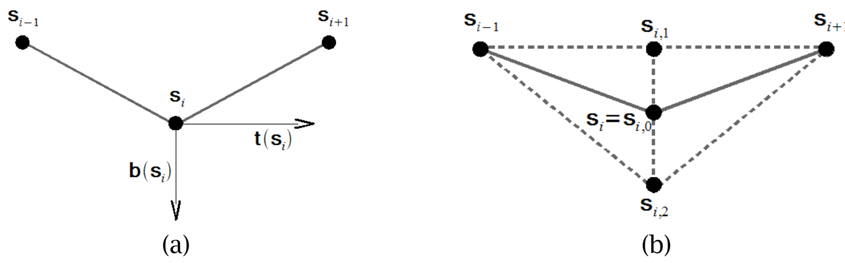


Fig. 5: For each snaxel $\mathbf{s}_i$ the unit normal vector and unit tangent vector are approximated (a). At each iteration step, two adjacent vertices are selected (b).

The active contour model used in this article consists of an ordered set of vertices $\mathcal{S} = (\mathbf{s}_1,\dots,\mathbf{s}_n)$ with $\mathcal{S} \subset \mathcal{V}$. Furthermore, successive snake vertices $(\mathbf{s}_i,\mathbf{s}_{i+1}), i = 0,\dots,n-1$ are restricted to lie on adjacent vertices $\mathbf{s}_{i+1} \in \mathcal{N}(\mathbf{s}_i)$. We will call snake vertices “snake elements” or for short “snaxel”. In this paper, we only consider simple and closed active contours ($\mathbf{s}_0 = \mathbf{s}_n$ whereas $\mathbf{s}_i \neq \mathbf{s}_j$ for $i \neq j : i,j = 0,\dots,n-1$). On that note all subscript arithmetic is modulo n. Vertices which are enclosed by the snake are called interior vertices. At each snaxel $\mathbf{s}_i$ the unit tangent vector

$$\mathbf{t}(\mathbf{s}_i) = \frac{\mathbf{s}_{i+1} - \mathbf{s}_{i-1}}{\left\|\mathbf{s}_{i+1} - \mathbf{s}_{i-1}\right\|} \tag{0.9}$$

is approximated, as shown in figure 5(a). The snake will now be oriented such that the unit snaxel normal vector $\mathbf{b}(\mathbf{s})$ at each snake element $\mathbf{s} \in \mathcal{S}$ points away from the interior vertices of the contour. The unit snaxel normal is defined as the outer product between the unit tangent vector and the unit surface normal vector:

$$\mathbf{b}(\mathbf{s}) = \mathbf{t}(\mathbf{s}) \times \mathbf{n}(\mathbf{s}) \quad \forall\ \mathbf{s} \in \mathcal{S}. \tag{0.10}$$

In each iteration, a snaxel $\mathbf{s}_i \in \mathcal{S}$ can only move to a vertex within its neighborhood $\mathcal{N}(\mathbf{s}_i)$. We further restrict the possible candidates in the same way as proposed by Milroy et al [14]. Each snaxel is moved to one of two adjacent vertices $\mathbf{s}_{i,1},\mathbf{s}_{i,2} \in \mathcal{N}(\mathbf{s}_i)$, perpendicular to the snake curve, or remains at its current position $\mathbf{s}_i = \mathbf{s}_{i,0}$. Figure 5(b) shows an example for the choice of candidates. The energy of the snake element $\mathbf{s}_i \in \mathcal{S}$ is defined as:

$$\mathbf{e}(\mathbf{s}_i) = \mathbf{e}_{\text{int}}(\mathbf{s}_i) + \mathbf{e}_{\text{ext}}(\mathbf{s}_i) \tag{0.11}$$

where $\mathbf{e}_{\text{int}}$ is the internal energy and $\mathbf{e}_{\text{ext}}$ is the external energy. Note that $\mathbf{e},\mathbf{e}_{\text{int}},\mathbf{e}_{\text{ext}} \in \mathbb{R}^3$ are column vectors $\mathbf{e}(\mathbf{s}_i) = (e_0(\mathbf{s}_i), e_1(\mathbf{s}_i), e_2(\mathbf{s}_i))^T$. The value of the topmost element of these vectors corresponds to the contour energy at the snake element $\mathbf{s}_i \in \mathcal{S}$. During an iteration, each element is moved to the vertex corresponding to the minimal value of $\mathbf{e}(\mathbf{s}_i)$. The energy functional to be minimized is written as:

$$E^*_{\text{snake}}(\mathcal{S}) = \sum_{i=0}^{n} \min_{0\le j\le 2} \{e_j(\mathbf{s}_i)\} \tag{0.12}$$

and the evolution is carried out according to greedy principle [25].

The internal energy generally consists of the first and second order derivative of the curve, which are also called tension and bending energy, respectively. Here, we approximate the derivatives by finite differences. The internal energy term for each element $e_{\mathrm{int},j}(\mathbf{s}_i)$ of the vector $\mathbf{e}_{\mathrm{int}}(\mathbf{s}_i)$ is therefore defined as follows:

$$e_{\mathrm{int},j}(\mathbf{s}_i) = \alpha \left\| \mathbf{s}_{i,j} - \mathbf{s}_{i-1} \right\| + \beta \left\| \mathbf{s}_{i+1} - 2\mathbf{s}_{i,j} + \mathbf{s}_{i-1} \right\| \tag{0.13}$$

where α and β are user defined parameters. Large values of α shrink the snake, while large values of β reduce the curvature of the snake.

The external energy $\mathbf{e}_{\mathrm{ext}}$ can take various forms; in general, it is derived from information of the surface curvature. At an explored feature vertex the value of the external snake energy will be a local minimum. Here, the vertex set $\mathcal{D}$ contains the features to be detected and the external energy is derived from the curvature function κ, defined in equation (2.2). Unfortunately, we cannot use this curvature function directly, since their value is only nonzero at feature vertices. This means, that the external energy would be constant on big parts of the mesh, and so the evolution of the snake depends completely on the internal energy. As one can see in figure 3, the relative position between the cutting contour obtained by the principle component analysis and the transition between teeth and gum, changes completely from denture to denture. In figure 3(a) the contour lies completely above the searched feature, while in figure 3(b) the contour lies completely below. For this purpose, the external energy

$$\mathbf{e}_{\mathrm{ext}}(\mathbf{s}_i) = \mathbf{e}_{\mathrm{feat}}(\mathbf{s}_i) + \mathbf{e}_{\mathrm{press}}(\mathbf{s}_i) \tag{0.14}$$

is composed of two newly defined energy functionals, the *feature attraction energy* $\mathbf{e}_{\mathrm{feat}}$ and the *pressure energy* $\mathbf{e}_{\mathrm{press}}$. The *feature attraction energy*, which will be explained in section 2.4, expands the scope of feature regions by using a pre-calculated vector field, the *feature attraction flow*.

The second energy term, the *pressure energy* $\mathbf{e}_{\mathrm{press}}$, is similar to the balloon force proposed by Cohen [2]. His aim was to modify the external forces in such a way as to obtain more stable results for image snakes. In order to archive this goal, he introduced an inflation force by which the snake behaves like a balloon. This force is written as $E_{\mathrm{ball}} = k \cdot \mathbf{n}(s)$ where $\mathbf{n}(s)$ is the unit outward normal of the curve with arc-length s and k determines the magnitude of the force. This avoids that the curve is trapped by spurious noisy feature points and makes the result much more insensitive to the initial position.

In our case, the pressure force is used as a tool to accelerate the movement of the snake and to reduce the influence of swirling effects in the *feature attraction flow*. It also affords the opportunity to control the evolution directly, for example when the snake is used as a semiautomatic segmentation tool and the initialization is done by the user. The energy term for each element $e_{\mathrm{press},j}$, in the vector $\mathbf{e}_{\mathrm{press}}$ is written as the inner product:

$$e_{\mathrm{press},j}(\mathbf{s}_i) = \delta \left\langle \mathbf{b}(\mathbf{s}_i), \frac{\mathbf{s}_{i,j} - \mathbf{s}_i}{\left\| \mathbf{s}_{i,j} - \mathbf{s}_i \right\|} \right\rangle \tag{0.15}$$

where $\mathbf{b}(\mathbf{s}_i)$ is the outward unit normal of $\mathcal{S}$ at the snake element $\mathbf{s}_i$ and $\mathbf{s}_{i,j}$ is its candidate vertex. The weight δ determines the strength of this force, whereas the sign of δ specifies the direction in which the force acts. Negative values let the snake grow, while a positive value shrinks the snake. In general, the magnitude of δ should be small, otherwise the snake may not converge.

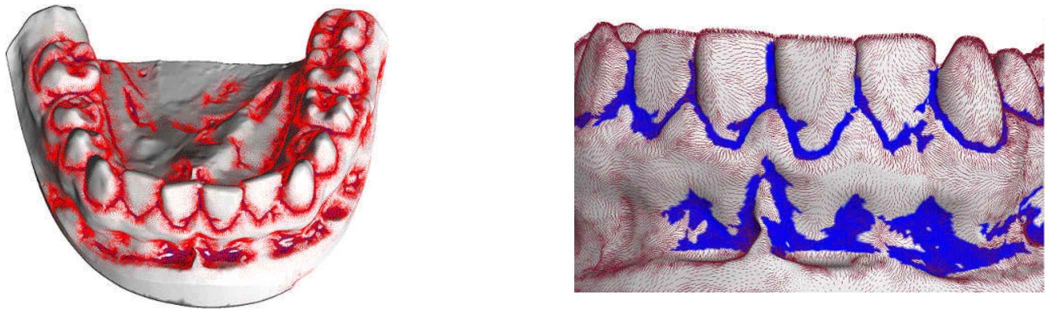


Fig. 6: Examples of the derived *feature attraction flow*.

2.4 Feature Attraction Energy

As mentioned above, the relative position between the cutting contour and the transition between teeth and gum could not be determined in the first place. Therefore, especially for automatic segmentation, we have to expand the capture range of features to attract the snake from an arbitrary position. This problem was first addressed by Xu and Prince [26, 27]. They proposed the *gradient vector flow* as a new external force for parametric image snakes. In order to compute such a field, one has to define a scalar field as an overlay of the image. After that, the gradient of this scalar field is calculated. The final vector field is obtained by diffusion of the gradient information over the entire image. One can interpret this field as the direction to be followed to reach the nearest object boundaries.

In our case, the computation is based on the region included in the set $\mathcal{D}$ and the above defined curvature function $\kappa(\mathbf{v})$. The gradient of a scalar field is defined as a vector field whose components are the partial derivatives of the underlying scalar field. Here we deal with a discrete and mostly irregular grid, so we have to approximate the gradient vectors. Therefore, the gradient is determined using finite differences:

$$\nabla(\mathbf{v}) = \frac{1}{\left|\mathcal{N}(\mathbf{v})\right|} \sum_{\mathbf{u}\in\mathcal{N}(\mathbf{v})} (\kappa(\mathbf{v}) - \kappa(\mathbf{u})) \cdot \frac{\mathbf{u}-\mathbf{v}}{\left\|\mathbf{u}-\mathbf{v}\right\|}. \tag{0.16}$$

Three general properties of the gradient vectors are essential for the following calculations. First of all, the vector $\nabla(\mathbf{v})$ at each vertex $\mathbf{v}\in\mathcal{V}$ points in the direction of the nearest feature vertex. Second, these vectors generally have large magnitudes only in the immediate vicinity of a feature region. Third, in regions without features $\left\|\nabla(\mathbf{v})\right\|$ is nearly zero.

We define the *feature attraction flow* to be the vector field $\mathbf{f}(\mathbf{v})$, where for each $\mathbf{v}\in\mathcal{V}$ the vector of this field points towards the nearest feature region. The *feature attraction flow* is initialized with the derived curvature gradient $\mathbf{f}_0(\mathbf{v}) = \nabla(\mathbf{v})$. After that, an iterative diffusion process is performed, which distributes the gradient information over the entire mesh:

$$\mathbf{f}_t(\mathbf{v}) = (1-\lambda)\left(\frac{1}{\left|\mathcal{N}(\mathbf{v})\right|} \sum_{\mathbf{u}\in\mathcal{N}(\mathbf{v})} \mathbf{f}_{t-1}(\mathbf{u})\right) + \lambda\cdot\nabla(\mathbf{v}), \quad \forall\mathbf{v}\in\mathcal{V}. \tag{0.17}$$

After t iterations we obtain a smooth vector field $\mathbf{f}_t$, which has non-zero magnitude at each vertex of the mesh. Figure 6 shows an example of the *feature attraction flow*. In particular, at vertices $\mathbf{v}\in\mathcal{V}$ with large magnitude $\left\|\nabla(\mathbf{v})\right\|$ the second term of equation (2.14) dominates the sum, which keeps the vector $\mathbf{f}_t(\mathbf{v})$ nearly equal to the initial gradient vector. The first term of equation (2.14) dominates at vertices $\mathbf{v}\in\mathcal{V}$ where $\left\|\nabla(\mathbf{v})\right\|$ is nearly zero, yielding a slowly varying field. The parameter λ is a regularization parameter controlling the tradeoff between the first term and the second term in the sum. With small values, one can reduce the influence of noise. After each iteration we perform an immediate step where the length of each vector is clamped to the range $[0,1]$. This prevents the magnitude of vectors near or within feature regions from rising rapidly.

The *feature attraction energy* is then defined by

$$e_{\text{fae},j}(\mathbf{s}_i) = \delta\left(-\left\|\mathbf{f}(\mathbf{s}_{i,j})\right\| - \left\langle \frac{\mathbf{f}(\mathbf{s}_{i,j})}{\left\|\mathbf{f}(\mathbf{s}_{i,j})\right\|}, \frac{\mathbf{s}_{i,j}-\mathbf{s}_i}{\left\|\mathbf{s}_{i,j}-\mathbf{s}_i\right\|} \right\rangle\right) \tag{0.18}$$

where $\mathbf{f}(\mathbf{v})$ is the *feature attraction flow* after t iterations. The minuend ensures that the snake moves in direction of increasing *feature attraction flow* vectors. Due to the minimization of the subtrahend the snake is moving in direction of the nearest feature region and converges there.

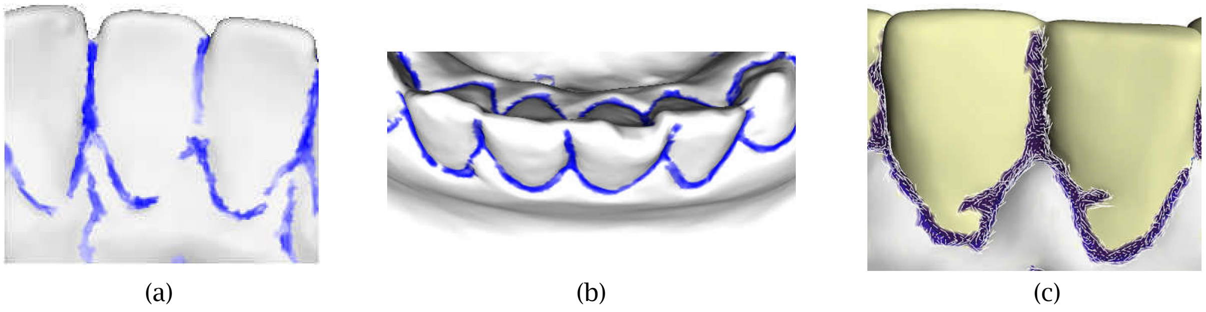
2.5 Edge Tangent Flow

In digitized dental casts, often interstices between adjacent teeth vanish due to severe malocclusion or low resolution scanners. In cases such as shown in figure 7(a-b), automatic segmentation may fail. Therefore, we have to trace and reconnect the interstices, prior to the final segmentation. Kang et al. [10] first solved this problem for two-dimensional images by using *edge tangents*. Their aim was the development of an enhanced edge detector to produce high-quality line drawings. In general, the task of creating line drawings is difficult, since salient edges often appear as piecewise unconnected lines in natural photographs. Only in an overall view one can decide whether there is a salient edge or not. In order to maximize the line coherence, they construct a smooth vector field with which one is able to follow the shape of lines. For this purpose, they define the *edge tangents*, which are perpendicular to the image gradient.

In this paper, we extend the definition of the *edge tangent flow* to three-dimensional triangle meshes. We define the *edge tangent vector*, denoted $\mathbf{t}_{\text{ETF}}(\mathbf{v})$ for each vertex $\mathbf{v} \in \mathcal{V}$, as the outer product between curvature gradient $\nabla(\mathbf{v})$ and surface normal $\mathbf{n}(\mathbf{v})$:

$$\mathbf{t}_{\text{ETF}}(\mathbf{v}) = \nabla(\mathbf{v}) \times \mathbf{n}(\mathbf{v}) \tag{0.19}$$

An example is shown in figure 7(c).



(a) (b) (c)

Fig. 7: Unconnected feature regions due to low resolution scanners (a) and severe malocclusion (b). An example of the *edge tangent flow* is shown in (c).

3 TOOTH SEGMENTATION

An overview of the proposed segmentation algorithm is shown in figure 1. The algorithm starts with estimating the mean curvature for each vertex $\mathbf{v} \in \mathcal{V}$. Now, the user defines the threshold ε to obtain the vertex set $\mathcal{D}$, as well as the threshold η to obtain the vertex set $\mathcal{P}$, according to the method described in section 2.1.

The vertices belonging to the set $\mathcal{D}$ are located at the transition between the teeth and the gum. In order to obtain an initial estimate of this region, we perform the principal component analysis on the vertices of $\mathcal{D}$. As stated in section 2.2, the best fitting plane is defined by the vectors belonging to the two larger eigenvalues. By cutting the dental cast with that plane, we obtain we obtain one or more closed and oriented contours (figure 3). Each of them is transformed into a snake and evolves independently.

After each snake element is moved, in order to minimize the energy functional, it may happen that adjacent snake elements are no longer located on adjacent vertices. In those cases we have to reconnect them. During evolution, a snake may shrink until it encloses only a single triangle. These snakes will be discarded. On the other hand, two snakes can expand and collide with each other, and then they are merged. At the end, the remaining snake encloses the teeth completely and is called initial snake.

Now, we use the *edge tangent flow* to detect and reconnect the interstices. For each snake element, the neighboring vertices are examined. If the *edge tangent flow* vectors at these vertices are orthogonal to the tangent vector of the current snaxel, then it is most likely that there is an interstice. First, the corresponding snaxel at the other end of the interstice is estimated. After that the shortest path in direction parallel to the *edge tangent flow* vectors and within the interior of the initial snake to the corresponding snaxel at the other end of the interstice is searched. If the feature region along the searched path is unconnected, then these gaps will be closed as long as they are smaller then five percent of the Euclidean distance between these corresponding snaxel.

The remaining task is to detect the surface of each individual tooth exactly. Unfortunately, we cannot ensure that all snake elements of the initial snake are located at the transition between teeth and gum. As a consequence, some interstices have not been detected yet. But besides that, we have reduced the considered surface of the mesh significantly. The remaining parts are the surface of the teeth and maybe some small parts of the gum. Therefore, we can be sure that all regions included in the vertex set $\mathcal{P}$ and at the interior of the initial snake, are cusps of teeth. Each of these regions will be used to initialize an individual snake. Under the influence of the *feature attraction flow* and the *pressure force*, each snake expands until it reaches the tooth boundary. In order to avoid expansion, when tooth boundaries are missing, we introduce the hard constraint that the snake has to stop when it reaches the initial snake or detected interstices.

4 EXPERIMENTAL RESULTS

The proposed method was tested on ten different dental casts and performs constantly well. The resulting segmentation is shown in figure 10 and the overall runtime for these meshes is listed in table 1. The internal energy was weighted by setting $\alpha = \beta = 0.6$ throughout the tests. The experimental results showed that weighting the *feature attraction energy* with $\gamma = 0.5$ is sufficient. Since the *pressure energy* accelerates the movement of the snake, we set $\delta = 0.2$ for the initial snake and $\delta = -0.2$ to weight the snakes started within the teeth.

Model (number of vertices)	*Model 1 (142k)*	*Model 2 (208k)*	*Model 3 (155k)*	*Model 4 (209k)*
Curvature estimation	1.34 s	1.9 s	1.4 s	1.85 s
PCA and snake initialization	1.15 s	1.7 s	1.3 s	1.7 s
Feature attraction flow	3.2 s	4.86 s	3.6 s	4.76 s
Contour calculation	0.22 s	0.3 s	0.24 s	0.2 s
Outer snake optimization	4.4 s	3.3 s	3.5 s	5.2 s
Single tooth segmentation	0.09 s	0.1 s	0.14 s	0.09 s
Total time	12 s	14 s	13 s	15 s

Tab. 1: Runtime statistic for all models illustrated in Figure 11.

Nevertheless in dental casts, where the boundary between tooth and gum is very smooth, the segmentation fails. The same error occurred, when neighboring teeth overlap each other due to severe malocclusion. In these cases, the user needs to segment the tooth manually.

5 CONCLUSION

We have presented a highly automated segmentation method for separation of teeth from triangular meshes. The algorithm is based on active contours, wherefore it is robust even in the presence of heavy noise. Through our experiments, we found that our approach is fast and requires minimal user interaction. Hence it enables orthodontics to plan dental treatments much faster.

The proposed snake model is not restricted to operate on this special type of three-dimensional models. Therefore, as future work, we will investigate their behavior for the segmentation of several other models, like digitized archeological models.

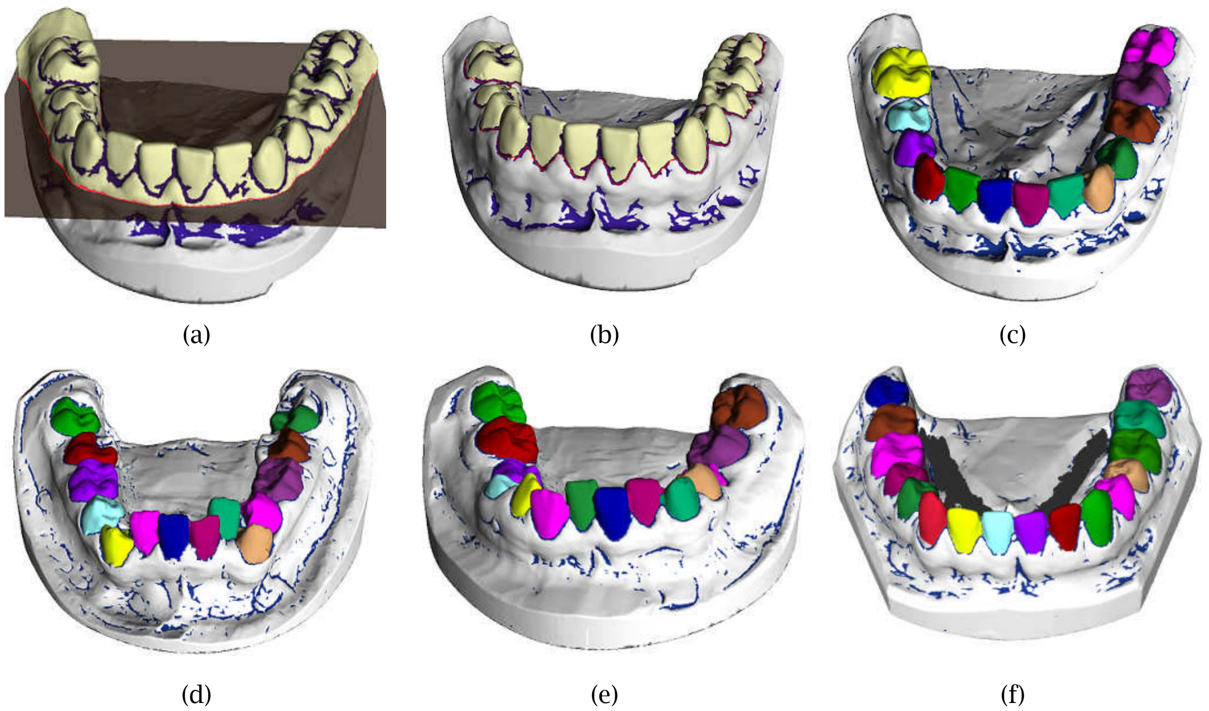


Fig. 10: The dental model is cut by the best fitting plane (a). Vertices belonging to the cut triangles are transformed into the initial snake, which is then optimized (b). The resulting segmentation for model 1 is shown in (c). The segmentation results for model 2-4 are shown in (d)-(f).

ACKNOWLEDGEMENTS

This work was partially supported by the Sächsische Aufbaubank and financed from budget of the European Union and the Free State of Saxony. We like to thank Image Instruments for providing the dental models.

REFERENCES

[1] Cadent, http://www.orthocad.com, OrthoCAD.
[2] Cohen, Laurent D.: On active contour models and balloons, Computer Vision, Graphics and Image Processing: Image Understanding, 53(2), 1991, 211-218.
[3] Dimennex, http://www.dimennex.com, Digital Lab.
[4] Geodigm, http://www.geodigmcorp.com, Emodel Software.
[5] Goshtasby, A.; Nambala, S.; Derijk, W.; Campbell, S.: A system for digital reconstruction of gypsum dental casts, IEEE Trans on Medical Imaging, 16(5), 1997, 664-667.

[6] Hirogaki, Y.; Sohmura, T.; Satoh, H.; Takahashi, J.; Takada, K.: Complete 3-D reconstruction of dental cast shape using perceptual grouping, IEEE Trans. on Medical Imaging, 20(10), 2001, 1093-1101.
[7] Hoffman, D. D.; Singh, M.: Salience of visual parts, Cognition, 63(1), 1997, 29–78.
[8] Hotelling, H.: Analysis of a complex of statistical variables into principal components, Journal of Educational Psychology, 24(6), 1933, 417-441.
[9] Jung, M.; Kim, H.: Snaking across 3d meshes, Proc. Of the 12th Pacific Conf. on Comp. Graphics and Applications, 2004, 87-93.
[10] Kang, H.; Lee, S.; Chui, C.: Coherent line drawing, Proc. ACM Symp. On Nonphotorealistic Animation and Rendering, 2007, 43-50.
[11] Kass, M.; Witkin, A.; Terzopoulos, D.: Snakes: Active contour models, Int. Journal of Comp. Vision, 1, 1988, 321-331.
[12] Kondo, T.; Ong, S. H.; Foong, K. W. C.: Tooth segmentation of dental study models using range images, IEEE Trans. on Medical Imaging, 23(3), 2004, 350-362.
[13] Li, C.; Wang, G.; Xu, T.; Liu, Y.: Orthodontic Simulation and Diagnosis: An Enhanced Tool for Dentists, 27th Annual Int. Conf. of the Engineering in Medicine and Biology Society, 2005, 4345-4348.
[14] Milroy, M. J.; Bradley, C.; Vickers, G. W.: Segmentation of a wrap-around model using an active contour, Computer-Aided Design, 29(4), 1997, 299-320.
[15] Motohashi, N.; Kuroda, T.: A 3D computer-aided design system applied to diagnosis and treatment planning in orthodontics and orthognathic surgery, European Journal of Orthodontics, 21, 1999, 263-274.
[16] Page, D.; Koschan, A.; Abidi, M.: Perception-based 3D triangle mesh segmentation using fast marching watersheds, Proc. IEEE Conf. on Comp. Vision and Pattern Recognition, 2003, 27-32.
[17] Pearson, K.: On lines and planes of closest fit to systems of points in space, Philosophical Magazine, 6(2), 1901, 559-572.
[18] Press, W. H.: Numerical recipes: the art of scientific computing, Cambridge University Press, 2007.
[19] Rossl, C.; Kobbelt, L.; Seidel, H.-P.: Extraction of feature lines on triangulated surfaces using morphological operators, Proc. of the AAAI Symp. on Smart Graphics, 2000, 71-75.
[20] Rusinkiewicz, S.: Estimating Curvatures and Their Derivatives on Triangle Meshes, Symp.on 3D Data Processing, Visualization and Transmission, 2004.
[21] Shamir, Ariel: A survey on mesh segmentation techniques, Comp. Graphics Forum, 27(6), 2008, 1539-1556
[22] Sinthanayothin, C.; Tharanont, W.: Orthodontics treatment simulation by teeth segmentation and setup, 5th Int. Conf. on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 1, 2008, 81-84.
[23] Tian-ran, Y.; Ning, D.; Guo-dong, H.; Xiao-sheng, C.; Hai-hua, C.; Wen-he, L.; Quing, Y.; Peijun, L.: Bio-Information Based Segmentation of 3D Dental Models, The 2nd Int. Conf. on Bioinformatics and Biomedical Engineering, 2008, 624-627.
[24] Vergne, R.; Berla, P.; Granier, X.; Schlick, C.: Apparent relief: a shape descriptor for stylized shading, Proc. of the 6th Int. Symp. on Non-photorealistic Animation and Rendering, 2008, 23-29.
[25] Williams, D. J.; Shah, M.: A fast algorithm for active contours and curvature estimation, Comp. Vis., Graphics and Image Proc.: Image Understanding, 55(1), 1992, 14-26.
[26] Xu, C.; Prince, J. L. : Gradient Vector Flow: A New External Force for Snakes, IEEE Proc. Conf. on Comp. Vision and Pattern Recognition, 1997, 66-71.
[27] Xu, C.; Prince, J. L.: Snakes, shapes, and gradient vector flow, IEEE Transaction on Image Processing, 7(3), 1998, 359-369.
[28] Zhao, M.; Ma, L.; Tan, W.; Nie, D.: Interactive Tooth Segmentation of Dental Models, 27th Annual Conf. on Engineering in Medicine and Biology, 2005, 654-657.