IDIAP RESEARCH REPORT

IDIAP

Martigny - Valais - Suisse

Segmentation of X-ray Image Sequences Showing the Vocal Tract (with tool documentation)

Georg Thimm

IDIAP–RR 99-01

January 1999

Dalle Molle Institute
for Perceptual Artificial
Intelligence • P.O.Box 592 •
Martigny • Valais • Switzerland

phone +41 − 27 − 721 77 11
fax +41 − 27 − 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

IDIAP Research Report 99-01

Segmentation of X-ray Image Sequences Showing the Vocal Tract (with tool documentation)

Georg Thimm

January 1999

Abstract. The tongue, the lips, the palate, and the throat are tracked in X-ray images showing the side view of the vocal tract. This is performed by using specialized histogram normalization techniques and a new tracking method that is robust against occlusion, noise, and spontaneous, non-linear deformations of objects.

Although the segmentation procedure is optimized for the X-ray images of the vocal tract, the underlying tracking method can easily be used in other applications.

Keywords: contour tracking, edge template, joined forward-backward tracking

Acknowledgements: This work has been performed with financial support from the Swiss National Science Foundation under Contract No. 21 49 725 96.

1 Introduction

Although speech and speaker recognition systems archive nowadays high performances in laboratory conditions, their performance is still unsatisfying under real-life conditions. Several authors have suggested that more knowledge about the speech production process (*e.g* co-articulation, dynamics, inter/-intro speaker differences) might lead to an improved feature extraction method. Having this in mind, the ARTIST project was proposed to Swiss National Science Foundation. In this project, firstly articulatory features form X-ray movies are extracted (see section 2), and then the speech production process is thoroughly analysed.

Tracking is very difficult in the X-ray database. Our first attempt with some variations of point distribution models and grey-level models the surrounding of the points failed [Luettin and Thacker, 1997]. One of these variations used the *condensation* algorithm [Isard and Blake, 1996]. Furthermore, we observed, that the optical flow fields obtained for this database are - depending on applied algorithm, its parameterization, and the filtering of the images - almost random or widely unrelated to the motions of the articulators (we tested the algorithms reported to be best performing in [J.L. Barron and Fleet, 1994]).

The approach described in the following is not the first that tries to track articulatory features in this database. In [Laprie and Berger, 1996] and [Berger and Laprie, 1996], the contour of the tongue is tracked in X-ray images, but no final results were published and they did not track the lips and the jaws. We are not aware of tracking results published by other authors.

For completeness, we want to mention that other researches gain quantitative data on the motion of articulators in various ways:

- [Davis *et al.*, 1996] uses MRI and tags (small metal pellets attached to the tongue). The movements in 500ms windows for 4 vowels are extracted. But due to technical restrictions, neither the movements corresponding to consonants, nor the movements of the throat can be observed.
- An alternative approach uses ultra sound [Stone and Davis, 1995] [Stone and Lundberg, 1996]. This approach has the disadvantage that lips, tongue, and the lower jaw can not be tracked at the same time, and that the experiments are difficult as the head has to be immobilized and the device has to be calibrated. However, a high sampling rate (> 1000 frames/second) is possible.

The marginal notes in the form "⇒ X.Y" point to a section where the tool(s) related to the current paragraph is explained.

2 The X-ray image Database

The ATR X-ray film database, which is the largest X-ray database available for speech research [Munhall *et al.*, 1995], was digitized at IDIAP. This database was only in small parts and in a low quality available in a digitalized form, the original movies are available on a NTSC video-disc. The digitized data is stored in *Quicktime* format including the sound and as separate images in GIF format (approximately 100,000 frames, non-relevant borders are removed). However, for speech analysis purposes, a tape with the original sound should be used as sound on the video disc is modified in order to be compatible with the reduced frame rate of NTSC. ⇒B.1

Concerning the articulatory features, the data imposes certain restrictions: the larynx is invisible on most of the X-ray movies of the ATR database. The features we decided to track are: lips, teeth, palate, tongue, and the upper part of the throat (we did not achieve the tracking of the velum). The tracking of the tongue and the velum are the most challenging problems, as their contours are often not well defined and blurred, or even disappear due to shadowing and contact between them or other organs (*i.e.* between tongue and palate). The contours obtained for the tongue are therefore sometimes incorrect and/or incomplete.

During the digitization and inspection, the characteristics listed in table 1 were observed. The digitized database can be obtained form IDIAP.

	total	first	last	region	GIF
	of	image	image	of	data
	num.	in	in	interest	in
	frames	film	film	$x - y, x - y$	MB
Laval.24	3912	89	3861	30-0, 610-480	
Laval.25	3970	32	3938	25-0, 610-480	997
Laval.26	4180	85	4084	25-0, 585-480	976
Laval.27	4046	66	4044	25-0, 585-480	988
Laval.29	3997	94	3954	40-20,620-465	956
Laval.30	????	81	4008	15-20,600-480	886
Laval.31	4105	54	4015	15-30,595-480	753
Laval.32	4125	76	4109	55-0 ,605-470	982
Laval.33	4149	71	4084	65-40,605-470	892
Laval.43	4033	89	4033	40,20,605-480	885
Laval.44	4034	108	4018	55-10,580-470	842
Laval.45	4001	45	3994	25-40,605-480	883
Laval.48	4208	365	4060	45-65,600-440	650
Laval.49	4152	83	4069	45-10,615-465	895
Laval.50	4060	91	4044	45-30,615-460	893
Laval.55	3944	121	4065	40-20,600-470	881
Laval.56	4147	93	4045	45-60,560-455	687
Laval.73	3908	99	3509	70-20,580-445	703
Laval.74	3648	82	3506	70-40,580-440	638
Laval.77	4335	256	4238	55-10,575-475	897
Laval.78	4151	79	4014	50-35,570-450	777
Laval.79	4136	91	4086	40-25,570-460	855
Laval.80	4114	91	4038	45-25,575-465	815
Laval.81	2898	95	2874	45-25,575-455	557
MIT-KNS	1913	179	1867	80-0,590-480	362

Table 1: The contents of the X-ray database.

From this database, the film Laval 43 with 3944 frames showing the vocal tract was completely analysed and the results are available via the WWW page of the IDIAP vision group (URL: `http:www.idiap.ch/vision`). The results are stored in an ASCII file for each image. Each file contains the coordinates, respectively coordinates of control points of a c-spline, following the name of the respective articulator. For demonstration purposes, the results are incorporated in a MPEG movie obtainable at the URL given above. The pronounced phrases are in French with a Canadian accent:

1. Mes amis ont crée le theatre.
2. Le léopard réintegre sa cage.
3. La trahison frappa le truand.
4. Le coauteur est aneanti.
5. Le brouhaha incite au chahut.
6. Les enfants avancent cahin-caha.
7. Les embruns invitent à la rêverie.
8. C'est une clareté extraordinaire.
9. C'est une situation engagée.
10. Il a obtenu un met infect
11. La prohibition est une defense.
12. Les indo-européens importent.
13. Les documents indiens coincident.
14. En coopération on est (?) frére.
15. Louis est un enfant orgueilleux.
16. Le valet ignorait où j'étais.
17. Il est dans un état euphorique.
18. C'est une assemblée eucharistique.
19. Il a lu auprès de la fenêtre.
20. Le cadet emporta un ballon.
21. Il a un comportement ableur.
22. J'aimais obéir à mes parents.
23. A l'opéra on chante et on danse.
24. Martin identifie un object.
25. L'hindou orphelin oeuvre pour les pauvres.
26. Mon parain et mon époux importent.
27. Les assistant oublient leurs devoirs.
28. Quelqu'un use mon crayon et ma gomme.

29. On lui reprocha apprement.
30. Le défunt est enfin ammené.
31. Elle a formulé des voeux hâtif.
32. Quelqu'un heureux re *noise*
33. Mon maris est vers la porte
34. Il frappa au dessus du heurtoir.
35. Mais il *noise* russe.
36. Quelqu'un autorise *noise*
37. S'il a eu un.

3 Overview of the Segmentation

A sophisticated sequential procedure was developed as traditional methods (edge detection, contour methods, optical flow, and so on) were found to provide unsatisfactory results.

Some articulators are more distinct because of a comparably higher contrast - and consequently are easier to find in an automated way - than others. Therefore, these are located first, and then, using the benefits of image normalization and constraints on relative positions, more difficult articulators are tracked. In this sense, the following steps are performed, figure 1 shows a scheme of the procedure:

1. The X-ray images are filtered with a Gaussian filter and their histograms are then zero-normalized (see section 4.1).

2. The upper front teeth are located by the means of a pattern matching algorithm using distorted grey-level histograms (section 4.2).

3. Similarly, a reference point in the rear upper teeth is tracked. Teeth fillings are very robust objects for this purpose. They have a good contrast and are brighter than any surrounding objects. Furthermore, the shadow of the tongue does not alter their appearance much. Note that the position of the upper front teeth and the reference point define also the position of the palate.

4. The position of the head in images is normalized (translation and rotation) using the coordinates of the upper front teeth and some point of the rear upper teeth.

At the same time, the histogram of the images is normalized (*i.e.* non-linear deformations of the histogram are compensated for) using the distortion value obtained for the matching of the upper front teeth.

5. and **6.** The position of the lower front teeth and a reference point of lower teeth is determined. Again, fillings are very useful.

7. Edges are extracted from the normalized images obtained in **4.** by the means of a *Canny edge detector*.

8., **9.**, and **10.** The edges corresponding to the lips and the rear throat are located in the edge images extracted in **7.** by matching edge templates (see section 5). During the matching procedure, the velocity of edges is restricted in order to improve the robustness of the tracking algorithm against spurious edges and situations where the edges of an articulator are partly invisible. However, this causes the algorithm to loose track in the case of fast motions which occasionally occur (see the comments in section 7). To counterbalance this deficiency, the sequences are tracked forward and backward in time and the results of both sequences are joined together.

11. Edges are extracted from the normalized images obtained in **4.** by the means of a *Canny edge detector*, but negative gradients in the x-direction of the image are set to zero before the edges are extracted.

12. The front throat is tracked in the edge images obtained in step **11.** using the same approach as for the lips and the rear throat.

13. Image parts representing the upper and lower jaw with the tongue in an advanced and lowered, as well as in a back and high position, are subtracted from the normalized images.

14. This gives two series of images, to which a Canny edge detector is applied.

15. In these images, templates of the edges of the tongue are matched with one of the series of the images, depending on the position of the front throat (which is also the location of the rear tongue). Furthermore, possible matches are limited by an approach similar to the one used for the lips and the rear throat.

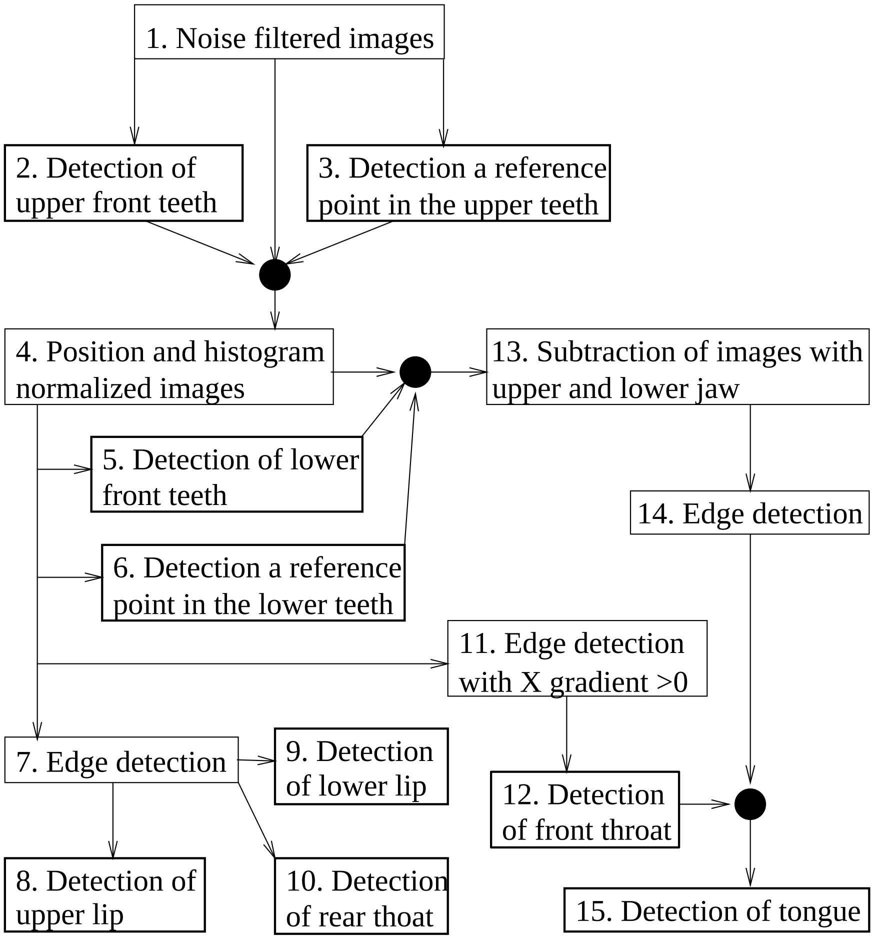

1. Noise filtered images
2. Detection of upper front teeth
3. Detection a reference point in the upper teeth
4. Position and histogram normalized images
13. Subtraction of images with upper and lower jaw
5. Detection of lower front teeth
14. Edge detection
6. Detection a reference point in the lower teeth
11. Edge detection with X gradient >0
7. Edge detection
9. Detection of lower lip
12. Detection of front throat
8. Detection of upper lip
10. Detection of rear thoat
15. Detection of tongue

Figure 1: Scheme for the detection of teeth, lips, front and rear throat, as well as the tongue

4 Localizing the Teeth

The X-ray films are affected by a variable illumination, caused by either an instable X-ray energy or a varying shutter speed of the camera. This effect can not be eliminated by standard linear histogram normalization. A standard pattern matching algorithm based on a simple distance measure between gray values is therefore likely to yield unsatisfactory results.

4.1 Zero-normalization of the Gray-level Histogram

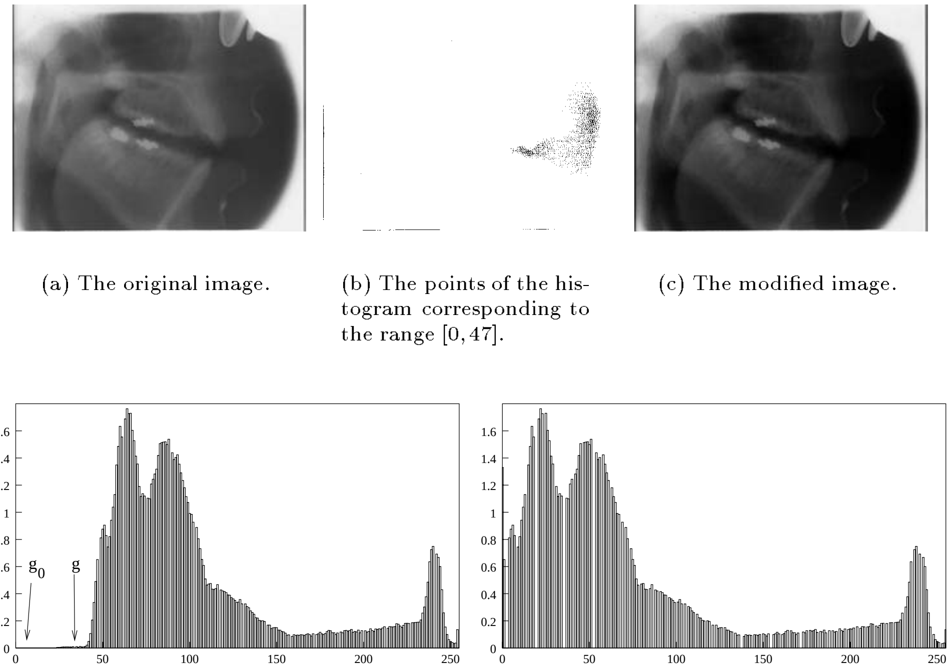
⇒B.2
⇒B.5

A first step to overcome this problem is to remove parts of the histogram: black parts with gray values smaller than some value g_0 correspond to noise and parts of the image that are of no interest (compare figures 2(a) and 2(b)). Furthermore, gray-values in the interval $[g_0, g]$ are almost not occurring, as shown in figure 2(d). With respect to these considerations, all pixels with gray values smaller than g are set to g. The modified image is then normalized to again have a gray-level histogram covering the whole possible range of values. The image 2(c) shows that the modified images have a higher contrast, but yield no other visible artifacts. Although the higher contrast is helpful for the human observer, this is not the main benefit of the approach, but to obtain the same gray level for the same object in all images.

As the cut-off value g is variable, it is chosen for each image separately according to the formula:

$$g = \operatorname*{argmax}_{K \in [g_0, 255]} \left(\frac{\sum_{i=g_0}^{K} \mathrm{hist}(i)}{N} < p \right). \qquad (1)$$

In this formula, $\mathrm{hist}(i)$ is the number of pixels with gray value i, N the number of pixels in the image, g_0 is chosen close to zero and to the left of the nearly empty part of the gray level histogram, and p the fraction of pixel values that are allowed in the interval[1] $[g_0, g]$. Figure 2(e) shows the histogram of the resulting image.



(a) The original image.

(b) The points of the histogram corresponding to the range [0, 47].

(c) The modified image.

(d) The histogram of the original image

(e) The modified histogram.

Figure 2: The lower part of the histogram of the images can be removed.

[1] For the ARTIST project the values $g_0 = 5$ and $p = 0.1$ appeared to be appropriate, but the results are not sensitive to the values of these two parameters. In the example given, $g_0 = 0$ would give a very similar result.

4.2 Compensating the Non-linear Distortion of the Histogram

⇒B.4 ⇒B.3

The zero-normalization of the histogram is insufficient, as the histograms are also subject to non-linear distortions. We compensated for this in the pattern matching process: the histogram of the template is modified during a comparison with some image location (for more details see [Thimm and Luettin, 1998]). The function projecting the histogram of the template to the histogram of the image can be regarded as a model of the illumination variation. The illumination changes are represented by a free variable which changes the shape of the mapping function. These changes are restricted according to three assumptions:

1. Black and white remain unchanged. Therefore, the lowest and highest intensities in the grey-level histogram will be mapped onto themselves.
2. Contrasts diminish or augment smoothly when the global illumination changes. Therefore, modifications of grey-levels must vary smoothly within neighboring intensity values.
3. The relative brightness of arbitrary objects must remain unchanged: if a certain spot in the image is brighter than another spot, it will remain brighter or, in the limit, assume the same intensity.

5 Tracking Articulators using Edges

5.1 The Basic Edge Template Matching Procedure

This section describes the basic matching procedure for edge templates with an edge image. The approach assumes that the object (*e.g.* the tracked articulatory feature)

- is exactly once present in an image,
- and it is invariably at the same place and has the same orientation and size, respectively all possible places, orientations, and sizes of the object are represented in the training data.

The procedure is, however, robust against **small** deformations, translations, and rotations, as well as occlusion and noise. It does not require that the edges corresponding to the object are connected.

The matching procedure uses edge images, as produced by a Canny edge detector (compare figure 3(a)). In a first step, edges are detected in all normalized images. From these edge images, representative edges that correspond to a certain object are extracted (see figure 3(b); how to select representative edges is discussed in section 5.2). Such images are called *state images* in the following.

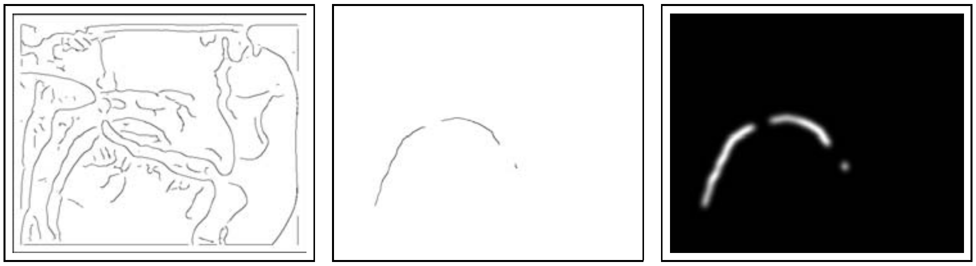
These state images are inverted and blurred by a Gaussian filter, resulting in so-called *matching* ⇒B.6 *images* $\mathcal{S}_i$ (for the ARTIST project, a variance $\sigma = 5$ pixel for the Gaussian filter is used, compare ⇒B.7 figure 3(c)). Both images are further associated with the same state which is proper to them. The variance of the Gaussian filter is directly related to the tolerance of the matching procedure towards the variability of an object.

The matching images $\mathcal{S}_i$ (the figure background is encoded as 0, the foreground as positive values) are used in the matching procedure. The score of a matching images $\mathcal{S}_i$ with respect to an image $\mathcal{X}$ is calculated as

$$\mathrm{score}(\mathcal{S}_i, \mathcal{X}) = \sum_{x,y} \mathcal{X}(x,y) * \mathcal{S}_i(x,y) \tag{2}$$

The matching image $\mathcal{S}_i$ with $i = \mathrm{argmax}_k(\mathrm{score}(\mathcal{S}_k, \mathcal{X}))$ with respect to some image $\mathcal{X}$ is defined as the optimal state and written as $\mathcal{S}(\mathcal{X})$.

Although equation (2) evokes a rather high computational complexity (per frame n multiplications of matrices in the size of the images, if n is the number of possible states), an implementation can



(a) The Canny-filtered image

(b) An example for an extracted edge or *state image*

(c) A matching image

Figure 3: Creation of a state (image).

be efficient: only non-zero parts of the matching image need to be considered in equation (2), which permits considerable optimizations. Furthermore, the tracking procedure described in section 5.3 limits the number of matching images for which the score has to be calculated to a small subset.

A simple tracking procedure would consist of calculating the optimal state for each image and an association with the contour of the corresponding optimal state image. As this procedure does not yield satisfactory results, temporal information is used to reduce the number of errors (see section 5.3).

5.2 Selection of State Images

In order to obtain good results with the matching procedure, the edges used for the state images should be selected consistently. In particular, the size of the selected edges and cut-off points should be similar.

As five articulatory features are tracked, five sets of state images are required. For the ARTIST project, the following choice for the size and cut-off points was done by the author (the number in parenthesis is the number of state images for the respective object):

Front Throat (83): from the lowest visible point up to the rim of the lower jaw.

Tongue (226): from the lowest visible point in the throat (thus including the front throat) forwards to tip - as far as visible. In some cases, the edge is invisible form the region of the rearrest teeth forward or for some intermediate parts.

Rear Throat (11): from the lowest visible point up to the place where, in the nasal cavity, the curve has an inclination of approximatively 60 degrees.

Lower Lip (105): from the point where it touches the lower front tooth to the recess point above the chin.

Upper Lip (75): from the point where it touches the tongue to the recess point below the nose.

Example choices are given in figure 4 by the bold lines. These edge images are generated without background subtraction, as it is used for the tongue (compare section 6.1).

The selection of a representative set of state images can be performed in an iterative manner: first, some edge images are selected randomly. Then, the image sequence is tracked using the corresponding set of matching images. If the feature is not well localized in some images, some of the respective edges are added to the set of state images and the whole procedure re-iterated.

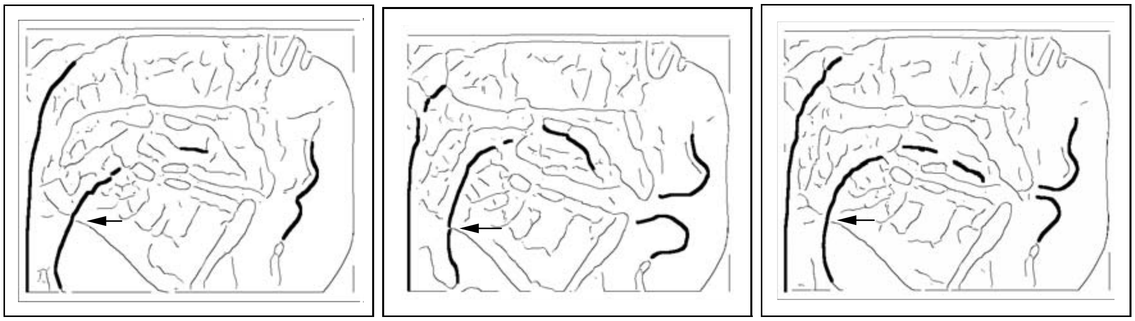


Figure 4: Example choices for selected edges: the bold lines are the selected edges. From the points marked by the flash downwards, the edge of the tongue is also the edge of the front throat.

5.3 Adding Temporal Information to the Tracking Procedure

The basic matching procedure described in section 5.1 can be disturbed by edges not belonging to the tracked object. This tracking procedure can be improved by using temporal information. *I.e.* under the assumption that the deformation of a feature between consecutive frames is small, the states that are reachable from a given state is a small subset of the whole training set. For this approach it is necessary to possess information on which state transitions are possible and which not. Whether or not a certain transitions is possible, is here estimated by calculating the some distance between states and then using only a percentage p of the transitions with the smallest distances.

Supposed that two edges $\mathbf{e}_1$ and $\mathbf{e}_2$ are approximated by the cubic splines $\mathbf{c}_1$ and $\mathbf{c}_2$, respectively. In this notation $c_{i,k}$ is a pair of x- and y-coordinates corresponding to the kth point of spline $\mathbf{c}_i$ with N points. These points are **not** the control points of the spline, which are never explicitly used in the following. See [Bartels *et al.*, 1998] for an introduction to splines. ⇒B.8

Then the distance between two edges is calculated by dividing the surface between the splines ⇒B.9 by the mean length of the splines. While calculation the surface between the splines, it has to be considered that generally the endpoints of the splines do not necessarily correspond to the same points of the feature. Two parts jutting out at the ends (F_1 and F_2 in figure 5), each delimited by a part of a spline, the connecting line between the endpoints, and the line between one endpoint and the closest point on the opposite spline, have to be neglected. Figure 5 illustrates the situation.

The surface F can be calculated by subtracting the surface of the surfaces F_1 and F_2 from the surface F_0 delimited by the splines and the lines connecting the endpoints ($\left|\left|P;Q;S\right|\right|$ is the surface

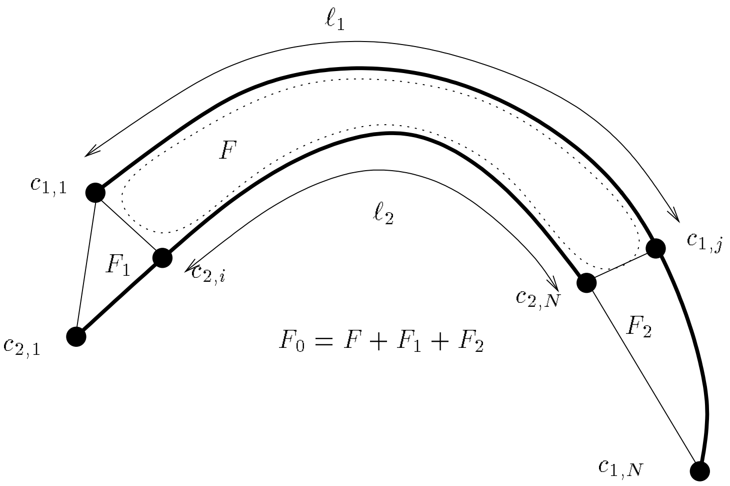


Figure 5: Measuring the distance between two splines.

of a triangle delimited by the points P, Q, and S):

$$\begin{aligned} F &= F_0 - F_1 - F_2 && (3)\\ &= \sum_{n=1}^{N-1}\Big(\big|\big|\mathbf{c}_{1,n};\mathbf{c}_{2,n};\mathbf{c}_{2,n+1}\big|\big| + \big|\big|\mathbf{c}_{1,n};\mathbf{c}_{1,n+1};\mathbf{c}_{2,n+1}\big|\big|\Big) - F_1 - F_2 \qquad \text{with} && (4)\\ F_1 &= \begin{cases} 0 & \text{if } a = c = 1\\ \sum_{n=1}^{c-1}\big|\big|\mathbf{c}_{1,1};\mathbf{c}_{2,n};\mathbf{c}_{2,n+1}\big|\big| & \text{if } c \neq 1\\ \sum_{n=1}^{a-1}\big|\big|\mathbf{c}_{1,n};\mathbf{c}_{1,n+1};\mathbf{c}_{2,1}\big|\big| & \text{if } a \neq 1 \end{cases}\\ F_2 &= \begin{cases} 0 & \text{if } b = d = N\\ \sum_{n=1}^{N-1}\big|\big|\mathbf{c}_{1,N};\mathbf{c}_{2,n};\mathbf{c}_{2,n+1}\big|\big| & \text{if } d \neq N\\ \sum_{n=1}^{N-1}\big|\big|\mathbf{c}_{1,n};\mathbf{c}_{1,n+1};\mathbf{c}_{2,N}\big|\big| & \text{if } b \neq N \end{cases} \end{aligned}$$

where a, b, c, and d are selected so that

1. $\mathbf{c}_{1,a}$ is the point on $\mathbf{c}_1$ that is closest to $\mathbf{c}_{2,1}$,
2. $\mathbf{c}_{1,b}$ is the point on $\mathbf{c}_1$ that is closest to $\mathbf{c}_{2,N}$,
3. $\mathbf{c}_{2,c}$ is the point on $\mathbf{c}_2$ that is closest to $\mathbf{c}_{1,1}$, and
4. $\mathbf{c}_{2,d}$ is the point on $\mathbf{c}_2$ that is closest to $\mathbf{c}_{1,N}$.

Note, that if $a = b$ or $c = d$, then $F = 0$, and that either a or c is equal to one, as well as that either b or d is equal to one. ⇒B.10

The distance $D_{i,j}$ between the splines is then defined as the surface F divided by the accumulated lengths of the parts of the splines $\mathbf{c}_1$ and $\mathbf{c}_2$ that delimit F:

$$\begin{aligned} D_{n,m} &= \frac{F}{\ell_1 + \ell_2} \qquad \text{with} && (5) \\ \ell_1 &= \sum_{i=a}^{b-1} |c_{n,i} - c_{n,i+1}| \qquad \text{and} \\ \ell_2 &= \sum_{i=c}^{d-1} |c_{m,i} - c_{m,i+1}| \end{aligned}$$

Furthermore, D is augmented by an initial state $\mathcal{S}_0$ for which $D_{0,j} = 1$.

To obtain finally the state transition matrix $\mathcal{T}$, a minimal limit $\mathcal{L}_i$, that is proper to each state $\mathcal{S}_i$, is searched, so that for p percent of the transitions $\mathcal{L}_i > \mathcal{D}_{i,j}$ is true. Then, $\mathcal{T}_{i,j}$ is defined as 1, if $\mathcal{L}_i > \mathcal{D}_{i,j}$ and 0 otherwise. For the ARTIST project, typically $p = 30\%$ was chosen.

5.4 Tracking a Feature

The tracking procedure is an iterative process, in which the selection of a set of possible states by means of the transition matrix $\mathcal{T}$ alternates with the calculation of the optimal state with respect to this selection. More precisely, the score of $\mathcal{S}_i$ with respect to matching image $\mathcal{X}_t$ and the optimal state $\overrightarrow{\mathcal{S}}(\mathcal{X}_{t-1})$ for the previous frame is calculated using the following formula:

$$\overrightarrow{\text{score}}(\mathcal{S}_i, \mathcal{X}_t) = \begin{cases} \text{score}(\mathcal{S}_i, \mathcal{X}_t) & \text{if } \mathcal{T}_{j,i} = 1 \text{ with } \overrightarrow{\mathcal{S}}(\mathcal{X}_{t-1}) = \mathcal{S}_j \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Whereas for the first image $\mathcal{X}_1$ in the sequence, the preceding state is defined as S_0 (or equivalently: $\overrightarrow{\mathcal{S}}(\mathcal{X}_0) \equiv \mathcal{S}_0$). Then, the optimal state $\overrightarrow{\mathcal{S}}(\mathcal{X}_t)$ is defined as the state with the maximal score.

The optimal state sequence $\overrightarrow{\mathcal{S}}$ is then defined as $(\overrightarrow{\mathcal{S}}(\mathcal{X}_1), \overrightarrow{\mathcal{S}}(\mathcal{X}_2), \ldots)$.

6 Joined Forward-Backward Tracking

One important assumption in section 5.3 is, that objects move slowly (*i.e.* only transitions permitted by the states transition matrix are performed). Although this is true most of the time, there are exceptions: tongue and lips can move so fast that they assume almost extreme positions in consecutive frames. Section 7 gives some quantitative analysis of maximal velocity and acceleration.

However, before and after those high-velocity movements, the velocity and acceleration of the respective articulators is low and fulfills for a certain time the assumption. The following approach reduces the severity of the assumption on the velocity of the movement according to this observation:

1. Calculate the forward tracking sequence $\overrightarrow{\mathcal{S}}$ as in section 5.3.
2. Calculate the backward tracking sequence $\overleftarrow{\mathcal{S}}$ in a similar manner, except for using a score that restricts the states in a backward manner:

$$\overleftarrow{\text{score}}(\mathcal{S}_i, \mathcal{X}_t) = \begin{cases} \text{score}(\mathcal{S}_i, \mathcal{X}_t) & \text{if } \mathcal{T}_{j,i} = 1 \text{ with } \overleftarrow{\mathcal{S}}(\mathcal{X}_{t+1}) = \mathcal{S}_j \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

3. Join state sequences $\overrightarrow{\mathcal{S}}$ and $\overleftarrow{\mathcal{S}}$ to form the forward-backward sequence $\overleftrightarrow{\mathcal{S}}$ (see also figure 6):

$$\overleftrightarrow{\mathcal{S}_i} = \begin{cases} \overrightarrow{\mathcal{S}_i} & \text{if } \overrightarrow{\text{score}}(\overrightarrow{\mathcal{S}_i}) \geq \overleftarrow{\text{score}}(\overleftarrow{\mathcal{S}_i}) \\ \overleftarrow{\mathcal{S}_i} & \text{otherwise} \end{cases} \tag{8}$$

This approach can be used for the lips as well as for the rear and front throat. The tracking of the tongue is discussed in section 6.1.

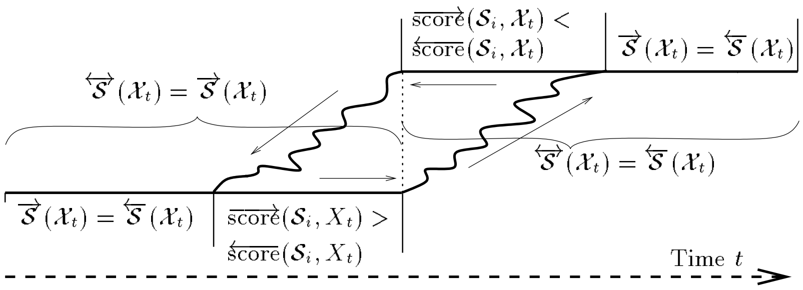
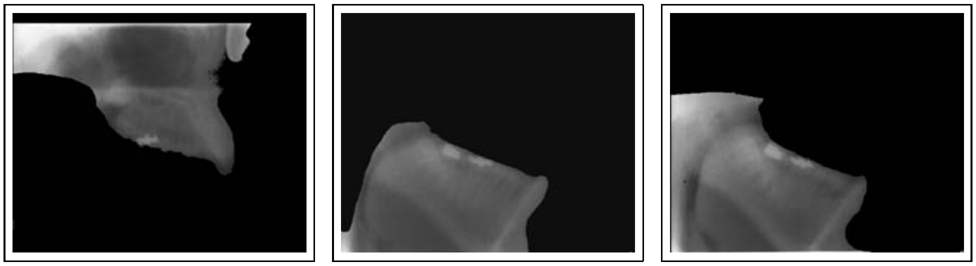

$\overrightarrow{\text{score}}(\mathcal{S}_i, \mathcal{X}_t) <$
$\overleftarrow{\text{score}}(\mathcal{S}_i, \mathcal{X}_t)$
$\overleftrightarrow{\mathcal{S}}(\mathcal{X}_t) = \overleftarrow{\mathcal{S}}(\mathcal{X}_t)$
$\overleftrightarrow{\mathcal{S}}(\mathcal{X}_t) = \overrightarrow{\mathcal{S}}(\mathcal{X}_t)$
$\overleftrightarrow{\mathcal{S}}(\mathcal{X}_t) = \overleftarrow{\mathcal{S}}(\mathcal{X}_t)$
$\overleftrightarrow{\mathcal{S}}(\mathcal{X}_t) = \overrightarrow{\mathcal{S}}(\mathcal{X}_t)$
$\overrightarrow{\text{score}}(\mathcal{S}_i, X_t) >$
$\overleftarrow{\text{score}}(\mathcal{S}_i, X_t)$
Time t

Figure 6: Joined forward-backward tracking: the straight lines represent the correct sequence resulting from this approach, the zig-zag lines the wrong parts of the forward, respectively backward, sequence.

6.1 Tracking the Tongue

The tongue causes another problem: it is often hidden by the jaws, which means that the contour of the tongue is not or only hardly visible. Sometimes even a human observer is unable detect the precise location of the tongue. The tracking procedure is consequently augmented by background subtraction. The background subtraction is, however, a little bit more complex than in standard applications. As the upper jaws is not moving, it can be directly be subtracted (figure 7(a)). However, the background image with the lower jaw has to be oriented according to the current position of the jaw, which is known from the tracking of the lower teeth. Furthermore, two background images of the lower jaw with different tongue positions are required (figures 7(b) and 7(c)), as the background image necessarily shows the tongue, and therefore the contour of the tongue will disappear if the tongue in the image is at the same position as in the subtracted background image. Therefore, according to the position of the front throat, which can be tracked more easily, one of the two different background images of the lower jaw are used.



(a) The upper jaw

(b) The lower jaw with the tongue advanced

(c) The lower jaw with the tongue drawn back

Figure 7: Background images subtracted from images before the edge detection for the tongue.

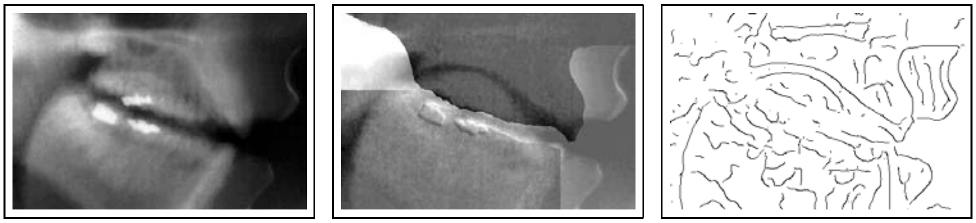
The for this approach required edge images are created in the following way:

1. From one normalized image which shows the tongue in a low position, everything but the upper jaw is removed (set to zero; see figure 7(a)).
2. Similarly, from an image where the tongue is in an advanced, lowered position, everything but the lower jaw is removed (figure 7(b)), and
3. from an image where the tongue is in a retracted, high position, everything but the lower jaw is removed (figure 7(c)). ⇒B.11

4. From each image, the image with upper jaw is subtracted. Then, depending on whether the position of the rear throat is in an advanced or retracted position, the partial image with the lower jaw and the tongue in the opposite position is also subtracted.

5. Finally, the edges in these images are detected.

Figure 8 shows an example for this process: from the original in image 8(a) the jaws are subtracted, resulting into the image 8(b). It can be seen that the image region corresponding to tongue is more uniform and the fillings in the upper teeth disappeared. In consequence, the edge of the tongue in the region of the mouth is nicely detected by a Canny edge detector (image 8(c)).



(a) A part of a normalized X-ray image.

(b) The image with the upper and lower jaw subtracted (the contrast is enhanced for better visibility).

(c) The corresponding edge image. The contour of the tongue shows up nicely.

Figure 8: An example for the subtraction of the jaws followed by an edge detection as used for the tracking of the tongue.

For the tracking procedure, a set of state is then prepared and the corresponding matching images are created as it is described in section 5.1. The tracking procedure is the same, except for using the special edge images and selecting possible states also by limiting the distance between the previously detected front throat and the contour of the tongue in the state image. The distance between all front throat state image and tongue state images is calculated in the following way:

- If there are horizontal lines, on which both, a point of the tongue edge and the front throat edge exists, then the maximal Euclidean distance between such points defines the distance.

- If no such point exist, the highest point of the front throat is extrapolated by a line with an inclination of 2. Then the horizontal distance of the lowest point of the rear part of the tongue to the extrapolating line defines the distance.

⇒B.12

During the match of a state with an edge image, again the assumption of slow motion is used and ⇒B.13
counterbalanced by joining forward and backward tracking results. Furthermore, the distance of the ⇒B.14
position of the previously detected front throat is used to restrict the matched states for the tongue ⇒B.15
to typically 30% of all states for the tongue.

7 Remarks

The frame rate of 50 images per second is rather low as compared to the maximal velocity observed for the different articulators. For example for the lower front tooth velocities of 10 to 40 pixel/frame (which corresponds roughly to 17 to 68 cm per second) were encountered. This is high as compared to a maximal observed distance of 76 pixel between arbitrary positions. Furthermore, as the lower jaw does not move visibly in the frames before and after the event, the movement jumps from zero to very high velocity and acceleration and then rapidly comes to a stop. This causes algorithms like *Kalman filters* [Blake *et al.*, 1995] or the *condensation algorithm* [Isard and Blake, 1996] to loose track, as the motion does not obey the assumption of regular motion (due to the low frame rate).

Similarly, for the lower lip and the tongue speeds above 20 pixel per frame (appr. 35 cm per second) are observable.

8 Conclusion

We proposed a contour tracking algorithm that can be applied to objects of which the general position is known (or limited to a small number of positions), but that are subject to non-linear, spontaneous deformations. The approach is very robust to noise and occlusion, and is based on the assumption that deformations, with the exception of rarely occurring spontaneous deformations, are slow. The approach associates contours with states, and motion, respectively object deformations with state transitions. During the tracking procedure, the state transitions are restricted to those associated with small movements, which are determined by calculating the distance between splines approximating the contours. Spontaneous deformations are dealt with by joining the state sequences obtained by tracking the image sequence forward and backward in time.

The segmentation of the vocal tract is done in an iterative manner: first the teeth were tracked by the means of two specialized histogram normalization techniques combined with a pattern matching algorithm, which also gives the position of the palate. Then by the means of a position normalization of the vocal tract, the movements of the lips, the tongue, and the throat are restricted to deformations. Based on this, the throat and the lips are tracked. Finally, the configuration of the front throat is used to restrict the number of deformations of the tongue, and a specialized background subtraction technique is used to enhance the contrast of the tongue.

We assume that the precision of the tracking procedure is sufficient for speech analysis purposes, although a quantization of the error is infeasible in practice. However, we estimate that the contours of the lips, the front, and rear throat are located in more than 98% with a sufficient precision. Similarly, the position of the tongue should be sufficiently precise in 95% of the frames. We further estimate the error for the position of the teeth below 7 pixel (approximatively 2mm).

With respect to the obtained results, the low quality of the X-ray database, and the difficulties proper to this type of data, the method can be estimated to be very robust.

A Assembling the results

Before executing the following steps, an edge template and corresponding spline should be created for the palate using a normalized edge image. ⇒B.18
⇒B.19

B Software

B.1 Extracting images and sound from a *Quicktime* film

Command: `unqt.pl` [*file.mov*]

Output: a sound file named *file*.raw (16 bit stereo linear at 22050 samples/second) and PPM files *file.number*.ppm (number starting at zero).

Remark: the program is very "beta" and likely to work only for the films of this database.

B.2 Extracting a Pattern

Usage: `calc_profile.m` [*GIF image*] [*x coordinate*] [*y coordinate*] [*x size*] [*y size*] [*Output file*]

This script extracts a portion of the *GIF image* and writes it in *Octave* formate to the file *Output file*. This file defines values for the matrix `profile`, as well as the variables `prof_xsize` and `prof_ysize`.

B.3 Tracking of a Pattern

Usage: `search.m [`*`x coordinate`*`] [`*`y coordinate`*`] [`*`profile file`*`] [GIF image files]`

Arguments: the x- and y-coordinates indicate where the pattern is approximatively in the first image. From this point, the pattern is searched a certain region (x and y coordinates plus/minus `max_x_distance = max_y_distance = 15` as initially configured in the script). For the following images, the central point is chosen to be the point with the smallest error for previous image.

Output goes to *standard output*: the names of the image and the x- and y-coordinates with the smallest error. The output of this script includes the whole path of the image, which should be removed after having controlled the results (see section C.4) and before using the results in other programs

B.4 Correction of the Distorted Illumination

Command: `correct_alpha.m [`*`coordinate file`*`] [`*`profile`*`]`

Arguments: in the *coordinate file*, for each GIF image, its filename, x- and y-coordinates of the location where the *profile* of some object (e.g. (the filling of) a tooth) is found.

Output: GIF images with the same name as the originals in the sub-directory filteredalpha.

B.5 Zero-normalization of images

Command: `zero_profile.m {Gif files}`

Arguments: Gray-leveled GIF files.

Usage: the script may require the configuration of the maximal percentage of pixels in the region between g_0 and g and the value of g_0 (see section 4.1).

Output: GIF files, with the same names in the local directory. The original images may be overwritten if they are in the directory where the script is started.

B.6 Creating State Images for Rear and Front Throat, as well as Palate and Lips

Matlab function: `create_map(`*`File`*`)`

Argument: The filename of a Canny filtered GIF image (the edges coded in black).

Requirements: functions `label_edges` and `mark_edge` in `thimm/Mathlab` as well as the *MatLab tools for multi-scale image processing* (see [Simoncelli, 1998]). The functions call further the scripts `make_cspline_label.m create_MAP.m`, calculating a spline approximating the marked objects (the generated files with the names *F*.ppm show the result. They can be deleted after inspection (compare section B.8).

Output: two GIF files:

1. a file with the selected edges (the path and all characters before the token `Laval` are removed from the filename *File* to form the new one. Example: `Laval.43-1999.gif`).
2. a file with the blurred and inverted image with the same name as the previous file except for the prefix `map` (example: `mapLaval.43-1999.gif`).

Marking edges: clicking with left mouse button on an edge selects it. The middle mouse button removes parts of (non-)marked edges, which permits to cut edges into pieces. This is useful when two edges corresponding to different objects are touching each other. The right mouse button removes non-marked edges from the image, stores the image, and starts the creation of the corresponding spline and map image.

B.7 Creating State Images for the Tongue

Matlab function: `create_map_tongue(`*`File`*`)`

Argument: The filename of a Canny filtered image (the edges coded in black).

Requirements: as in B.6, except for the script `create_MAP.m` instead of `create_map.m`

Output: two GIF files:

1. an image with the selected edges.
2. an image with the edges blurred orthogonally to the selected edge with the prefix `MAP`.

Marking edges: see section B.6.

B.8 Creating State Splines

Usage: The script `make_cspline_label_[`*`articulator`*`].m` is a script proper to each articulator. Main differences between the versions are the number of supporting points in the spline and the default location.

Argument(s): GIF file(s) with the object only as a black line.

Output: Per file, a spline in a file with an additional suffix `.spl` and a file with the suffix `.spl.ppm`. The later is only intended for control purposes and can be deleted after inspection. If the spline does not coincide with the object, try to change the initial default locations of the support points of the spline (in the middle of the program) and restart it.

B.9 Calculating the Distance Between two Splines

Overview: the approximation is done using the Octave scripts `make_cspline.m` which is proper to each object. *I.e.* depending on the script called, the number of points, the selection of the end points, and the initial positioning of the intermediate points differ.

Usage: `make_cspline.m [`*`GIF file`*`]`

The argument of these scripts is a GIF image showing the edge in black.

Output: The script creates two files: *file*`.spl` containing the points of the spline and *file*`.ppm` which is designed for the user having a control on the result (it shows an extraction of the edge image and the spline and can be removed after inspection. Note that the endpoints of the spline have to be tripled to be a correct c-spline.

B.10 Calculation of the Distance Matrix for Splines

Usage: `/ARTIST/states/calc_distances_for_splines.m [`*`Spline file`*`] [`*`output file name`*`]`

Arguments:

1. a file with the name of a GIF file with the edge and the coordinates of the associated spline.
2. The name where the result will be stored in OCTAVE format (string array `states` and distance matrix `smatrix`).

B.11 Subtracting jaws

Command: `subtract_jaws.pl` *`[file with lower teeth reference coordinates] [file with reference coordinates for lower front teeth]`*

Inputs: The two arguments files with reference coordinates for the rear and front lower teeth and normalized GIF images (mentioned in the argument files).

Usage: Before the script can be used, it needs some configuration: the name and reference coordinates of the images with the upper and lower jaw.

Output: For each image mentioned in the argument files, two GIF images with edges are created. The file with the prefix `RE` contains the edges for the case when the subtracted image of the lower jaw shows the tongue in a rear, upper position, and with the prefix `SE` for a front, lower position.

B.12 Calculating the distance between tongue and rear throat

Usage: `calc_distance_throat_tongue.m [front throat list] [tongue list] [distance file]`

Input: two files with the list of edge files for the front throat and tongue. The images listed in these two files should be in GIF format and contain extracted edges (compare section 5.2).

Output: a file named according to the third parameter in octave format. It defines the two string arrays (`front_throat` and `tongue`) with the image file names from which the data is generated and the distance matrix (`dist`). Note that the result files are stored under the name of the image file plus the postfix `.stats_`. See section C on how to display the results.

B.13 Tracking the Tongue

Usage: `calc_stats_extr_orient` *`[distances throat-tongue] [file with results for front throat] [fraction] [distances tongue-tongue] [directory for results]`*

Arguments: The *distance between the front throat and the tongue* is the name of the output file of `calc_distance_throat_tongue.m`, *fraction*$\in (0,1)$ a number, designating the fraction of the state images to be tested against the current image (those with the smallest distance are preferred). The results are stored in the directory named as forth argument. The images are tracked in the sequence they occur in the file listing the results for the front throat (2nd argument).

Usage: The program should be called twice. With directory (5th argument) named `forward`, respectively `backward`, and the results for the front throat in forward order, respectively backward order.

B.14 Joining Forward and Backward Tracking

Command: `combine_forward_backward.pl [forward] [backward] [result]`

Arguments: `forward` and `backward` are the directories where the results of the forward and the backward path are stored. The joined sequence is stored in the directory `result`. However the resulting files still have the postfix '_', in order to permit to see changes (using the command `show_stats_diff.pl`, see section C.2). The script `update.pl` permits to incorporate the results definitively (see section B.15).

B.15 Updating the results

Command:
```
update.pl [directory]
```

Usage: this script compares all files in the directory given as argument for whether the first string in files with the same name except for a terminating '_' is the same or not. if yes, the file with the terminating underscore is removed, otherwise it is used to replace the file without.

B.16 Canny edge detection

Command:
```
giftopnm <[infile]| imgPnmToFlt| imgSmooth 5| imgCanny | imgEdgeSynth
0.001 0.4 | ppmtogif > [outfile]
```

Usage: *infile* is a position and histogram normalized X-ray image, whereas *outfile* is a GIF image showing the edges (usually having the prefix `ALLE`). The commands are either part of the PBMPLUS or the IMG* package [Winder,].

B.17 Calculation of Edges

Command: Almost identical as the command in section B.16, `imgCanny` is replaced by `imgXCanny`.

B.18 Collecting the data

Command:
```
collect_results.pl
```

Usage: No arguments are used, all sources of information are specified in the script and need to be configured there.

Output: In the place where this script is called, a subdirectory named `RESULTS` is created. In this directory, for each image *IMAGE*`.gif` in a file *IMAGE*`.res` the wanted information are stored in ASCII format. Each line contains a tag and is followed by space separated numbers. These numbers represent either a point (x- and y-coordinate), which is the case for the tags:

- upper front tooth
- upper front tooth

The other object are followed by a list of points (pairs of x- and y-coordinates), representing the control points of a *c-spline* (see [Bartels *et al.*, 1998]) with the start and end point tripled (in order to obtain a spline that actually starts at the given point). The tokens are:

- upper teeth
- lower lip
- palate
- front throat
- tongue
- rear throat
- upper lip
- lower teeth

The lists that are read by `collect_results.pl` are usually created by first calling `update.pl` (section ??) and then `read_results.pl` (section C.3).

B.19 Displaying the data

Command:
```
show_all_features.m [result files]
```

Usage: for each result file (as produced by `collect_results.pl`), splines and points are drawn into the normalized image. The name of the normalized image is obtained by replacing the whole path in the name of the result file by `Normalized_images/N` (configure the variable `image_prefix` if needed). The configurable integer variable `reduce` defines the amount by which the image size is reduced.

Output: The script creates a subdirectory `MOVIE_TMP` with GIF images.

B.20 Creating a movie

Command: `make_mov` [*file with list of images*] [*file with a list of sound files*] [*output file*]

Usage: The source code (*i.e.* the module `make_mov.c`) needs a lot of configuration. This is chaotic, unsupported, badly documented, and hacked code. Usage on your own risk.

Output: a `Quicktime` movie - hopefully.

C Controlling the Results

C.1 The Tongue

C.2 Displaying differences between new and old results

Command: `show_stats_diff.pl` [*Result files*]

Arguments: the names of result files (without the terminating _).

Usage: the command creates a list of born shell commands to be piped, for example, into `do_parallel.pl` (see section D.6). The script creates a directory named `DIFFS`, in which the commands create images, showing the X-ray images with lines showing the detected contour in the file with and without the terminating '_' for which the files yield a different result.

C.3 Reading the results to on list

Command: `read_results.pl` *directory*

Usage: The result files in `directory` with a terminating `.stats` are read, ordered and wrote to standard out.

C.4 Position of patterns

The position of a pattern in a set of images can be efficiently performed on a list (*i.e.* a file) with a column of file names, followed by a column of x- and a column of y- coordinates) (as generated by the program `search.m`). This is used for

- a point in the upper teeth,
- the front teeth.
- a point in the lower teeth,

C.5 Controlling the position of patterns

Command: `extract_profiles.pl` [*x size*] [*y size*]

Usage: The command reads from standard in a list of files with coordinates. Then, it extracts small windows according to the arguments and the coordinates associated with an image, and glues them together in a big image, which is then displayed. Note that the length of the list should be not superior to 400 (use something like `tail` *+XXX File* | `head` *-YYY* | extract parts of a file).

D Used software packages

D.1 Netpbm

D.2 IMG* Image Processing Toolset and C Library

This package is freely available, although it is under copyright by Simon A.J. Winder.

The following filters were written by means of the included library:

- `imgCropCenter` crops an image relative to its center.
- `ThreshNullifiy` sets all values that are in a certain range to zero.

D.3 Octave

A multitude of scripts were written in Octave. Further, the following packages were added:

- **Strategy Simplex**. This is adapted from a source written for *Matlab* by Zeljko Bajzer `bajzer@mayo.edu` and Ivo Penzar `penzar@mayo.edu` at the Mayo Clinic and Foundation, Rochester, Minnesota, USA (installed in `/homes/thimm/octave/simps`).
- The **Image Processing Toolbox for Octave** written by Ariel Tankus `arielt@math.tau.ac.il` (with minor bug-fixes, see `/homes/thimm/octave/PNM`).

D.4 Auxiliary software

D.5 Gaussian Filter

Command: `imgSmooth [`*variance*`]`

Documentation: see the documentation of *IMG* Image Processing Toolset and C Library* [Winder,].

D.6 Execute shell commands in parallel

Command: `do_parallel.pl [-p` *priority*`] [`*parallel*`]`

Arguments: *priority* a number between 0 and 19 defining the priority of the subprocesses. *Parallel* defines how many commands are executed in parallel.

Usage: the command reads from standard in a list of commands (each line is associated with an independent command line in born shell syntax) and when the end of file is read distributed in the given number of processes.

References

[Bartels *et al.*, 1998] Richard H. Bartels, John C. Beatty, and Brian A. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling.* Morgan Kaufmann Publishers Inc., September 1998.

[Berger and Laprie, 1996] M.-O. Berger and Y. Laprie. Tracking articulators in X-ray images with minimal user interaction: Example of the tongue extraction. In *Proceedings of IEEE International Conference on Image Processing, Lausanne, Switzerland*, volume II, page 289ff, September 1996.

[Blake *et al.*, 1995] Andrew Blake, Michael Isard, and David Reynard. Learning to track the visual motion of contours. *Artificial Intelligence*, 78:101–133, 1995.

[Davis *et al.*, 1996] Edward P. Davis, Andrew S. Douglas, and Maureen Stone. A continuum mechanics representation of tongue deformation. In H. Timothy Bunnell and William Idsardi, editors, *Proceedings ICSLP96: Fourth Conference on Spoken Language Processing*, volume 2, pages 788–792, New Castle, Delaware, 1996. Citation Delaware.

[Isard and Blake, 1996] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *ECCV'96*, 1996.

[J.L. Barron and Fleet, 1994] S.S. Beauchemin J.L. Barron and D.J. Fleet. On optical flow. In *Int. Conf. on Artificial Intelligence and Information-Control Systems of Robots*, pages 3–14, Bratislava, Slovakia, September 12-16, 1994.

[Laprie and Berger, 1996] Y. Laprie and M.O. Berger. Towards automatic extraction of tongue contours in x-ray images. In *Proceedings of International Conference on Spoken Language Processing 96*, volume 1, pages 268–271, Philadelphia (USA), October 1996.

[Luettin and Thacker, 1997] Juergen Luettin and Neil A. Thacker. Speechreading using probabilistic models. *Computer Vision and Image Understanding*, 65(2):163–178, 1997.

[Munhall *et al.*, 1995] K.G. Munhall, E. Vatikiotis-Bateson, and Y. Tokhura. X-ray film database for speech research. *Journal of the Acoustical Society of America*, 98(2):1222–1224, 1995.

[Simoncelli, 1998] Eero P. Simoncelli. Matlab source code for multi-scale image processing, 1998. Available at http://www.cns.nyu.edu/˜eero/. email: eero.simoncelli@nyu.edu.

[Stone and Davis, 1995] M. Stone and E.P. Davis. A head and transducer support system for making ultrasound images of tongue/jaw movement. *J. Acoust. Soc. Am.*, 98(6):3107–3112, 1995.

[Stone and Lundberg, 1996] M. Stone and L. Lundberg. Three-dimensional tongue surface shapes of english consonants and vowels. *J. Acoust. Soc. Am.*, 99(6):1–10, 1996.

[Thimm and Luettin, 1998] Georg Thimm and Juergen Luettin. Illumination-robust pattern matching using distorted color histograms. IDIAP-RR 9, IDIAP, Rue de Simplon 4, CP 592, CH-1920 Martigny, Switzerland. Email: Thimm@idiap.ch, June 1998.

[Winder] Simon A.J. Winder. IMG* image processing toolset and c library. Available at http://www.il.debian.org/Packages/unstable.non-free/graphics/imgstar.html.