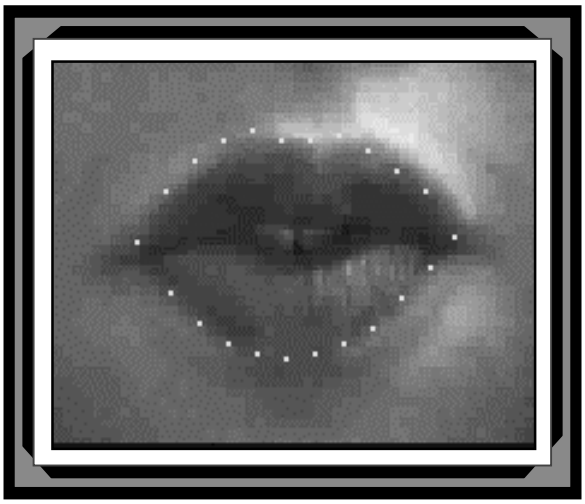
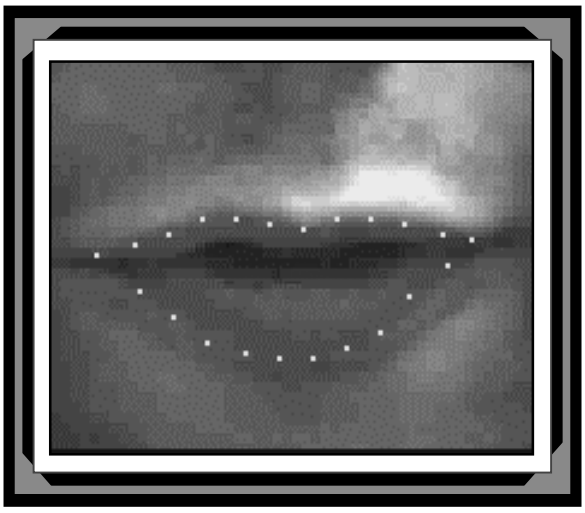
Deformable Models





SAMBA/24/00

Line Eikvil

November 2000

© Copyright Norsk Regnesentral

Norsk Regnesentral
ANVENDT DATAFORSKNING

NR-notat/NR Note

Tittel/Title: Deformable models

Dato/Date: November
År/Year: 2000
Notat nr: SAMBA/24/00
Note no:

Forfatter/Author: Line Eikvil

Sammendrag/Abstract:

The main purpose of this study was to implement a method based on deformable models suitable for tracking objects in video. The main idea for deformable models is to encode a variety of shape deformations while maintaining the inherent object structure in the model. Such models have been used in a wide range of applications including image/video database retrieval, tracking restoration etc.

The report gives a brief overview of deformable models, and describes in some more detail one class of deformable models, called active shape models. Active shape models are prototype-based deformable templates, where shape variation is extracted from a training set by applying principal component analysis to point distribution models, rather than handcrafting a priori knowledge into the model.

For the experiments the active shape models described in a work by Luettin et al. were chosen, and the methods used for training and matching, using this active shape model, are described in detail. The active shape model is a flexible model that should be easy to adapt to other problems. Training a new model does however include manual labelling of the points in a reasonably sized training set, and this process can be tedious.

Emneord/Keywords: Deformable models, active shape models, lip-tracking

Tilgjengelighet/Availability: Open

Prosjektnr./Project no.: 1030

Satsningsfelt/Research field: Image and video analysis

Antall sider/No. of pages: 31

Norsk Regnesentral / Norwegian Computing Center
Gaustadalléen 23, Postboks 114 Blindern, 0314 Oslo, Norway
Telefon 22 85 25 00, telefax 22 69 76 60

© Copyright Norsk Regnesentral

Deformable Models

Line Eikvil

November, 2000

Contents

Chapter 1

Introduction

This report describes an experiment with deformable models applied to the problem of tracking. The study of deformable models started with an attempt to solve shape matching, where a given shape needs to be located in an image using correlation-based matching. The correlation-based matching has limited applications because objects are not always undergoing rigid transformations in real life and they are also expected to deform to the varying imaging conditions, sensor noise and occlusions. Hence, deformable models became increasingly popular.

The main idea for deformable models is to encode a variety of shape deformations while maintaining the inherent object structure in the model. Such models have been used in a wide range of applications including image/video database retrieval, tracking restoration etc. Different models have been defined, where the differences lie in the specification of the model, the definition of the set of rules, and the recovery process which performs the segmentation.

In the following an overview of deformable models is given in Chapter 2. Then, in Chapter 3 we will describe one specific deformable model based on active shape models applied to the problem of lip-tracking. The results of some experiments with this method are reported in Chapter 4, while a summary of the study is given in Chapter 5.

Chapter 2

Deformable models

A deformable model can be characterised as a model, which under an implicit or explicit optimisation criterion, deforms the shape to match a known object in a given image. The model is active in the sense that it can adopt itself to fit the given data. It is a useful shape model because of its flexibility and its ability to both impose geometrical constraints on the shape and to integrate local image evidences. There has been a substantial amount of research on deformable models in recent years.

In [7] Jain et al. give an overview of the work in this area. They partition the work on deformable models into two classes; **free-form** models and **parametric** models. The free-form models can represent any arbitrary shape as long as some constraints like continuity, smoothness etc. are satisfied. These models are often called active contours. The other class, parametric deformable models, encode a specific shape and its variation where the shape can be characterised by a parametric formula or a prototype and its deformation modes.

- **Free-form deformable models** (active contours)
- **Parametric deformable models**
  which can be of two kinds:
  - *Analytical deformable templates*
  - *Prototype-based deformable templates*

2.1 Free-form deformable models

Free-form deformable models, also called **active contour models**, assume very little structure about the object shape except for some regularisation constraints like continuity and/or smoothness of the boundary. Such a free-form model can be deformed to match salient image features like lines and edges using potential fields (energy functions) produced by those features. Since there is no

global structure for the template, it can represent any arbitrary shape as long as the regularisation requirements are satisfied.

Kass, Witkin and Terzopoulos [8] introduced one of the earliest and most popular free-form deformable models; the **snake model**. A snake is a geometrical curve which approximates image contours through energy minimisation [6]. It behaves like an elastic rope that wriggles towards the contour or that slides down the potential hill. The internal forces keep the shape and ensure the spatial and temporal continuity, while the external forces pull and guide the snake in an interactive and dynamic iterative process.

The snake uses only the information from its local surroundings, whereas information about the global shape is missing, and the implementation of the original snake model is vulnerable to its initial position and image noise. Thus, the snake has to be placed in proximity to the contour and can be trapped in local energy minima. A number of approaches have been proposed to improve these points.

2.2 Parametric deformable models

Parametric deformable models control the deformations using a set of parameters which are capable of encoding a specific characteristic shape and its variations. This type of model is used when more specific shape information is available, which can be described by a set of parameters. The advantage of these models, also called **deformable templates**, is that they are better at bridging boundary gaps and they are more consistent.

Parametric deformable models can be categorised into **analytical** deformable templates and **prototype-based** deformable templates. For the former the shape variation is parametrised through handcrafting of a parametric formula for the curves in the shape template such that different shape instances can be obtained using different parameter values. For the latter a prototype is designed for a shape class, and parametric transformations are then applied on the prototype to obtain different deformed templates.

In both these parametric models, the deformable templates interact dynamically with the image features by adjusting the parameters according to the image forces. Similar to the active contour approach, an objective function which is a weighted sum of an internal energy term and an external energy term is used to quantify how well a deformed template matches the objects in the given image.

2.2.1 Analytical templates

Analytical deformable templates are defined by a set of analytical curves. Hence, the geometrical shape is parametrised directly. The template is represented by a set of curves which is uniquely described by some parameters. The geometric shape of the template can be changed by varying the parameters in the analytical expressions, and the variations in the shape are determined by the

distribution of the parameter values. This representation requires that the geometrical shape is well structured.

2.2.2 Prototype-based templates

Prototype-based deformable templates are more flexible than the analytical templates because they are derived directly from a set of examples of objects which are expected in the images. This approach was first presented by Grenander et al. who described a systematic framework to represent and generate patterns from a class of shapes. A shape is represented by a set of parameters and a probability distribution on the parameters is specified to allow a flexible bias towards a particular shape. The active shape model, suggested by Cootes, which will be presented in the next section is a prototype based deformable template.

2.3 Active shape models

Cootes et al. have in [2] defined a type of prototype-based deformable templates, which they call active shape models (ASM). Contrary to many other deformable models the active shape models represent a general way of performing non-rigid object segmentation. Shape variation is extracted from a training set by applying principal component analysis to point distribution models, rather than handcrafting a priori knowledge into the model. Active shape models have been successfully applied to areas like face recognition, industrial inspection and medical image interpretation.

Active shape models are statistically based and they almost completely avoid the use of constraints, thresholds or penalties imposed by the user. The models are derived from the statistics of labelled images containing examples of the objects. Each model consists of a flexible shape template – a point distribution model (PDM) – describing how the relative locations of important points on the shapes can vary, and a statistical model of the expected grey-levels in a region around each point.

PDM's are flexible models which represent an object by a set of labelled points. The points describe the boundary or other significant parts of an object. By using PDM's the idea is to avoid the use of heuristic assumptions about legal shape deformations. Instead, knowledge about legal shape deformation is obtained by examining a representative training set. During image search, the model is only allowed to deform to shapes similar to the one seen in the training set.

Other deformable templates and snakes align to strong gradients for locating the object, regardless of their actual appearance in the image. However, model points are not always placed on the strongest edge in the locality - they may represent a weaker secondary edge or some other image structure. ASMs will instead learn the typical grey level appearance perpendicular to the contour from the training set and build a statistical model of the grey level structure to use for image search.

During a training phase the mean position and variation of each point and grey level profile of the contour are calculated. This mean shape is used as the generic template of the class of shapes. A number of modes of variation, i.e., the eigenvectors of the covariance matrix, are determined for

describing the main factors by which the exemplar shapes tend to deform from the generic shape. A small set of linearly independent parameters are used to describe the deformation. In this way, the shape model allows for considerable meaningful variability, but is still specific to the class of structures it represents.

The mean shape from the training set, the principal components transformation matrix, and the set of weights associated with each mode of variation collectively constitute the active shape model which can be fitted to image data in a variety of ways. One can, for example, use conventional optimisation algorithms to fit the model in parameter space, obtaining an initial crude fit of the mean shape by a template matching scheme.

The major advantage of these models is that they can be used to represent a wide variety of objects, both man-made and biological, the same techniques being applied in every case (they do not have to be handcrafted for each example). The models explicitly describe the variations in shape and, being compact linear representations, can be used efficiently in image search. The limitations of the approach are its sensitivity to partial occlusion, and its inability to handle large scale and orientation change.

A generalisation of the active shape model is the active appearance model (AAM), that uses all the information in the image region covered by the target object, rather than just that near modelled edges. An AAM contains a statistical model of the shape and grey-level appearance of the object of interest which can generalise to almost any valid example. Matching to an image involves finding model parameters which minimise the difference between the image and a synthesised model example, projected into the image. The potentially large number of parameters makes this a difficult problem.

Chapter 3

A method for lip-tracking

Facial speech feature extraction and modelling has become an important issue in both automatic speech processing and automatic face processing. Potential applications include audio-visual speech recognition, recognition of talking persons, lip synchronisation, speech-driven talking heads, and speech-based image coding.

Many approaches for lip-tracking simplify the problem by marking the subjects lips with colour or a reflective marker, by locating the lips in the first image by hand, by performing experiments for one subject only, or by using very controlled lighting conditions. In the following a short summary of methods for lip-tracking based on deformable models is given. None of these approaches use special marking of the lips, but they assume one subject and the approximate position of the mouth is usually located prior to the tracking.

In [8] Kass et al. have described active contour models (snakes) for lip-tracking. These are able to resolve fine details, but shape constraints are difficult to incorporate and one has to compromise between the degree of elasticity and the ability to resolve fine contour details. Bregler et al. [1] laso describe a method based on snakes for tracking the outer lip contour where the contour is limited to lie within a subspace learned from training.

Yuille et al. have used deformable templates for locating facial features [20], and this approach has been applied for lip-tracking by Hennecke et al. in [5]. Here they make use of a piecewise parabolic/quartic template which seeks to lock on to the upper and lower edges of each lip. In a manner similar to that of snakes, the deformable lip template adjusts its shape according to the value of a number of integrals along the relevant contours.

In what remains of this report we will describe in some more detail another approach for lip-tracking which is based on active shape models. This method has been developed by Luettin et al. [9, 14] and is based on the work of Cootes et al. [2] on deformable models for medical applications. In this chapter this method will be described, first the training and then the matching process. In the next chapter some results from experiments with this method are reported.

3.1 Training

This section describes the training of the active shape model for lip-tracking. The training process starts by manually marking the contour in a set of training images, and for each marked point on a contour, a grey level profile is extracted. The mean and the covariance matrix for both shape coordinates and grey level profiles are then computed, and finally the eigenvectors and eigenvalues of the covariance matrix are calculated using principal component analysis. More details on this procedure is described in the following.

Shape

The training starts by manually marking points along the contour of the objects in a set of training images. This contour may take on any shape and can consist of several separate parts, like the inner and the outer contour of the lips. In the experiment described in this report, the outer contour of the lip is used.

The training examples need to be labelled in a consistent manner to be able to compare equivalent points from different shapes. When marking the lip contour, the outer two corners of the mouth were marked first and used as reference points. The remaining points on the contour were then placed equidistantly along the line between these two points (see Figure 3.1, left). The resulting coordinates of the contour for shape i can be described as a vector: $v_i = (x_{i0}, y_{i0}, ...x_{iN-1}, y_{iN-1})$. For the outer contour $N = 22$ points were used.

When all the coordinates of the contour have been marked, they should be normalised for translation, scaling and rotation. Normalisation of the contour coordinates is performed based on the two corner points. Their distance is defined as the scale s, their orientation as the angle θ, and their centre as the origin (t_x, t_y) (see Figure 3.1, right). The normalised coordinates of shape i are represented in the vector $x_i = norm(v_i)$, and the normalisation parameters in the vector $u_i = \{t_x, t_y, s, \theta\}$.

After the contour coordinates of all the shapes in the training set have been marked and normalised, the shape model can be created. The approach for this is as follows: First the mean shape, $\overline{\mathbf{x}}$, and the covariance matrix of the data are computed. Then the eigenvectors, ϕ_i, and the corresponding eigenvalues, λ_i, of the covariance matrix are computed. (The eigenvalues give the variance of the data around the mean in the direction of the corresponding eigenvector.) The eigenvectors and their corresponding eigenvalues are sorted by decreasing eigenvalues so that $\lambda_i \geq \lambda_{i+1}$. Letting $\mathbf{P}$ be the matrix containing the t sorted eigenvectors corresponding to the t largest eigenvalues, the training set $\mathbf{x}$ can then be approximated by:

$$\mathbf{x} \approx \overline{\mathbf{x}} + \mathbf{P}\mathbf{b}$$

where $\mathbf{b}$ is a t dimensional vector given by:

$$\mathbf{b} = \mathbf{P}^{\mathbf{T}}(\mathbf{x} - \overline{\mathbf{x}})$$

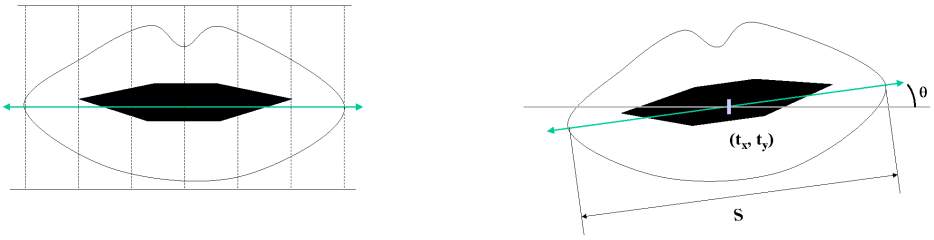


Figure 3.1: *Left: The points on the contour should be equally spaced along the line between the two corners of the mouth. Right: Normalisation of the contour is performed based on the translation, scaling and rotation of the line between the two corners of the mouth.*

defining the parameters of the deformable model. By varying the elements of $\mathbf{b}$ we can vary the shape, $\mathbf{x}$. The variance of the i^{th} parameter, b_i, across the training set is given by the eigenvalue λ_i of the corresponding eigenvector.

The number of eigenvectors t which is included in the model, can be chosen so that the model represents some proportion (eg. 98%) of the total variance of the data, or so that the residual terms can be considered as noise. (In the experiments described here, the number of eigenvectors were set to 8). Any shape $\mathbf{x}$ can then be approximated by:

$$\mathbf{x} = \overline{\mathbf{x}} + \mathbf{P}\mathbf{b}$$

using the mean shape, $\overline{\mathbf{x}}$, the matrix of eigenvectors, $\mathbf{P}$, and the weights of the eigenvectors, $\mathbf{b}$.

In summary, each shape model is represented by an average shape and a set of eigenvectors describing the statistical deformations calculated from a learning set of shapes, where these eigenvectors are calculated as the principal components of the covariance of coordinates of corresponding points along the shape outlines. The data stored for this shape model are the normalised mean shape vector $\overline{\mathbf{x}}$, the mean and the variance of the normalisation parameters, $\overline{\mathbf{u}}$ and $var(\mathbf{u})$, and the t largest eigenvectors with corresponding eigenvalues are stored.

Profile

To measure the fit between the image and the shape model, a cost function will be is used (see Section 3.2). A representation of image features along the lip contour is therefore needed. Instead of using gradients to represent the contours, the actual grey level around each contour point is used. This is obtained by extracting a grey level profile along a line perpendicular to the contour in all the contour points which are marked on the shape (see Figure 3.2). This results in a small vector of grey values $\mathbf{g}$ for each point on the contour, which are concatenated into one large vector

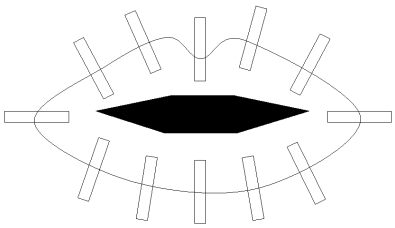


Figure 3.2: *A grey level profile is extracted for each contour point along a line perpendicular to the contour.*

for each shape i:

$$\mathbf{h_i} = (\mathbf{g_{i0}}, \mathbf{g_{i1}}, ..., \mathbf{g_{in-1}})^{\mathbf{T}}$$

In the same way as for the shape coordinates, the mean of all the grey level profiles of the training set are calculated, giving the global mean profile $\overline{\mathbf{h}}$. The covariance matrix of the profile data is also computed, and then the eigenvectors and the eigenvalues of this matrix. Again, the eigenvectors corresponding to the t largest eigenvalues are used to make up a matrix $\mathbf{P_g}$. Any profile can then be approximated using:

$$\mathbf{h} = \overline{\mathbf{h}} + \mathbf{P_g b_g}$$

where $\mathbf{b_g}$ is a vector containing the weights of each eigenvector.

Lip model

The lip model consists of a model of the shape and its variations, and a model of the grey level profile and its variations. In summary the data used to represent the lip model are:

- The normalised mean shape $\overline{\mathbf{x}}$.
- The mean and the variance of the normalisation parameters, $\overline{\mathbf{u}}$ and $var(\mathbf{u})$.
- The eigenvectors and eigenvalues of the shape coordinates, corresponding to the t largest eigenvalues.
- The mean grey level profile $\overline{\mathbf{h}}$.
- The eigenvectors and eigenvalues of the grey level profiles corresponding to the t largest eigenvalues.

In our experiments we have used $t = 8$ modes, and set the allowed variation to 3 times the standard deviation.

3.2 Matching

When the training of the model is complete, the model is ready to be used for locating contours in a set of unseen images. We then want to match the shape and its deformations to the test images. This requires that the search process moves and deforms the model gradually to shapes with lower cost. To perform this multidimensional optimisation, the **downhill simplex method** is used.

Downhill simplex method

The downhill simplex method performs multidimensional minimisation of a function $f(\mathbf{x})$, where $\mathbf{x}$ is an n-dimensional vector, using the method of Nelder and Mead [17]. It is an iterative algorithm for solving unconstrained minimisation problems numerically for several but not too many variables. The method requires only function evaluations, and no derivatives. It is however not very efficient in terms of the number of function evaluations that it requires, but it may still often be the best method to use.

The method attempts to enclose the minimum inside an irregular volume defined by a simplex. A simplex is a geometrical figure in N dimensions with N+1 vertices and edges. From a chosen starting point the algorithm is supposed to make its own way downhill through the unimaginable complexity of the N-dimensional topography, until it encounters an (at least local) minimum. It must be started not just with a single point, but with N+1 points, defining the initial simplex.

When searching for the lip contour the algorithm uses the translation parameters t_x, t_y, the scaling s, the rotation θ and the vector with the weights of the eigenvectors, $\mathbf{b}$, as variables of the multidimensional optimisation process, optimising a cost function $f(t_x, t_y, s, \theta, \mathbf{b})$. These parameters define the possible movements and deformations, and the downhill simplex method searches through this space of movements and deformations.

Initialisation of the search

The search is initialised with the mean shape, $\overline{\mathbf{x}}$, and placed in a random location. We have used the location given by the mean vector for the normalisation, $\overline{\mathbf{u}}$, as a starting point for the search. From this starting point each parameter is then varied within the allowed interval of variation until the cost reaches a specified low tolerance.

To perform this search, we first need to initialise the simplex S, which is a matrix with $N + 1$ rows of length N. Each row should contain an initial setting of the vertices of the simplex. In our case one vertex corresponds to one setting of the parameters of the two vectors $\mathbf{u}$ and $\mathbf{b}$.

The first vertex, or first row of S, is initially set to the mean shape at the mean position. We have the mean position from the mean vector of the normalisation parameters, $\overline{\mathbf{u}}$, estimated during training. To obtain the mean shape, $\overline{\mathbf{x}}$, we know from the equation $\mathbf{x} = \overline{\mathbf{x}} + \mathbf{Pb}$ that we will need to set all the weights of $\mathbf{b}$ to zero. Hence, using $t = 8$ modes, the first vertex of the simplex is set to:

$$S_0 = \{\overline{\mathbf{u}}, 0, 0, 0, 0, 0, 0, 0, 0\}$$

To compute the initial settings for the remaining vertices, we first define the vector $\mathbf{V}$ containing the allowed variation of each of the variables in S_0. This variation has been set to 3 times the standard deviation computed from the training set:

$$V = \{\sigma(\mathbf{u}), 3\sqrt{\lambda_1}, ..., 3\sqrt{\lambda_t}\}$$

Then for each of the remaining rows (vertices) of S we set for $i = 1...N$ and $j = 1...N$:

$$S_{ij} = \begin{cases} S_{0j} + V_j & \text{if } i = j \\ S_{0j} & \text{otherwise} \end{cases}$$

The final step of the initialisation consists of computing the value of the cost function $y(i)$ for each vertex in the simplex, i.e., for each row of S:

$$y(i) = cost(S_i)$$

When the simplex and the cost vector have been initialised, the search process can start.

Searching the image

The search is performed using the `amoeba()` function from numerical recipes [18] (NB! Use the one from the second edition - the version in the first edition may not be correct). The function takes the following parameters:

```
void
  amoeba(  double **p,            the matrix S described above.
           double *y,             the cost for each vertex (row) of S.
           int ndim,              the dimension of the vector of variables.
           double ftol,           fractional tolerance used to determine convergence.
           double (*funk)(...),   the cost function (specific for each problem).
           int *nfunk)            the number of times the cost function is called.
```

For the parameters controlling the convergence, the following values were used: `ftol = 1E-5`, `nfunk = 500`.

The optimisation is actually run twice for each image, restarting the multidimensional minimisation function at a point where it claims to have found a minimum. At restart the simplex is reinitialised with the minimum found in the previous run.

For the next frames in the video sequence, the position from the previous frame is used as an initial estimate.

Cost function

The cost function computes the cost for a given setting of the variables $(\mathbf{u}, \mathbf{b})$, and uses a measure which describes the fit between the grey level profile model and the image.

To compute the cost function for a new setting of the variables $(\mathbf{u}, \mathbf{b})$, the current coordinates of the shape are first computed. The deformation of the shape is calculated for the current setting of the eigenvector weights in $\mathbf{b}$ from the mean shape, $\overline{\mathbf{x}}$, and the matrix, $\mathbf{P}$, of the t largest eigenvectors:

$$\mathbf{x} = \overline{\mathbf{x}} + \mathbf{P}\mathbf{b}$$

Then the movement of the shape is computed from the current setting of the parameters in $\mathbf{u}$, the translation (t_x, t_y), the scaling s, and the rotation θ.

When the coordinates for the current shape are computed, the profile $\mathbf{h}$ for the current parameter setting can be extracted from the image. Then the current profile weights, $\mathbf{b_g}$, can be computed from the mean profile, $\overline{\mathbf{h}}$, and the matrix, $\mathbf{P_g}$, of the t largest profile eigenvectors:

$$\mathbf{b_g} = \mathbf{P_g^T}(\mathbf{h} - \overline{\mathbf{h}})$$

To measure how well the model fits, the mean square error between the image profile and the aligned model profile is computed as:

$$E = (\mathbf{h} - \overline{\mathbf{h}})^\mathbf{T}(\mathbf{h} - \overline{\mathbf{h}}) - \mathbf{b_g^T}\mathbf{b_g}$$

which gives the cost for the given parameter setting.

Chapter 4

Experiments

This chapter describes some experiments performed using the method presented in the previous chapter. Information on the specific programs that have been implemented and how they are used can be found in Appendix A and B.

4.1 Data

The "TULIPS1" database [16] was used for the experiments. This is an audio-visual database of 12 subjects saying the first 4 digits in English. Only the video files from the database were used. These consist of grey level images of 100x75 pixels, containing a small area of the face around the mouth (see Figure 4.1). This means that the region of interest is already identified, and this simplifies the problem.

From the data set of 12 subjects, only videos of 6 subjects were used in this experiment, 3 men and 3 women. The reason for using just half of the data set was that both downloading and training of a larger set would be too time consuming for this limited study. For each person, there are two sequences where each of the first four English digits are spoken, giving 8 sequences per subject and a total of 48 sequences. For our experiment we split this set in two, using 4 sequences from each person as training data and the other 4 sequences as test data.

The sequences consist of a varying number of frames, and in some frames the lip contour is outside the image. For the training set we had to omit frames with the lip contour outside or too close to the edge of the image as grey level profiles could not be extracted from these images. In the image set used for testing, no images were omitted. As a result the set of training images, consisted of 157 images and the set of test images contained 244 images. See Figure 4.1 for examples of the images of each of the 6 subjects.

Figure 4.1: *Examples of images of each of the 6 subjects used in the experiment.*

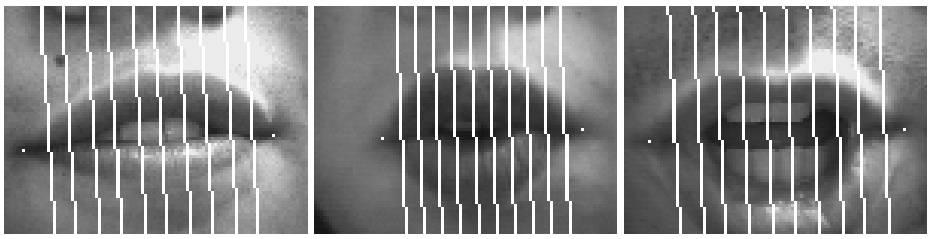


Figure 4.2: *Equally spaced lines, perpendicular to the line between the corners of the mouth, are used to help the manual labelling process.*

4.2 Test

The outer contour of the lip in each image of the training set was first manually labelled. 22 points along each contour were identified. To facilitate consistent labelling of the points, each corner of the mouth was first marked, and then a program was used to produce images where equally spaced lines perpendicular to the line between the corner points were marked. The result of this process is shown in Figure 4.2. The remaining points on the contour can then easily be identified as the points where the marked vertical lines cross the outer lip contour.

When all the contour points of the training set were identified, the grey level profiles were extracted. For each point i of the contour the grey levels along a line perpendicular to the line between point $i-1$ and point $i+1$ is sampled. This line is centred at the current contour point, and 9 samples with a distance of 1 pixel were extracted. Then the contour coordinates were normalised with respect to the translation, scaling and rotation of the line between the corners of the

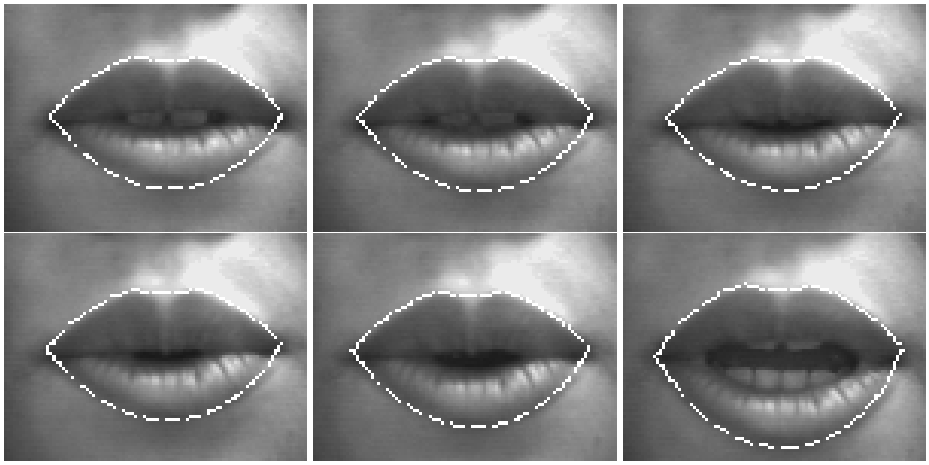


Figure 4.3: *Results from tracking one sequence, where the resulting contours are marked.*

mouth. Finally, the contour and grey level profile features are stored along with the normalisation parameters used for the contour. When all the features are extracted for all the images of the training set, the lip model is built. This model is then used to track the lip contours in the images of the test set, deforming it as described in Section 3.2 by varying the parameters determining the shape and its translation, orientation and size.

4.3 Results

The results of the tracking are shown in Figure 4.3 to 4.5. The resulting contours follow the original quite well for the first four subjects shown in Figure 4.3 and 4.4. The sequences in Figure 4.4 have different difficulties; in the leftmost sequence the contour is very near the edge and the mouth corner to the left of the image is quite diffuse, for the sequence in the middle there is a rotation of the mouth, and for the sequence to the right the shape of the mouth changes quite much. Also, the lighting and the contrast differ quite much between the images. Still, the algorithm has been able to overcome all these problems.

Figure 4.5 shows two sequences where the algorithm has had problems finding the contour in the initial frames, but has in both cases been able to recover after a few frames. The problems in these images may be due to the much weaker edges along the contours of the mouth in these sequences. The weak edges will also influence the training, as it makes consistent marking of the contours difficult. For the lower sequence in Figure 4.5 better initial estimates of the position of the mouth might have helped. Like all other methods for deformable templates, this one is also sensitive to a good initial estimate of the area of interest.

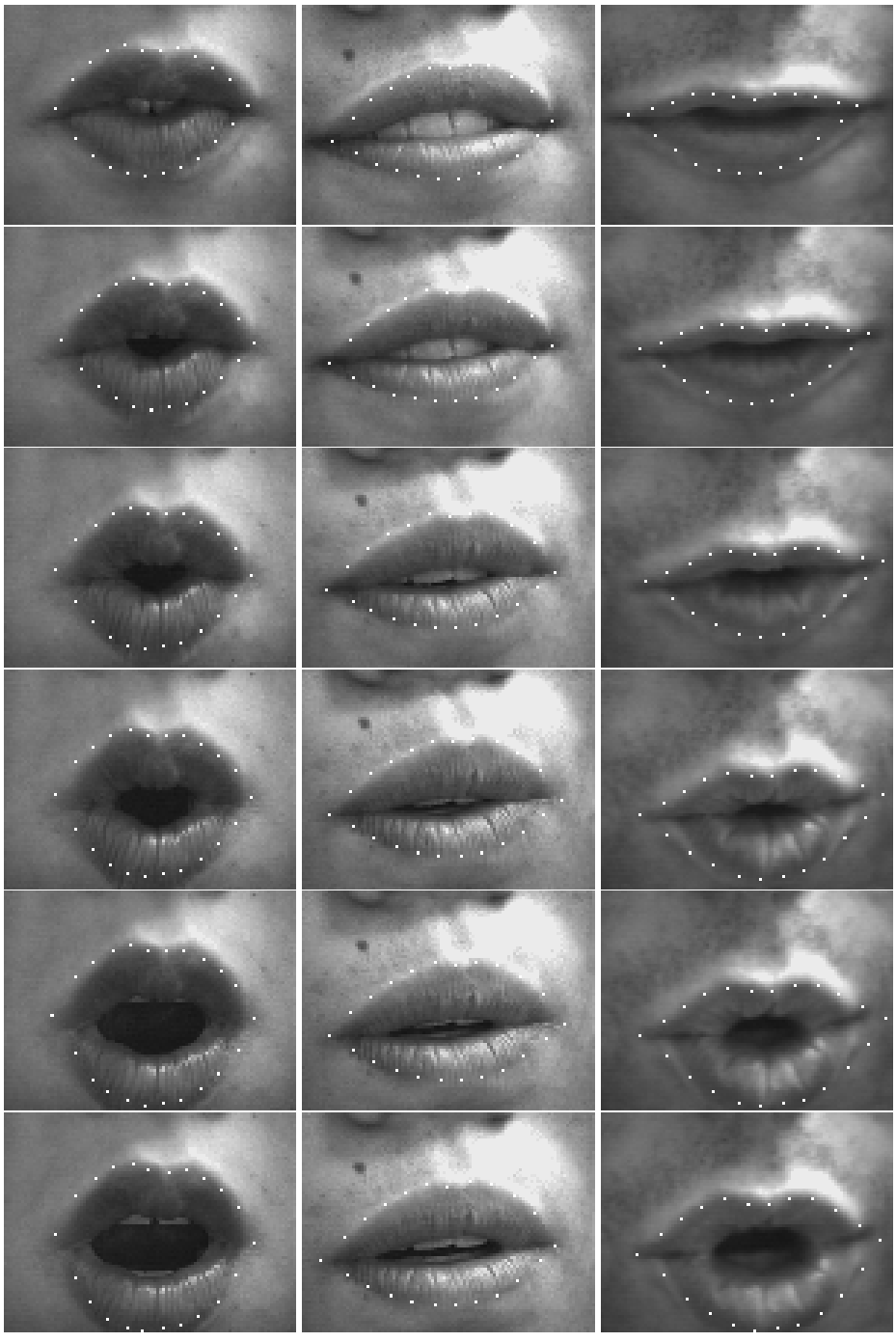


Figure 4.4: *Results from tracking of three sequence, where the tracked contour points are marked.*

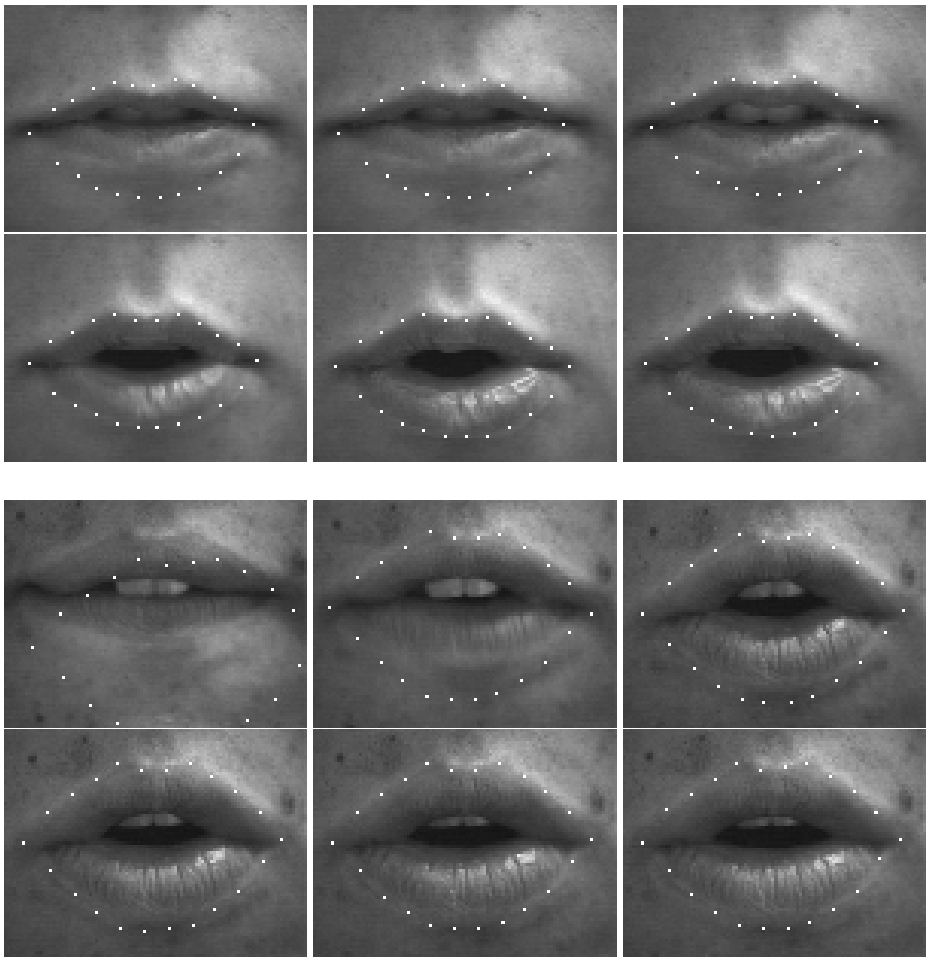


Figure 4.5: *Results of the tracking of two sequences, with problems in the first few frames.*

Chapter 5

Summary and Discussion

In this study of deformable models, a brief overview of the different models was first given. The types of models can be partitioned into two classes; free-form models and parametric models. Free-form models are often called active contours and comprise methods like the snake model by Kass et al. Parametric models may be of two types using either analytical templates or parametric templates.

We have studied in some more detail one kind of parametric deformable models called active shape models, which were developed by Cootes et al. in 1994 [2]. These models consist of a flexible shape template, a point distribution model, describing how the relative locations of important points on the shapes can vary, and a statistical model of the expected grey-levels in a region around each point The major advantage of these models is that they can be used to represent a wide variety of objects, both man-made and biological, with the same techniques being applied in every case.

Luettin et al. have applied these models for the problem of lip-tracking, and their method was implemented in this study performing experiments on a small data set. An active shape model consisting of 22 points is used to model the outer contour of the lip, and a grey level profile of 9 samples is extracted in each point along a line perpendicular to the contour. The model built from a manually labelled training set is matched to images of the test set by deforming and moving the contour within 3 standard deviations of the variation seen in the training set. The multidimensional searching process is performed using the downhill simplex method.

The results of the experiments in our small study were good. The problem was however simplified as the images in the experiment only covered a very small area of the face, meaning that the initial location of the mouth was already found. We could therefore just use the mean position of the mouth as the initial estimate. The method does however rely on good initial estimates, especially when the contours are weak.

For lip-tracking occlusion is avoided, but this is a situation which is often encountered for other problems of tracking, and it is hard to say how the method will behave for these cases. We have

however seen that the method performs well when part of the contour is outside the image, which indicates that at least partly occlusion might be handled by the method.

Another problem, which is avoided for lip-tracking, is that of moving objects. In our case the object in question, will only deform and not actually change position in the image. For moving objects, additional processing would probably be required between frames to locate the object in the image. For video tracking a process of initial location in the first two frames, combined with a prediction of location in the following frames based on speed and direction of movement, could be used.

Although the method was tested on unseen video sequences, it was not tested on unseen subjects, and for the method to work in these cases a much larger training set would be needed. However, one of the disadvantages of this method, is that the training process is quite time consuming. All points of the model must be consistently marked in a number of training images. The results are very dependent on the training process, and consistent marking of the training set can be difficult. For the lip tracking the mouth corners were used as guidelines to get a consistent labelling. However, for other types of shapes, this can be more difficult.

Although, the training process is tedious, the template model is very flexible and does not put a lot of restrictions on the shape. In addition it is not necessary to design a new model and change the actual analysis for new problems. Hence, although the training process can be time consuming and tedious, it is quite simple compared to the problem of designing a new model. More experience with this method for different problems is needed, but it seems to be very flexible and useful when a good training set can be achieved. As for all methods for deformable templates, it should be combined with additional initial object location.

Appendix A

Programs

This chapter treats the programs used to create the lip model and perform the tracking. The next sections describe the programs needed both for the training and the matching, their syntax and how they are used. All the programs work on images of the blab-format.

A.1 Training

The training of the lip model was performed by first marking manually the two corner points of the mouth in each image. The result of this process was stored in a file (see Appendix B.1). From these corner points a program for helping the process of manually labelling the contour, produces images where equally spaced lines perpendicular to the line between the corner points are marked:

```
mark <infile>
```

Input to the program should be the name of the manually produced file containing the name of the images to be processed and the identified positions of the two corners of the mouth. For each image in the list a new image with the marked lines is produced. This image will have the same path and filename as the original image except for an extra extension `.lines`.

Using the marked images produced by `mark`, the remaining points on the contour must be identified and stored in a file following the format described in Appendix B.2. The resulting file with the coordinates of all the marked contours is then input to the program `norm`, which will extract the profiles in each point and normalise the shape coordinates:

```
norm <infile> <outfile>

Purpose : Extract profiles and normalise shape.
Infile  : Shape coordinates (manually obtained).
```

```
Outfile : Grey level profiles and normalised shapes.
```

Input to the program are two filenames. The first should be the name of the file containing for each training image, the path and the full filename of the image and the identified x- and y-coordinates along the contour. The second filename should give the name of the file to which the resulting normalised features should be written.

The result of the normalisation program is a file with grey level features and normalised shape features for all the labelled training images following the format described in Appendix B.3. From this file containing all the features of the training set, the final step of the training process will produce the lip model, using the program `model`:

```
model <infile> <outfile>

Purpose : Compute models from normalised shapes.
Infile  : Normalised shape and intensity features.
Outfile : Model.
```

The input to the program is the name of the file produced in the previous step containing the normalised features, and the name of the outfile to which the computed object model will be written. The result is the active shape model for the lip contour that is written to a file following the format described in Appendix B.4.

In summary the training process is as follows:

- Manually label the mouth corners in each training image.
- Run `mark` to produce images with help lines.
- Manually label the remaining coordinates in each training image using the images with the help lines.
- Run `norm` to normalise the manually identified coordinates and to extract the grey level profiles in each point.
- Run `model` to produce the lip model from these features.

A.2 Matching

The matching should be performed on a set of unseen images, and will use the lip model produced by the program `model`. This is done using the program `search`:

```
search <model-file> <image-sequence>

Purpose : Find contour in image sequence.
```

Input to the program should be the name of the file containing the object model computed during training, and the prefix of the image sequence to be processed. The program currently expects the image filenames to be of the form `<prefix><number>`, where the images should be numbered consecutively, using 5 digits for the number. The resulting tracked contour points in each frame will be marked in a new image: `<prefix><number>.res`.

Appendix B

File Formats

Between the different steps during training data are written to file. The 4 different files which are needed contain:

- Initial corner points.
- Labelled contour points.
- Normalised features.
- Object model.

In this appendix the format of each of these files are described.

B.1 Initial corner points

This file should be produced manually and contain:

- The path and filename of the training image.
- The coordinates of the left and the right corner of the mouth.

The format of the file is as follows:

```
<path and filename of training image no 1>
<x,y of left corner> <x,y of right corner>
...
<path and filename of training image no M>
<x,y of left corner> <x,y of right corner>
```

B.2 Labelled contour points

The first line of the manually produced file containing the labelled contour points, should contain:

- The dimension of the profile (9 grey levels)
- The number of points along the contour (22 points)

where the numbers in parenthesis specifies what was used when tracking thepouter lip contour.

The rest of the file contains for each image of the training set:

- The path and filename of the image.
- The image coordinates determined along the contour. The coordinates should be organised clockwise, starting from the left corner point of the mouth. (22 points)

Hence, the format of the file can be expressed as follows:

```
<profile dimension> <number of contour points>

<path and filename of training image no 1>
<x,y-coordinates of point 1 to N on the contour>
....
<path and filename of training image no M>
<x,y-coordinates of point 1 to N on the contour>
```

B.3 Normalised features

The feature file is produced by the program `norm`, and the first line of this file contains:

- The dimension of the profile (9 grey levels)
- The number of points along the contour (22 points)

Then follows for each image:

- The path and filename of the training image.
- The normalised coordinates of the marked contour (2*22 numbers representing the x- and y-coordinates, starting from the left corner of the mouth and moving clockwise).

- The extracted profile of each point (9*22 grey levels), with the ordering of the profiles corresponding to the ordering of the contour points.

This can be summarised as:

```
<profile dimension> <number of contour points>

<path and filename of image no 1>
<x,y-coordinates of point 1 to N on the contour>
<Grey level profile of point 1 to N of the contour>
....
<path and filename of image no M>
<x,y-coordinates of point 1 to N on the contour>
<Grey level profile of point 1 to N of the contour>
```

B.4 Model file

The file containing the final model is produced by the program `model`. The first line of this file contains four numbers:

- the number of shape features (i.e. contour coordinates). For the experiment described in this report 22 contour points are used, giving a total of 44 coordinates, or shape features.
- the number of modes (i.e. the number of eigenvectors used in the approximation), which has been 8 in these experiments.
- the number of normalisation parameters. The normalisation parameters are the parameters giving the translation (t_x, t_y), scaling s and rotation θ, and this number will always be 4 for the 2-dimensional case.
- the number of profile features. For this experiment, profiles of length 9 were used for each of the 22 contour points, giving a total of 198 profile features.

Then follows the shape information:

- the mean shape vector (44 mean coordinates)
- the mean parameter vector (4 mean parameters)
- the standard deviation of the parameter vector (4 values)
- then for each mode, $t = 1...8$, the eigenvalue and the eigenvector for each mode of the shape model are given.

And then the profile information:

- the mean profile vector (198 mean grey level values)
- then for each mode, $t = 1...8$, the eigenvalue and the eigenvector for each mode of the profile model are given.

The following gives a summary of the model file format:

```
<nof shape features> <t=nof modes> <nof parameters> <nof profile fea-
tures>

<mean shape vector, length=nof shape features>
<mean parameter vector, length=nof parameters>
<st.dev. of parameter vector, length=nof parameters>

<eigenvalue-1 of shape>
<eigenvector-1 of shape, length=nof shape features>
...
<eigenvalue-t of shape>
<eigenvector-t of shape, length=nof shape features>

<mean profile vector, length=nof profile fetures>

<eigenvalue-1 of profile>
<eigenvector-1 of profile, length=nof profile features>
...
<eigenvalue-t of profile>
<eigenvector-t of profile, length=nof profile features>
```

Bibliography

[1] C. Bregler and S.M. Omohundro.
Surface Learning with Applications to Lipreading.
In Cowan, J.D., Tesauro, G., and Alspector, J. (eds.), Advances in Neural Information Processing Systems 6. San Francisco, CA: Morgan Kaufmann Publishers, 1994.

[2] T. F. Cootes, C.J.Taylor and J. Haslam.
The Use of Active Shape Models for Locating Structures in Medical Images.
Image and Vision Computing, Vol.12, No.6, July 1994. pp.355-366.

[3] T.F. Cootes, G. Edwards and C.J. Taylor.
Comparing Active Shape Models with Active Appearance Models.
Proc. British Machine Vision Conference (ed T.Pridmore, D.Elliman), Vol. 1, 1999, pp. 173-182.

[4] T. F. Cootes, C.J.Taylor.
Statistical Models of Appearance for Computer Vision.
Report of University of Manchester, July 2000.

[5] M.E. Hennecke, K. Venkatesh Prasad, and David G. Stork.
Using deformable templates to infer visual speech dynamics.
In 28th Asilomar Conference on Signals, Systems, and Computers, pages 578-582, Pacific Grove, CA, November 1994. IEEE Computer Society Press.

[6] S. Horbelt and Jean-Luc Dugelay.
Active contours for lipreading - combining snakes with templates.
15'th GRETSI Symposium on Signal and Image Processing, Juan les Pins, 18-22 Sept, 1995, France.

[7] A.K. Jain, Y. Zhong and M-P. Dubuisson-Jolly.
Deformable template models: A review.
Signal Processing 71 (1998), pp. 109–129.

[8] M. Kass, A. Witkin, and D. Terzopoulos.
Snakes: Active Contour Models,
Int. Journal of Computer Vision, pages 321-331, 1988.

[9] J. Luettin, N.A. Thacker and S.W. Beet.
Active Shape Models for Visual Speech feature Extraction.
In Speechreading by Humans and Machines, Springer Verlag, pp.383-390, 1996.

[10] J. Luettin and N.A. Thacker.
Speechreading using Probabilistic Models.
Computer Vision and Image Understanding, Vol. 65, No. 2, Feb. 1997, pp. 163-178.

[11] J. Luettin, N.A. Thacker and S.W.Beet.
Speaker identification by lipreading.
Proceedings in the International Conference on Spoken Language Processing, Philadelphia, PA, USA, 1996.

[12] J. Luettin, N.A. Thacker and S.W. Beet.
Locating and Tracking Facial Speech Features.
Proceedings of International Conference on Pattern Recognition, Vienna, Austria, 1996.

[13] J. Luettin, N.A. Thacker and S.W. Beet.
Statistical Lip Modelling for Visual Speech Recognition.
Proceedings of the VIII European Signal Provessing Conference, Trieste, 1996.

[14] J. Luettin, N.A. Thacker and S.W.Beet.
Active Shape Models for Visual Speech Feature Extraction
D.G.Stock (ed): NATO ASI Speechreading by Man and Machine: Models, Systems and Applications, Springer Verlag 1996.

[15] I. Matthews, J.A. Bangham, R. Harvey and S. Cox.
A Comparison of Active Shape Model and Scale Decomposition Based Features for Visual Speech Recognition.
Proceedings of ECCV (2) 1998: pp. 514-528.

[16] J.R. Movellan.
Visual Speech Recognition with Stochastic Networks
In G. Tesauro, D. Toruetzky, T. Leen: "Advances in Neural Information Processing Systems", Vol. 7, MIT Press, Cambridge 1995.

[17] J.A. Nelder and R. Mead.
A Simplex Method for Function Minimization.
Computer Journal, Vol 7, pp 308-313, 1965.

[18] W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P Flannery.
Numerical Recipes in C : The Art of Scientific Computing, Second Edition.
Cambridge University Press, 1995.

[19] M.B. Stegmann, R. Fisker and B.K.Ersbøll.
On Properties of Active Shape Models.
Technical Report of Technical University of Denmark, IMM-REP-2000-12, March 2000.

[20] A.L. Yuille, P.W. Hallinan and D.S. Cohen.
Feature extraction from faces using deformable templates.
International Journal of Computer Vision. 8:2, pp 99-111. 1992.