

# Building Anatomically Realistic Jaw Kinematics Model from Data

Wenwu Yang, Nathan Marshak, Daniel Sýkora, Srikumar Ramalingam, and Ladislav Kavan

**Abstract**—Recent work on anatomical face modeling focuses mainly on facial muscles and their activation which generate facial expressions. In this paper, we consider a different aspect of anatomical face modeling: kinematic modeling of the jaw, i.e., the Temporo-Mandibular Joint (TMJ). Previous work often relies on simple models of jaw kinematics, even though the actual physiological behavior of the TMJ is quite complex, allowing not only for mouth opening, but also for some amount of sideways (lateral) and front-to-back (protrusion) motions. Fortunately, the TMJ is the only joint whose kinematics can be accurately measured with optical methods, because the bones of the lower and upper jaw are rigidly connected to the lower and upper teeth. We construct a person-specific jaw kinematic model by asking an actor to exercise the entire range of motion of the jaw while keeping the lips open so that the teeth are at least partially visible. This performance is recorded with three calibrated cameras. We obtain highly accurate 3D models of the teeth with a standard dental scanner and use these models to reconstruct the rigid body trajectories of the teeth from the videos (markerless tracking). The relative rigid transformations samples between the lower and upper teeth are mapped to the Lie algebra of rigid body motions in order to linearize the rotational motion. Our main contribution is to fit these samples with a three-dimensional nonlinear model parameterizing the entire range of motion of the TMJ. We show that standard Principal Component Analysis (PCA) fails to capture the nonlinear trajectories of the moving mandible. However, we found these nonlinearities can be captured with a special modification of autoencoder neural networks known as Nonlinear PCA. By mapping back to the Lie group of rigid transformations, we obtain parameterization of the jaw kinematics which provides an intuitive interface allowing the animators to explore realistic jaw motions in a user-friendly way.

**Index Terms**—Motion capture, motion processing, jaw kinematics, face animation.



## 1 INTRODUCTION

Anatomical modeling of the face has been explored in the pioneering work of [1], [2], but recent years witnessed a resurgence of interest in anatomically-based facial animation [3], [4], [5], [6]. Naturally, the primary focus is accurate modeling of facial muscles and their ability to generate facial expressions. However, the shape and expressions of the face are significantly affected by two major bones: the skull and the mandible (lower jaw). This is evidenced by people who underwent jaw surgery, which is a relatively frequent surgery to correct congenital malformations such as the overbite. The face after the surgical treatment looks quite different and often much better than before the surgery. We argue that realistic anatomically-based facial animation needs to start with an accurate jaw kinematics model. With physics-based simulation of facial soft tissues, the relative rigid transformation between the skull and the mandible has a significant effect on the result, because the bones are used as Dirichlet boundary conditions. Even though, strictly speaking, the bones and their attachment to the teeth is elastic, in normal physiological motions these deformations

are negligible and we can safely assume that hard tissues behave as rigid bodies. However, the rigid motion of the mandible relative to the skull is not arbitrary, but is constrained by the anatomy of the Temporo-Mandibular Joint (TMJ). The TMJ is a very complicated joint and enables functions such as chewing of food or talking. Due to its complicated anatomy, the TMJ is also prone to pathologies which are a common concern in medicine [7].

In this paper we focus on accurate modeling of the kinematics of the TMJ for the purposes of computer animation. An ideal interface for animation (known as a “rig”) should be user friendly. The most prominent mode of motion is opening of the mouth. Even this common, everyday motion is, kinematically, a non-trivial composition of rotation and translation (sliding). This sliding occurs on a curve which reflects the geometry of the mandibular condyle and the zygomatic process which are held in close proximity by connective tissues (Fig. 1 for anatomy of the TMJ). Additionally, the jaw also allows for some amount of sideways and front-back translation, even without opening the mouth. Normally, when the mouth is closed, the upper teeth rest naturally in front of the lower teeth. We invite the reader to try translating their lower teeth forward – they can be moved *in front of* the upper teeth. Similarly, it is also possible to move the lower teeth from left to right. All of these motions are combined together to endow the jaw with its basic functions, such as chewing or talking. Our goal is to provide an intuitive animation interface allowing the users to explore realistic jaw motions. In particular, the users can synthesize realistic jaw motions by just varying three parameters that correspond to anatomically prominent

- W. Yang is with the School of Computer and Information Engineering, Zhejiang Gongshang University, China, Hangzhou 310012. This work was done when he is a visiting scholar at the University of Utah. Email: wwyang@zjgsu.edu.cn
- N. Nathan, S. Ramalingam and K. Ladislav are with the University of Utah. Email: {srikumar.ramalingam, ladislav.kavan}@gmail.com
- D. Sýkora is with the Czech Technical University in Prague, Faculty of Electrical Engineering. Email: sykora@fel.cvut.cz

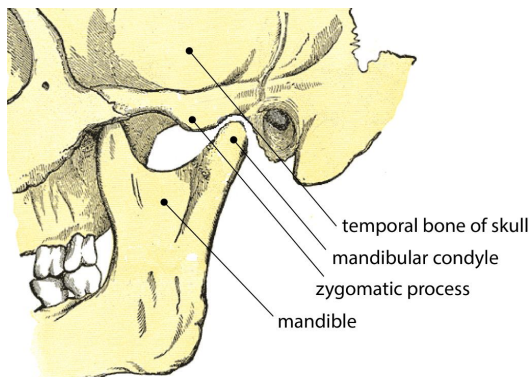


Fig. 1. The temporomandibular joint is the joint between the mandible and the temporal bone of the skull.

modes of motion: opening, sideways sliding (known as “lateral excursion” in medical terminology), and front-back translation (known as “protrusion” and “retrusion”). In addition to synthesizing anatomically accurate jaw motions using an intuitive interface, our modeling will also allow us to validate if a given jaw motion is anatomically accurate or not.

Previous kinematic models for the jaw were qualitative, designed by researchers who observed the relevant anatomy and proposed models for its geometric behavior. We propose a different, data-driven approach, taking advantage of the fortuitous fact that the kinematics of the jaw can be measured with high accuracy using optical methods. This is because the skull and the mandible bones are connected with the upper and lower teeth, and the motion of the teeth is directly visible if the lips are open. To obtain data to train our model, we have asked an actor to exercise the entire range of motion of the jaw while keeping the lips at least partially open. This performance is recorded by multiple (we use three) cameras which have been calibrated (both intrinsically and extrinsically, and also synchronized in time). We also create a highly accurate 3D models of the teeth of the actor using a standard dental 3D scanner, see Fig. 2. We use the 3D models of the upper and lower teeth for tracking the video, reconstructing the 3D position and orientation of the upper and lower teeth in each frame. Computing the relative rigid motion between the upper and lower teeth poses in each frame gives us point samples of physiologically possible jaw motions. These point samples are rigid transformations, i.e., points on the Lie group of rigid motions, often denoted as  $SE(3)$ . The  $SE(3)$  is a non-linear manifold due to the non-linearity of rotations. To simplify our task of creating an intuitive parameterization of jaw kinematics, we map our point samples from  $SE(3)$  to the corresponding Lie algebra  $se(3)$  via the logarithmic mapping. The Lie algebra  $se(3)$  is a standard linear (vector) space and therefore permits standard Principal Component Analysis (PCA). However, performing the PCA on our point samples in  $se(3)$  fails to capture the nonlinearity of trajectories of motions such as mouth opening, where the mandibular condyle slides along the zygomatic process along a curved path (see Fig. 5).

In order to extract anatomically meaningful modes of

motion from our input data ( $se(3)$  samples), we turned to autoencoder neural networks. In particular, we applied a special type of autoencoder termed Nonlinear PCA (NLPCA) [8]. A modified version of NLPCA allowed us to explain all of our input data with a generative model with only three parameters, corresponding to the main modes of jaw motion: 1) mouth opening, 2) lateral excursion, 3) protrusion and retrusion. Furthermore, we use our data to obtain explicit boundaries on each of these three modes of motion, with the bounds on lateral excursion and pro/re-trusion depending on the amount of mouth opening. The result is an intuitive and anatomically-realistic 3-parameterization for jaw kinematics. There are two potential applications of our resulting jaw kinematics model: (1) a control interface allowing animators to intuitively synthesize meaningful jaw motions, e.g., for special effects animation [6]; (2) allow computer vision researchers to automatically discard invalid jaw poses while tracking recorded facial performances [9].

## 2 RELATED WORK

The need for jaw kinematic modeling was identified in previous work in computer graphics. Sifakis et al. [2] created a jaw kinematics rig by designing sliding tracks of the condyles identified from magnetic resonance images. Because MRI is not always available and may be even medically contraindicated, [9] proposed a geometric rig offering two rotational degrees of freedom along a common pivot point and one translational degree of freedom along a fixed axis. This model was slightly generalized by [6] who proposed using three translational degrees of freedom instead of just one, in addition to the two original rotational degrees of freedom. Li et al. [10] use an articulated model including the neck and eyeballs, with three rotational degrees of freedom for each of the joints, including the jaw. Our approach does not require MRI but still produces highly accurate data-driven jaw kinematics model for a given actor.

The mechanical function of the jaw has been studied in biomechanics, often using full six degrees of freedom for the rigid body motion of the jaw relative to the skull [11], even though reduced models were also considered, e.g., [12] who proposed a planar constraint along which the condyles can slide, resulting in four degrees of freedom. A simulation platform can be used to create computational models using variables for modeling gravity, external forces, and jaw muscle activity [13]. These models were shown to be capable of predicting jaw movements for mundane, but complex actions like chewing [14] or post-reconstruction surgery [13].

An established tool to study bone kinematics is fluoroscopy (X-ray videos). Fluoroscopic studies of the temporomandibular joint kinematics have been carried out on rabbits [15], but the use of ionizing radiation (X-ray) for research purposes on humans is not acceptable. Fortunately, the motion of the mandible relative to the skull can be captured using optical methods if the lips are at least partially open. Tracking in videos is a well studied computer vision problem [16], and popular methods include template-based tracking such as Lucas-Kanade [17], active appearance models [18], feature-based tracking [19], and edge or

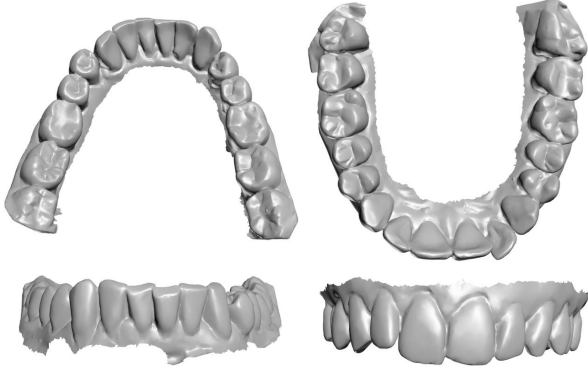


Fig. 2. 3D models (meshes) of the lower and upper teeth.

boundary-based tracking [20], [21], [22], [23]. A key challenge in teeth tracking and pose estimation problem is that it violates many common assumptions that are commonly satisfied in a tracking framework. In particular, teeth are usually textureless and often partially occluded. Tracking methods such as Lucas-Kanade [17] and active appearance models [18] require the object to remain free from occlusion during the tracking process and undergo only relatively small appearance changes with respect to the original template. Unfortunately, in practice teeth are usually highly occluded and their appearance changes considerably due to glossy nature of enamel. More robust keypoint-based methods such as SIFT [19] are also hardly applicable since teeth are usually smooth and self-similar therefore it is difficult to find sufficient number of distinct feature points that can be tracked consistently.

There are a few tracking algorithms that are customized for teeth [23], [24], [25], and these tracking methods are primarily developed for augmented reality applications, where the goal is to overlay an AR image on the patient for providing additional assistance during dental surgery. In [23], a stereo camera is used to capture and reconstruct the 3D contour of a patient's teeth. This 3D contour is registered with the 3D model obtained using CT scans using Iterative Closest point (ICP) algorithm. Using this registration, we can have augmented reality (AR) overlay of 3D image on the patient during dental surgery. Nevertheless, these techniques still assume avoidance of teeth occlusion.

In this work, we are primarily interested in principal component analysis (PCA) on 6-dimensional jaw motions in  $se(3)$ . Since linear PCA fails to capture the non-linear trajectories of motions, we resort to non-linear PCA (NLPCA) methods [26]. Popular non-linear methods include principal curves [27], locally linear embedding (LLE) [28] and Isomap [29]. In particular, we show that the input data can be explained using only three parameters corresponding to three main modes of jaw motions using a special type of autoencoder termed Nonlinear PCA (NLPCA) [8].

### 3 METHOD

#### 3.1 Data Acquisition and Preparation

We start by obtaining the models of the upper and lower teeth of our actor by a standard dental scanning procedure,



Fig. 3. Our recording setup, featuring three tripod-mounted GoPro cameras and diffuse light sources.



Fig. 4. Sample teeth segmentation results from synchronized images in three video cameras.

producing detailed tooth geometry with distances in millimeters, see Fig. 2.

We capture the dynamic teeth performances using three tripod-mounted GoPro Hero 5 Black cameras, see Fig. 3. To reconstruct the teeth poses from the three camera videos, we exploit the teeth's position and shape information that are implicitly encoded in the video frames. We extract this information by segmenting out the teeth from the video frames, as shown in Fig. 4. The segmentation of the video frames is performed by employing the Roto brush tool in Adobe Effect [30]. The tool required minimal user interaction (i.e., only a few strokes) to achieve the teeth segmentation. Please note that gums between the teeth are allowed to be part of the teeth segmentation, since the color variation between them may be very small, as shown in Fig. 4. For additional technical details regarding our data capture and processing please see Section 5.

Only the relative motion between the skull and the mandible is relevant for further processing. The actual pose or orientation of the head is not important. To remove the global pose of the head from our study, we proceed as follows. For a given image, let  $(R^L, t^L)$  and  $(R^U, t^U)$  be the rigid motion (rotation, translation) of the lower and upper teeth, respectively. Let us denote  $(R, t)$  to represent the relative motion between the skull and mandible bones and it can be derived as follows:

$$\begin{bmatrix} R^U & t^U \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R^L & t^L \\ 0 & 1 \end{bmatrix}. \quad (1)$$



From Eqn. (1), we can compute

$$R = (R^U)^T R^L \quad (2)$$

$$t = (R^U)^T (t^L - t^U) \quad (3)$$

### 3.2 Learning Jaw Kinematics from Data

Let  $(R_i, t_i)$  be the relative jaw transformation of the frame indexed by  $i$  from the input video. Our goal is to construct an anatomically realistic jaw kinematics model which can explain all of the measured relative jaw transformations  $\{(R_i, t_i)\}_{i=1}^n$  (We used  $n = 833$  frames from our input training video) while providing the user with intuitive parameters for controlling the jaw's poses.

Each of our input data points  $(R_i, t_i)$  lies on a  $SE(3)$  manifold (the manifold of rigid body transformations), which is nonlinear and thus not ideal for further analysis. Therefore, we map each of our data points from  $SE(3)$  to the corresponding Lie algebra  $se(3)$  using the logarithmic mapping, which has a closed-form expression in the case of  $SE(3)$  [31]. Geometrically, this corresponds to unfolding the relevant part of the non-linear manifold  $SE(3)$  to a linear space (the Lie algebra). All of our subsequent analysis will be performed on the data points in  $se(3)$ . We denote the resulting vectors as  $\{v_i\}_{i=1}^m$ , where  $v_i \in \mathbb{R}^6$ .

Our first attempt to analyze the input training data  $v_1, \dots, v_m$  is to apply Principal Component Analysis (PCA), producing six principal components  $\mathbf{p}_i \in \mathbb{R}^6$  with  $i = \{1, 2 \dots 6\}$ , where their corresponding six singular values are given by the set  $\{10.38, 0.73, 0.44, 0.34, 0.25, 0.07\}$ . With this PCA basis, each jaw pose can be written as

$$\mathbf{p} = \sum_{j=1}^6 w_j \mathbf{p}_j, \quad (4)$$

where  $w_j \in \mathbb{R}$  are weights (PCA loadings) of each of the principal components. The resulting  $\mathbf{p} \in se(3)$  can be mapped to  $SE(3)$  by computing the exponential mapping (closed-form expression [31]). The resulting rigid body transformation can be then used to transform the mandible mesh, producing visualization of the jaw motion represented by the linear combination of the principal components.

In Fig. 5(a), we visualize the effect of the first principal component  $\mathbf{p}_1$  which, not surprisingly, captures the dominant motion – opening of the mouth. However, the visualization in Fig. 5(a) also reveals the limitations of PCA (please see also the animation in the accompanying video). Specifically, even though the jaw motion generated by scaling the first principal component roughly corresponds to opening of the mouth and includes both rotation and translation, it fails to accurately capture the trajectory of the mandibular condyle. When opening the mouth, the mandibular condyle slides over the surface of the zygomatic process (see Fig. 1). Because the zygomatic process is curved (as opposed to flat), this results in a non-linear trajectory. The non-linearity of this trajectory can be observed by real-time magnetic resonance imaging of a human subject performing voluntary mouth opening and closing [32] (please see the accompanying video). Unfortunately, the first principal component produces only a crude approximation of this nonlinear trajectory (specifically, this approximation

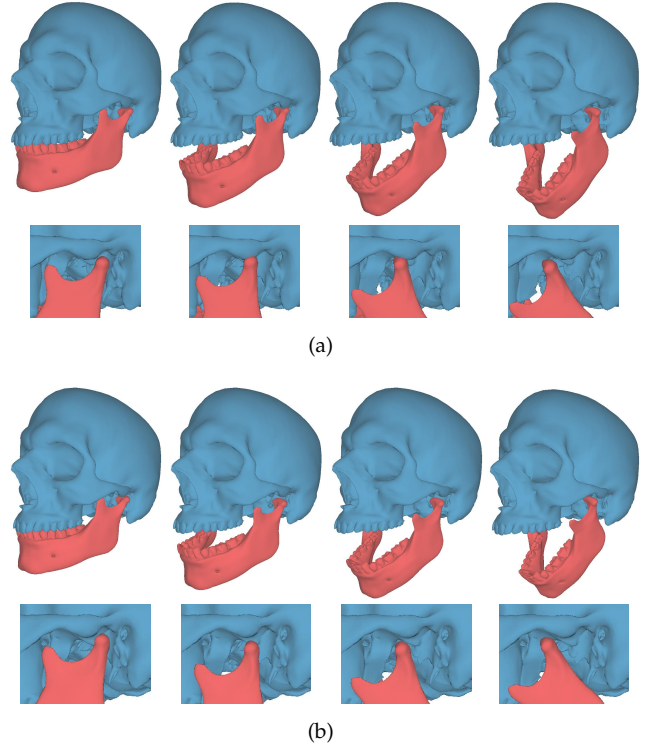


Fig. 5. Motions of mouth opening that are generated using PCA (a) and NLPCA autoencoder (b). The jaw poses in (a) are generated by scaling the first principal component:  $w_1 \mathbf{p}_1$ , for  $w_1 = \{-1.6157, -0.5717, 0.4723, 1.5165\}$  (left to right), while the jaw poses in (b) are generated as  $\Phi_{gen}(c_1, 0, 0, 0, 0)$  where  $c_1$  is  $\{-0.1943, -0.0195, 0.1552, 0.3300\}$  (left to right). Note that the zoom out in the bottom row of (a) illustrates that the PCA does not learn the correct non-linear trajectory and the mandibular condyle intersects the zygomatic process, while in (b) the condyle slides along the zygomatic process, which is the desired anatomically-realistic behavior.

corresponds to a straight line in  $se(3)$ ). There are other limitations with PCA. The second and the remaining principal components do not correspond to anatomically meaningful motions such as lateral excursions and pro/re-trusion. It is also challenging to generate anatomically realistic motions by considering the 5 or 6 principal components within their boundary values.

To be able to capture the correct anatomical behavior of the temporomandibular joint during mouth opening with a single parameter, we turn to a more powerful, non-linear data analysis tool: autoencoder neural networks [33]. We trained various autoencoder architectures using TensorFlow [34] and found that substantial dimensionality reduction can be achieved, e.g., using only three dimensions can reproduce the training data  $\{v_1, \dots, v_m\}$  very accurately (unlike PCA, which produces relatively large error with only three principal components). Even though autoencoders without any non-linear units are almost equivalent to PCA [35], there is a catch in the “almost.” Specifically, the reduced parameters produced by the encoder part of an autoencoder do not have any hierarchical meaning as is the case in PCA. In PCA, the first principal component explains the largest variance in the data, but this is not the case for the first component produced by the encoder. Fortunately, we found that a solution to this problem has already been described by Scholz and colleagues [8], [36], who proposed a modified

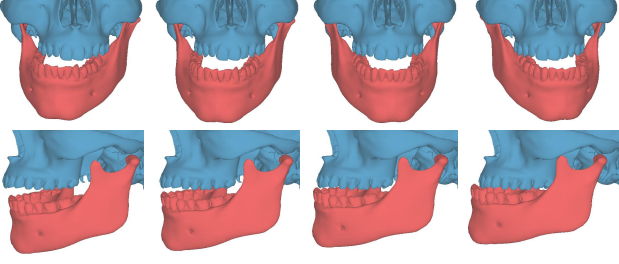


Fig. 6. Top row: lateral excursions learned by non-linearly reducing  $D_{23}$  to a single dimension; Bottom row: the same approach applied to  $D_{45}$  produces one-dimensional parameterization of protrusion and retrusion.

autoencoder network which mimics the hierarchical property of the principal components. Their approach is called NLPCA (for Non-Linear PCA).

Let  $\mathcal{S}$  be a data space given by our 6-dimensional data points in  $se(3)$  and  $\mathcal{P} \subseteq \mathbb{R}^n$  a component space with  $n \leq 6$ . NLPCA learns two nonlinear functions  $\Phi_{extr}$  and  $\Phi_{gen}$ , where  $\Phi_{extr} : \mathcal{S} \mapsto \mathcal{P}$  is called *component (or feature) extraction function* (corresponding to the encoder part of an autoencoder) and  $\Phi_{gen} : \mathcal{P} \mapsto \mathcal{S}$  is called *data generation function* (corresponding to the decoder part). The extraction function  $\Phi_{extr}$  transforms a 6-dimensional data point into the corresponding component representation  $\mathbf{c} = (c_1, c_2, \dots, c_n)$ , while the generation function  $\Phi_{gen}$  performs the reverse, i.e., reconstructs the original data points from its lower-dimensional component representation.

With our training data  $v_1, \dots, v_m$ , we found that only five dimensions are needed with NLPCA; the sixth dimension introduces only minimal modifications which can be attributed to noise in the input data. Furthermore, visualizing the effect of the individual components, we observe that the nonlinear characteristics of the mouth opening motion are effectively captured by first component ( $c_1$ ). Visualizing the results of  $\Phi_{gen}(c_1, 0, 0, 0, 0)$  for varying  $c_1$  produces anatomically realistic nonlinear trajectory of the mandibular condyles, see Fig. 5(b). Most importantly, the mandibular condyle now slides along the zygomatic process, as opposed the unrealistic intersection produced by PCA (Fig. 5(a)). Even though this five-dimensional model can accurately represent all possible motions of the jaw, we found it is possible to reduce the number of parameters further, providing a more compact and intuitive interface to the user while increasing the error only negligibly.

By visualizing the effect of the components  $c_2$  and  $c_3$ , we found that both of them correspond to lateral excursions (moving the jaw sideways, left and right). Similarly, the last two components  $c_4$  and  $c_5$  correspond to forward and backward motion of the jaw, known as protrusion and retrusion. This observation motivates the final refinement of our model, which we describe next. First, we generate a set of data points (denoted as  $D_{23}$ ) which correspond to zeroing all components except for  $c_2$  and  $c_3$ , i.e.,  $\Phi_{gen}(0, c_2, c_3, 0, 0)$ , where the values of  $c_2$  and  $c_3$  are given by encoding of our training data. Similarly, we construct another set of data points (denoted as  $D_{45}$ ) in terms of the components of  $(0, 0, 0, c_4, c_5)$ . Next, we run NLPCA [8] on the  $D_{23}$  dataset to nonlinearly reduce its dimensionality to one. We denote the resulting one-dimensional generation function

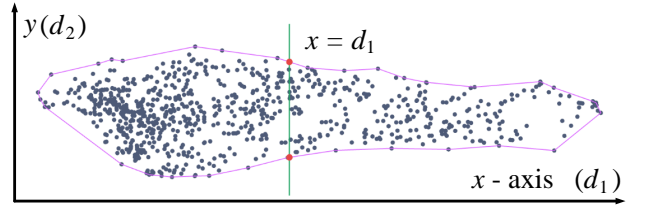


Fig. 7. The range of the parameter  $d_2$  is determined with respect to a given  $d_1$  (mouth opening). The parameters  $(d_1, d_2)$  of our original data points are projected onto the  $x$ - $y$  plane and the valid range of  $d_2$  is determined by  $y$ -coordinates of the intersection points between the line ( $x = d_1$ ) and a polygonal boundary of the 2D projection points.

(decoder) as  $\Phi_2(d_2)$ , where  $d_2$  is a scalar input parameter. We observe that by varying  $d_2$ , the function  $\Phi_2$  can explain the entire range of lateral excursions (sideways motions of the jaw from left to right), during which the trajectory of the condyles again succeeds to avoid inter-penetrations with the zygomatic process (similarly as before for jaw opening). Even better, it turns out the similar trick works also for  $D_{45}$ ! Again, we execute NLPCA on the  $D_{45}$  dataset and obtain a one-dimensional generation function  $\Phi_3(d_3)$ , such that varying the scalar parameter  $d_3$  produces an entire range of anatomically realistic protrusion and retrusion. These motions are visualized in Fig. 6. To make our notation succinct, let us introduce a shorthand  $\Phi_1(d_1) := \Phi_{gen}(d_1, 0, 0, 0, 0)$  where  $d_1$  is a parameter corresponding to mouth opening.

Our final model parameterizing the jaw kinematics with only three dimensions is a linear combination of the individual parts:

$$\Phi(d_1, d_2, d_3) = \Phi_1(d_1) + \Phi_2(d_2) + \Phi_3(d_3) \quad (5)$$

Linear combination is justified by the fact that the Lie algebra  $se(3)$  is a linear space and the maximal relative rotations are bounded (far below 180 degrees) [37]. As we discuss in more details in Section 5, this final  $\Phi(d_1, d_2, d_3)$  introduces only marginally higher error compared to the five-dimensional NLPCA model, while using fewer parameters and being much more intuitive to the user.

*Valid Component Representation.* It is evident that for a valid parameter representation  $\mathbf{d} = (d_1, d_2, d_3)$ , each of its parameters should be in a specified range. Since each of our original data points has a corresponding parameter representation, a simple way to determine the parameter ranges would be to bound each parameter using the parameter representations of the given data points. However, in real jaw motion, the ranges of parameters  $d_2$  and  $d_3$  are dependent on the current value of  $d_1$ . To demonstrate this, we invite the reader to try sliding their lower teeth to the right and then opening the mouth – the lower teeth are automatically moved back to the center. Therefore, we need to dynamically determine the valid ranges for the parameters  $d_2$  and  $d_3$ , with respect to current  $d_1$ . As shown in Fig. 7, we use  $d_2$  as an example to describe our solution. First, we project the parameters  $(d_1, d_2)$  of our original data points onto a  $x$ - $y$  plane. Then, a polygonal boundary is formed using the alpha shape ( $alpha = 0.12$  in our experiments) of these planar points. For a given  $d_1$ , the current range of  $d_2$  is determined by the intersection points between the polygonal boundary and the line  $x = d_1$ .

#### 4 MODEL-BASED TEETH MOTION TRACKING

Essentially, our teeth motion tracking problem is a 3D-2D matching problem [38], [39] where the task is to find the best pose of a 3D model so that its 2D projections are well aligned with the 2D images. In our scenario, the 3D model is a teeth model, corresponding to either the upper or lower teeth of the performer, and the model pose corresponds to a rigid motion. More formally, we denote the teeth model as  $\mathcal{M}$  and represent the rigid motion as a 6-tuple  $T = \{\theta_x, \theta_y, \theta_z, t_x, t_y, t_z\}$ , where  $\theta_x, \theta_y, \theta_z$  are the rotation angles around the  $x$ -,  $y$ -, and  $z$ -axes, respectively, and  $t_x, t_y, t_z$  are the translation offsets along the  $x$ ,  $y$ , and  $z$  directions, respectively. To make the projections of  $\mathcal{M}$  most consistent with the (three in our case) images that corresponds to a common video frame, we can solve the following optimization problem:

$$\arg \min_T \sum_{i=1}^3 E(\mathbb{P}_i(R\mathcal{M} + t), I_i), \quad (6)$$

where  $R = R_z(\theta_z)R_y(\theta_y)R_x(\theta_x)$  with  $R_x(\cdot)$ ,  $R_y(\cdot)$ , and  $R_z(\cdot)$  being the rotation matrices about the  $x$ -,  $y$ -, and  $z$ -axes, respectively, with a specified angle,  $t = (t_x, t_y, t_z)$  is a translation vector,  $\mathbb{P}_i(\cdot)$  is an operation that projects a 3D model to the image plane of the  $i$ th camera,  $I_i$  corresponds to a video frame image that is captured by the  $i$ th camera, and  $E(\cdot, \cdot)$  defines an energy function which measures the inconsistency or discrepancy between the 2D projection of the 3D model and the real 2D image. In our implementation, we use hardware pipeline via OpenGL to conduct the projection operation  $\mathbb{P}_i(\cdot)$ , with respect to the cameras' intrinsic and extrinsic parameters.

To define  $E$ , our idea is to fully exploit the geometry information of the teeth model (see Fig. 8) as well as the shape and position information of the performer's teeth that are encoded in the teeth segment of the captured images. We define  $E$  as follows:

$$E = E_{SP} + \lambda E_{BND}. \quad (7)$$

where  $E_{SP}$  measures the overlap of the teeth model with the actual visible teeth segment while  $E_{BND}$  ensures alignment of the roof region of frontal teeth which we call boundary of interest (BOI). We use fixed weight  $\lambda = 0.4$  for all of the results in this paper.

To compute  $E_{SP}$  for every captured image we render its teeth segmentation with white color into clean (black) screen buffer (see Fig. 9a). Then, the teeth model at current pose is rendered in red color (with alpha blending  $\alpha = 0.5$ ), i.e., projected with respect to the intrinsic and extrinsic parameters of the corresponding camera. Thanks to this second rendering pass the pixels of the teeth segmentation that are overlapped with the projection of the teeth model will change to pink color (see Fig. 9(b)). The closer the poses of the teeth model are to the actual performer's teeth, the fewer pixels with white color remain. Therefore, we define the energy term  $E_{SP}$  as follows:

$$E_{SP} = \#Pixel\_White, \quad (8)$$

where  $\#Pixel\_White$  refers to the number of white pixels in the screen buffer.



Fig. 8. The teeth model is roughly marked out from the 3D scans of the performer's teeth rows. The geometry of the teeth model consists of the triangles in the red and green regions, where the green parts correspond to the boundary of interest of the teeth model.

Due to the occlusion by lip or tongue, the teeth segmentation of the performer is usually smaller than the 2D projection of its teeth model, such that the energy term  $E_{SP}$  may lead to sub-optimal result, as shown in Fig. 9 (c). We use the energy term  $E_{BND}$  to tackle this problem. To compute the boundary term  $E_{BND}$ , during the rendering of teeth model we change the color of corresponding BOI to green (see Fig. 9d, e). This ensures that when the 2D projection of the roof region is perfectly aligned with the teeth segmentation, the number of pixels with light green color is increased while the number of pixels with dark green color is minimal. Thus, we compute the energy term  $E_{BND}$  as:

$$E_{BND} = \#Pixel\_DGreen, \quad (9)$$

where  $\#Pixel\_DGreen$  refers to the number of dark green pixels in the screen buffer. Please note that the energy term  $E_{SP}$  will prevent the 2D projection of the roof region being pulled into the segmentation too much, since it will increase the number of white pixels.

To minimize  $E$ , we combine a gradient descent-like approach with dense sampling which, in practice, ensures good local optima while being computationally efficient. During the descent we estimate the direction of the energy gradient relative to  $T$  using finite differences. Along this search direction, we then use golden-section search [40] to find the new optimal value for  $T$ . To reduce the number of degrees of freedom we first fix translation ( $t_x, t_y, t_z$ ) and update rotation ( $\theta_x, \theta_y, \theta_z$ ); then, rotation is fixed and translation updated. We repeat this process until convergence. To avoid getting stuck in poor local minima, we subsequently refine the pose  $T$  by sampling the space of possible configurations more densely in a small neighbourhood around the current pose  $T$ .

To further improve the optimization process we use the pose from previous frame as the initial pose  $T$ . For the first frame, we bootstrap the process by manually selecting four vertices on the teeth model and find their corresponding positions in two captured images. Through triangulation in stereo analysis, we obtain the corresponding 3D positions for selected vertices; then, an initial guess  $T$  for the first



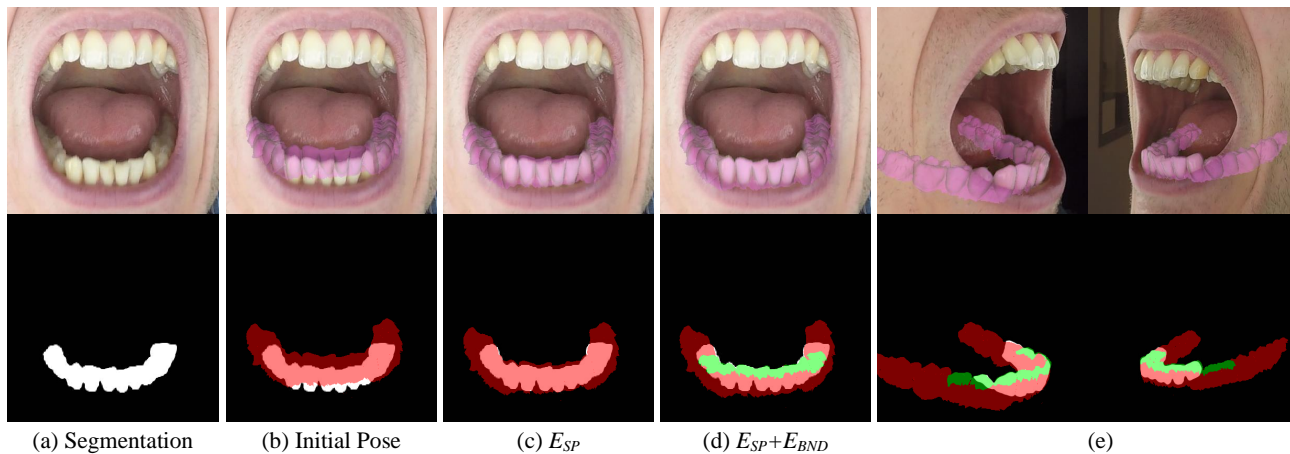


Fig. 9. Optimization energy of the (lower or upper) teeth pose. Given the captured images in Fig. 4 and the initial pose of the teeth model, we render for each camera the 2D projection of the model's geometry and the teeth segmentation into the same screen buffer (b). The overlapping pixels are used to define energy terms (c,d) which measure how consistent is between the poses of the teeth model and the real teeth at the current video frame. In (d,e), the projections of the optimal teeth pose in all of the three cameras are shown.

frame can be obtained by fitting the initial positions of the selected vertices to their new positions [41].

## 5 EXPERIMENTAL RESULTS

The algorithms were implemented in Matlab/C++ and were run on a 2.9GHz Intel Core(TM)i7 processor (32GB RAM) with a single CPU thread.

The teeth performance capture is achieved using three GoPro Hero 5 cameras at  $1920 \times 1080$  resolution, and 24p linear video mode settings. The performer places his mouth approximately 20 cms from each of the three cameras to allow for optimum coverage and larger overlap between the cameras. Using planar checkerboard patterns, the cameras are calibrated for both intrinsic and extrinsic parameters. The three cameras are synchronized by turning on and off lights. We locate the frames with sudden intensity changes to match the frames from different cameras. An accurate hardware synchronization could also be employed to eliminate this step.

**Teeth motion tracking.** On average, the total optimization time is about 15 seconds per frame and the main bottleneck is the computational time for computing the energy function  $E$  in Eqn. (7) which involves counting the number of pixels of specific color in the screen buffer. In our current implementation, we use  $940 \times 480$  screen buffer with 24 bits per pixel, where the pixels are transferred from the GPU into CPU and counted. The computational efficiency can be improved by considering parallel counting on the GPU, e.g., using histogram operation available in CUDA SDK.

We recorded 833 teeth poses of the actor and performed teeth tracking on all these frames for jaw kinematic modeling. We show the results in the supplementary video; a sample of teeth tracking results is shown in Fig. 10.

Error in teeth tracking can come from several sources: calibration, errors in 3D scans of the teeth, and segmentation errors. Unfortunately, it is difficult to get the ground truth for this experiment and therefore, the validation is mainly done qualitatively. Since we have images from three camera views, we are able to visualize the tracked poses without



Fig. 10. Tracking results from several teeth poses. Each row corresponds to a specific teeth pose and its 2D projections in the three cameras are shown. Note that the 2D projection (pink overlaid region) from the 3D model covers both the visible and the occluded teeth.

depth ambiguity, as shown in Fig. 10. Furthermore, we apply the tracked poses of the upper and lower teeth to the skull and mandible models of the performer, respectively. As shown in Fig. 11, we can create augmented reality sequence by overlaying the skull and mandible models on the captured images. From the reconstructed motions of the skull and mandible models, we can also stabilize skull motions and visualize only the relative motion of the mandible as shown, e.g., in Fig. 14. Note that the skull and mandible models are segmented and reconstructed from the performer's MRI data. While these models are helpful for the visual evaluation of the tracking results, they are not necessary for our tracking approach. It can be seen from



Fig. 11. The tracked poses of the upper and lower teeth can also be illustratively shown using the projection of the skull (blue) and mandible models (red) on the captured images.

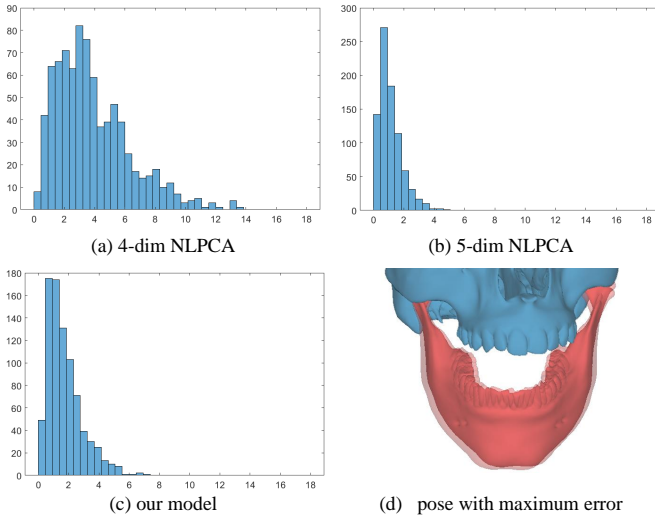


Fig. 12. For each of captured poses, we use NLPKA with different number of component dimensions to reconstruct its original pose in (a) and (b). In (c), the original pose is reconstructed using our final kinematic model according to Eqn. (5). The histogram of the reconstruction errors of the 833 captured poses is shown (horizontal axis: error in millimeter; vertical axis: frequency). For the pose with maximum reconstruction error (i.e., 7.2216) using our final kinematics model, its original and reconstructed poses are shown in an overlapping way (d), where the transparent part corresponds to the difference between the two poses.

the present examples and the supplementary video that the tracked poses of the teeth model are visually well aligned with the real teeth of the performer.

In addition to visual comparison, we provide an approximate metric for measuring the accuracy. Since feature points and their correspondences are difficult to obtain in the case of teeth images, we instead measure the difference in the area from the projection of the teeth model and the teeth in the images. The difference in area can be approximately computed as the ratio between energy term  $E_{SP}$  in Eqn. (8) and the total pixel number of the teeth segmentation. The average error on the captured images is about 0.754%.

**Jaw kinematics model.** With the tracked poses of the performer’s lower and upper teeth, the jaw motion extraction, including the data mapping from  $SE(3)$  to  $se(3)$ , can be computed efficiently at the rate of 4.6 seconds for all the 833 teeth poses. In Matlab, the nonlinear PCA takes 108 seconds for learning the jaw kinematics model.

In Fig. 12, we use the learned jaw kinematics model to reconstruct the originally captured jaw poses. Then, we measure the reconstruction error by computing the differ-

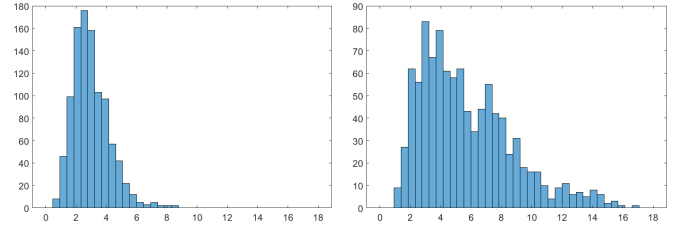


Fig. 13. One thousand jaw poses are randomly synthesized using our kinematics model with and without the dynamic adjustment of the valid ranges for the parameters  $d_2$  and  $d_3$ . The histogram of the accuracy of the 1000 synthesized poses is given (left is with the dynamic adjustment while right is without the dynamic adjustment), whose layout is similar to that of Fig. 12.

ence between each captured pose and its reconstruction. To measure the difference between a pair of given poses, we apply them to the mandible mesh, respectively, and then compute the maximum vertex distance between the two transformed mandible meshes. As shown in Fig. 12, our jaw kinematics model can reconstruct the jaw kinematics of a real person up to an error of 1.73 millimeters on average. To further validate that our jaw kinematics model generates the anatomically realistic poses, we randomly synthesized 1000 jaw poses using our model. For each synthesized pose, we measure how realistic it is by computing the difference between the pose and its closest one in the real jaw poses that are captured from the performer. The result is shown in Fig. 13.

Finally, using the jaw kinematics model, we can then synthesize anatomically realistic and visually pleasing jaw poses, i.e., by adjusting the parameters  $\{d_i\}_{i=1}^3$  of the model in their respective valid ranges. To facilitate this step, we designed a simple user interface where the user can easily explore the constrained nonlinear space of the jaw kinematics through three sliders, each of which is used to tune a parameter and thus corresponds to a semantically meaningful mode of the jaw motion. An example of this user interface is visible in Fig. 14.

**Inverse Kinematics (IK).** In addition, we allow the user to generate the jaw poses via direct manipulation. To edit the jaw pose, the user can pick a vertex on the mandible mesh and mouse-drag it to a new position. The system then automatically updates the parameters of our jaw kinematics model to find the optimal pose that is consistent with this user-specified positional constraint. Given the new position of the selected vertex ( $p'$ ), our goal is to find parameters  $\mathbf{d} = (d_1, d_2, d_3)$  so that the transformed position of the selected mesh vertex ( $p_d$ ) is as close as possible to  $p'$ :

$$\arg \min_{\mathbf{d}} \|p' - p_d\|^2 \quad \text{subject to valid } \mathbf{d}. \quad (10)$$

We can efficiently solve Eqn. (10) with a brute-force approach. We sample the space of possible configurations densely in a small neighbourhood around the current pose and evaluate the best candidate. This takes only few milliseconds. An example of an interactive IK session is presented in the accompanying video.

**Animation.** To generate continuous jaw motions, the user can design several key poses of the jaw through our user interface, and then the intermediate poses can be



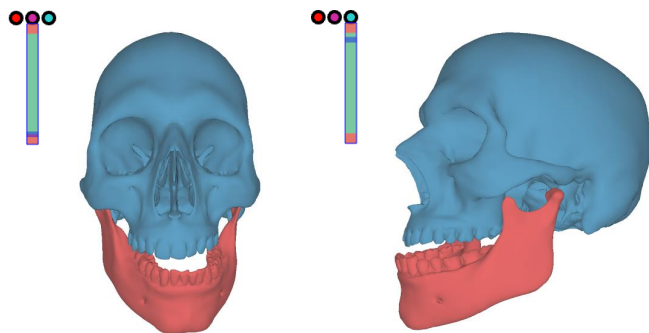


Fig. 14. Our user interface includes three circle points which correspond to the modes of mouth opening/closing, lateral excursions, and pro/re-trusion, respectively. When a circle point is selected, a slider appears and the user can move the slider to adjust the corresponding parameter for the desired jaw pose.

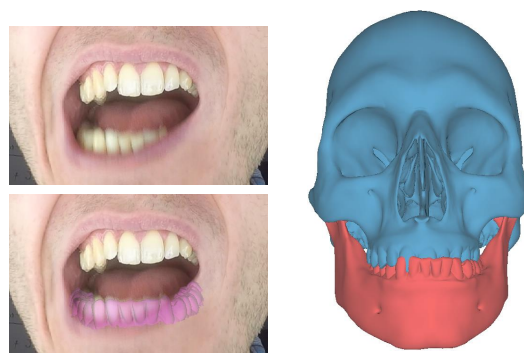
computed using simple linear interpolation between the parameters of the key poses. As shown in Fig. 16, such a simple interpolation scheme already generates high-quality jaw motions which are comparable to real jaw motions. Furthermore, note that our anatomically realistic jaw kinematics model can be easily integrated into various applications such as anatomically-constrained monocular face capture [9] or physics-based facial animation [6].

**Limitations.** Rapid motion of the performer can lead to blurred images, which in turn can lead to segmentation errors and incorrect teeth tracking as shown in Fig. 15(a). The use of cameras with higher frame rate can eliminate this problem.

Another limitation stems from the valid ranges of the parameters in our jaw kinematics model. Currently, the ranges of the parameters correspond to a roughly linear approximation of the boundary of the original data points. While such a scheme is simple and practical, invalid jaw poses may still happen when extreme poses are synthesized using the parameters that locate near the boundary of their valid ranges. An example of such scenario is shown in Fig. 15(b), where the mouth is nearly closed, while the jaw is in the rightmost of the lateral excursion and in the maximum of protrusion. A possible solution might be a smooth and tighter bounding volume, e.g., the isosurface representation, as done for the kinematic modeling of joints [42].

## 6 CONCLUSION

In this paper, we build an anatomically realistic jaw kinematics model from data. We use three tripod-mounted GoPro Hero 5 Black cameras to capture the dynamic teeth performances of an actor, from which 833 jaw poses are tracked and extracted. Then, the jaw kinematics is learned from these jaw poses using the Non-Linear PCA (NLPCA), which effectively captures the nonlinear characteristics of the jaw's motion. The resulting jaw kinematics model has three parameters, each of which corresponds to an intuitive mode of the jaw's motion, i.e., mouth opening, lateral excursions, and pro/re-trusion. Finally, our jaw kinematics model provides an intuitive interface allowing the animators to explore realistic jaw motions in a user-friendly



(a)

(b)

Fig. 15. Limitations and failure cases: (a) We show poor teeth tracking results in the case of images with blur. (b) Invalid jaw poses can be produced when the parameters  $\{d_1, d_2, d_3\}$  are chosen near their respective boundary values. In particular, the boundary conditions for parameters  $d_2$  and  $d_3$  vary and they depend on the current value of  $d_1$ . However, such boundary conditions are just linear approximations, and may not be accurate.

way. Such model is also useful for various application scenarios to guarantee anatomically correct results, such as anatomically-constrained facial animation [9] or physics-based face simulation [6].

In future, we plan to create a library of user-tailored jaw-kinematics models since each person has her personalized capacity of jaw motion. We plan to obtain completely automatic segmentation algorithm using graph cuts algorithm, where the parameters of the energy function will be learned from the training data with manually annotated segmentation labels. An interesting line of research would be to actually use the learned jaw kinematics model in the teeth tracking problem and vice versa.

## ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-1617172 and IIS-1622360. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Wenwu Yang was partially funded by the Science and Technology Agency projects of Zhejiang Province (2016C33171). Daniel Sýkora was funded by the Fulbright Commission in the Czech Republic and by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center). We also gratefully acknowledge the support of Research Center for Informatics (No. CZ.02.1.01/0.0/0.0/16\_019/0000765), Activision and Mitsubishi Electric Research Labs (MERL) as well as hardware donation from NVIDIA Corporation.

## REFERENCES

- [1] D. Terzopoulos and K. Waters, "Physically-based facial modelling, analysis, and animation," *Computer Animation and Virtual Worlds*, vol. 1, no. 2, pp. 73–80, 1990.
- [2] E. Sifakis, I. Neverov, and R. Fedkiw, "Automatic determination of facial muscle activations from sparse motion capture marker data," *Acm transactions on graphics (tog)*, vol. 24, no. 3, pp. 417–425, 2005.

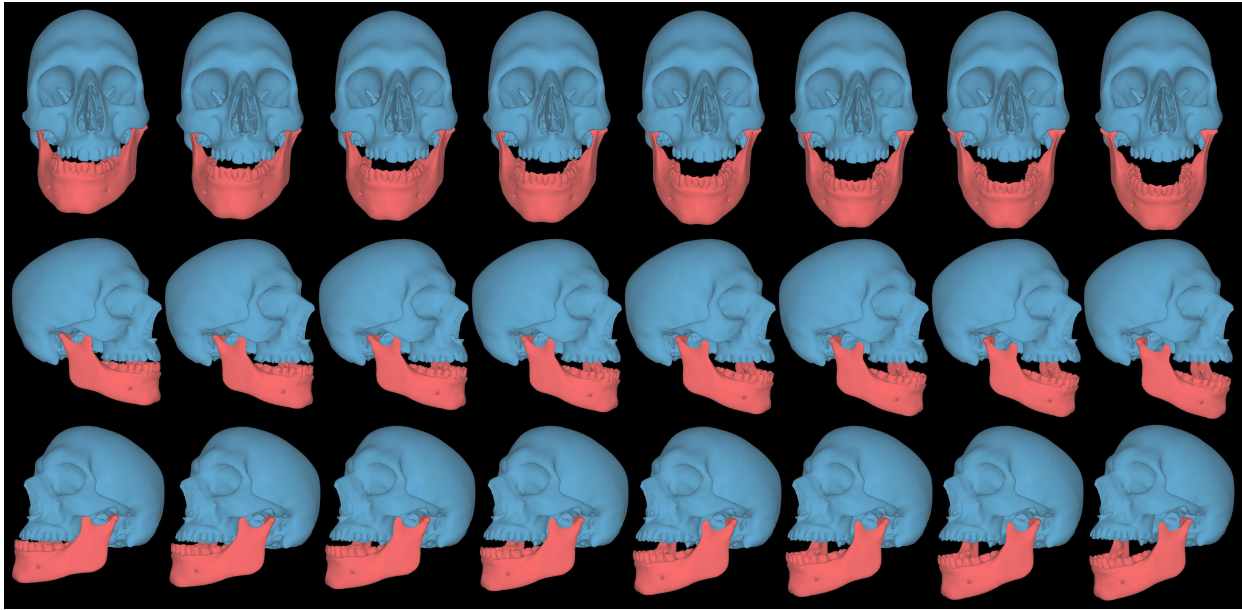


Fig. 16. Using our interface, the user can create two jaw poses (left and right) and the system automatically generates a jaw animation by keyframe interpolation, where each row corresponds to a different perspective.

- [3] M. Cong, M. Bao, K. S. Bhat, R. Fedkiw *et al.*, "Fully automatic generation of anatomical face simulation models." *ACM*, 2015, pp. 175–183.
- [4] L. Lan, M. Cong, and R. Fedkiw, "Lessons from the evolution of an anatomical facial muscle model," in *Proceedings of the ACM SIGGRAPH Digital Production Symposium*. *ACM*, 2017, p. 11.
- [5] Y. Kozlov, D. Bradley, M. Bächer, B. Thomaszewski, T. Beeler, and M. Gross, "Enriching facial blendshape rigs with physical simulation," in *cgf*, vol. 36, 2017, pp. 75–84.
- [6] A.-E. Ichim, P. Kadleček, L. Kavan, and M. Pauly, "Phace: Physics-based face modeling and animation," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 153:1–153:14, 2017.
- [7] H.-J. Yoon, E. Baltali, K. D. Zhao, J. Rebellato, D. Kademani, K.-N. An, and E. E. Keller, "Kinematic study of the temporomandibular joint in normal subjects and patients following unilateral temporomandibular joint arthroscopy with metal fossa-eminence partial joint replacement," *Journal of oral and maxillofacial surgery*, vol. 65, no. 8, pp. 1569–1576, 2007.
- [8] M. Scholz, M. Fraunholz, and J. Selbig, "Nonlinear principal component analysis: Neural network models and applications," *LNCSE*, vol. 58, pp. 44–67, 2008.
- [9] C. Wu, D. Bradley, M. Gross, and T. Beeler, "An anatomically-constrained local deformation model for monocular face capture," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 115, 2016.
- [10] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, "Learning a model of facial shape and expression from 4d scans," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, p. 194, 2017.
- [11] J. H. Koolstra, "Dynamics of the human masticatory system," *Critical reviews in oral biology & medicine*, vol. 13, no. 4, pp. 366–376, 2002.
- [12] M. De Zee, M. Dalstra, P. M. Cattaneo, J. Rasmussen, P. Svensson, and B. Melsen, "Validation of a musculo-skeletal model of the mandible and its application to mandibular distraction osteogenesis," *Journal of biomechanics*, vol. 40, no. 6, pp. 1192–1201, 2007.
- [13] A. G. Hannam, I. K. Stavness, J. E. Lloyd, S. S. Fels, A. J. Miller, and D. A. Curtis, "A comparison of simulated jaw dynamics in models of segmental mandibular resection versus resection with allopastic reconstruction," *The Journal of Prosthetic Dentistry*, vol. 104, no. 3, pp. 191 – 198, 2010.
- [14] A. G. Hannam, I. Stavness, J. E. Lloyd, and S. Fels, "A dynamic model of jaw and hyoid biomechanics during chewing," 2008.
- [15] S. E. Henderson, R. Desai, S. Tashman, and A. J. Almaraz, "Functional analysis of the rabbit temporomandibular joint using dynamic biplane imaging," *Journal of biomechanics*, vol. 47, no. 6, pp. 1360–1367, 2014.
- [16] V. Lepetit and P. Fua, "Monocular model-based 3d tracking of rigid objects: A survey," *Foundations and Trends in Computer Graphics and Vision*, p. 189, 2005.
- [17] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.
- [18] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 681–685, 2001.
- [19] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [20] G. Klein and D. Murray, "Full-3d edge tracking with a particle filter," in *British Machine Vision Conference (BMVC)*, vol. 3, 2006, p. 11191128.
- [21] S. Ramalingam, S. Bouaziz, P. Sturm, and M. Brand, "Skyline2gps: Localization in urban canyons using omni-skylines," in *International conference on intelligent robotics(IROS)*, 2010.
- [22]
- [23] J. Wang, H. Suenaga, K. Hoshi, L. Yang, E. Kobayashi, I. Sakuma, and H. Liao, "Augmented reality navigation with automatic marker-free image registration using 3-d image overlay for dental surgery," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 4, pp. 1295–1304, 2014.
- [24] A. Aichert, W. Wein, A. Ladikos, T. Reichl, and N. Navab, "Image-based tracking of the teeth for orthodontic augmented reality," in *Proceedings of the 15th International Conference on Medical Image Computing and Computer-Assisted Intervention - Volume Part II*, ser. MICCAI'12, 2012, pp. 601–608.
- [25] J. Wang, H. Suenaga, L. Yang, E. Kobayashi, and I. Sakuma, "Video see-through augmented reality for oral and maxillofacial surgery," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 13, no. 2, p. e1754, 2017.
- [26] M. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE*, vol. 37, pp. 233–243, 1991.
- [27] T. Hastie and W. Stuetzle, "Principal curves," *American Statistical Association*, vol. 84, pp. 502–516, 1989.
- [28] S. T. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–6, 2000.
- [29] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, p. 23192323, 2000.
- [30] X. Bai, J. Wang, D. Simons, and G. Sapiro, "Video snapcut: Robust video object cutout using localized classifiers," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 70:1–70:11, 2009.

- [31] R. M. Murray, S. S. Sastry, and L. Zexiang, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., 1994.
- [32] S. Zhang, N. Gersdorff, and J. Frahm, "Real-time magnetic resonance imaging of temporomandibular joint dynamics," *The Open Medical Imaging Journal*, vol. 5, pp. 1–9, 2011.
- [33] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [34] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'16, 2016, pp. 265–283.
- [35] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Netw.*, vol. 2, no. 1, pp. 53–58, 1989.
- [36] M. Scholz and R. Vigarío, "Nonlinear pca: a new hierarchical approach," in *Proceedings of European Symposium on Artificial Neural Networks*, 2002, pp. 439–444.
- [37] M. Alexa, "Linear combination of transformations," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 380–387, 2002.
- [38] S. Gold, A. Rangarajan, C. ping Lu, and E. Mjolsness, "New algorithms for 2d and 3d point matching: Pose estimation and correspondence," *Pattern Recognition*, vol. 31, pp. 957–964, 1997.
- [39] B. Rosenhahn, C. Perwass, and G. Sommer, "Pose estimation of 3d free-form contours," *International Journal of Computer Vision*, vol. 62, pp. 267–289, 2005.
- [40] J. Kiefer, "Sequential minimax search for a maximum," *Proceedings of the American Mathematical Society*, vol. 4, no. 3, pp. 502–506, 1953.
- [41] O. Sorkine-Hornung and M. Rabinovich, "Least-squares rigid motion using svd," *Technical notes, ETHZ*, 2009.
- [42] L. Herda, R. Urtasun, P. Fua, and A. Hanson, "An automatic method for determining quaternion field boundaries for ball-and-socket joint limits," in *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, 2002, pp. 88–93.