A model of facial behaviour

F. Bettinger and T. F. Cootes
University of Manchester
Division of Imaging Science and Biomedical Engineering,
Stopford Building, Oxford Road,
Manchester M13 9PT, U.K.
bettingf@cs.man.ac.uk, tim.cootes@man.ac.uk

Abstract

We wish to model the way in which faces move in video sequences. We represent facial behaviour as a sequence of short actions. Each action is a sample from a statistical model representing the variability in the way it is performed. The ordering of actions is defined using a variable length Markov model. Action models and variable length Markov model are trained from a long (20000 frames) video sequence of a talking face. We propose a novel method of quantitatively evaluating the quality of the synthesis by measuring overlaps of parameter histograms. We apply this method to compare our technique with an alternative model that uses an autoregressive process.

1. Introduction

This paper describes a prototype of a facial behaviour learning system and demonstrates its performance at learning facial behaviours by comparing the results with an alternative model proposed by Campbell *et al.* [3].

We seek to develop a system which can model both the appearance and behaviour of a person's face. We would like to be able to present the system with a sufficiently long training sequence of an individual speaking, moving their head and changing expression, and have the system learn a model capable of simulating their behaviour. Such a system would be useful for many applications from computer games to the generation of believable avatars for human-computer interaction.

Our model is designed for analysing relatively low-level behaviour (how a person tends to shake their head or the particular way they smile) rather than more high-level behaviours (such as when they smile or the order in which they tend to perform actions). We assume these low-level behaviours are characterised by relatively short time scales and are repeated sufficiently often in a training sequence that we can recognise them and model their variability. Implicit in the work is the assumption that people do not repeat any action exactly (no-one smiles the same way twice), but that it is possible to learn a distribution representing the variations on a particular action that an individual tends to make.

In the following we review related work, describe our system in more detail and show how we compare models of facial behaviour. The results of the comparison between the two models of facial behaviour are given in section 5.2.

2. Previous Work

In [14], Schödl *et al.* introduce the concept of video textures. Their aim is to generate an infinitely long video sequence based on frames from an existing video sequence. Loops in the video sequence are created by jumping from one frame to another in the original video sequence. The chosen frames are selected to exhibit similar appearance and dynamics while avoiding dead ends in the generated video sequence.

Bregler, in [2], uses a hierarchical framework to recognise human dynamics. His framework can be decomposed into four steps: the raw sequence, a model of movement using a mixture of Gaussians, a model of linear dynamics and a model of complex movements using a hidden Markov model. He highlighted the need of high level information for a correct model of behaviour.

In [10], shapes are approximated by splines. The parameters controlling those splines as well as their speed are first clustered into prototype vectors using a competitive learning neural network. A compressed sequence derived from the prototype vector sequence is learnt using a Markov chain. A cubic Hermite interpolation is used along with the learnt Markov chain to recover the temporal structure of the sequence before compression and to extrapolate a behaviour. Furthermore, for generation purposes, a single hypothesis propagation and a maximum likelihood framework are described. During the generation, states of the Markov chain can be chosen according to the state of the shape of a tracked person. This can allow generation of a shape of a virtual partner driven by a tracked real person. In [7], Devin and Hogg added sound and appearance to the framework in order to demonstrate that producing a talking head is possible. [8] introduces the use of variable length Markov model with prototype vectors to learn the structure of the sequence.

Hack *et al.* model trajectories by a sequence of pathlets in the appearance parameter space [9]. The joint probabilities of consecutive pathlets are modelled using a hidden Markov model to capture longer-term behaviour. New videos can be generated by sampling from the model.

In [16], Walter *et al.* model gestures by groups of trajectory segments. The trajectory segments are extracted by detecting discontinuities in the gesture trajectory. After normalising the trajectory segments, their dimensions are reduced using a principal component analysis. Clusters are then extracted from the component space using an iterative algorithm based on minimum description length. The clusters form atomic gesture components. There is a parallel between groups of trajectory segments and the actions or visual units we want to extract from the video sequence. However our segmentation and grouping algorithms are both different.

Finally, Ng and Gong [11] use the Levenshtein distance along with an unsupervised version of the Normalised Cut algorithm [15] to group gesture trajectories. Unfortunately, outliers are associated with some groups, which leads to problematic modelling of those groups.

3 Structure of the model

3.1 Introduction

In order to be able to generate video sequences of faces, we first need an underlying model that is able to synthesise a face for each frame. Thanks to its synthesis facility, the active appearance model of Cootes *et al.* [5] is a perfect candidate for this task.

In order to encode each frame from the training sequence, we use a full appearance model that combines shape and texture information. After having computed the mean shape from the training set, the number of parameters of the model is reduced by applying consecutive principal component analysis to both the shape and the texture part of the model. The details of the model are described in [6]. The shape and a shape-free texture are modelled by the set of linear equations:

$$\begin{cases} x = \overline{x} + Q_x c \\ t = \overline{t} + Q_t c \end{cases} \tag{1}$$

where x is a vector describing the shape, t is a vector describing the shape-free texture, Q_x and Q_t are matrices learnt from the training set. $\overline{x}$ and $\overline{t}$ represent the mean shape and mean shape-free texture computed from the training set.

Given a vector of appearance parameters c, the shape x can be computed. A shape-free texture t can be warped to the shape to reconstruct the full appearance of a face.

Each vector from the appearance parameter space represents a face while each facial image can be approximated by a vector in the appearance parameter space. A sequence of a face can be represented by a trajectory in the appearance parameter space. Visual units are therefore sub-trajectories within this trajectory.

Figure 1 shows an overview of the model of facial behaviour. First, the face has to be tracked in the video sequence (1). The active appearance model parameters have then to be deduced from the tracked face ($1 \to 2$). The trajectory formed by those appearance parameter vectors is then broken into sub-trajectory groups ($2 \to 3$) and the sequence is now a sequence of sub-trajectory groups (3). The sequence of sub-trajectory groups is learnt ($3 \to 4$) by a variable length Markov model (4).

In order to generate new trajectories, a sequence of sub-trajectory groups has to be sampled from the variable length Markov model ($4 \to 3$). A new sub-trajectory has to be sampled from each group in the sequence of sub-trajectory groups ($3 \to 2$) to give a sequence of sub-trajectories, that is a trajectory (2). Each point in that new trajectory in the appearance parameter space can then be synthesised ($2 \to 5$) to give a video sequence of faces (5).

3.2 Extracting the sequence of parameters

The active appearance model is able to fit an appearance model onto a face image, by minimising the difference of texture between the synthesis of the model and the image of interest. As it is a local minimisation, it requires a good first approximation. For each training sequence, the first frame is annotated. In order to get a good approximation for each frame we use the state of the face (pose, scale, position and appearance) from the previous frame as the first approximation for the fitting procedure in the current frame. By comparing the synthesised face and the corresponding pixels in the current frame (using a mean square error on the grey-level pixel values), we can determine whether the fitting procedure has failed or not. If it has failed, a global search is performed. In practice, only a few global searches need to be done in order to track the face accurately.

This method has been used to represent an image sequence as a sequence of parameter vectors.

3.3 Segmenting into sub-trajectories

Given a long sequence of points in the parameter space, we want to divide it into sub-trajectories. These sub-trajectories correspond to actions or visual units in the video sequence.

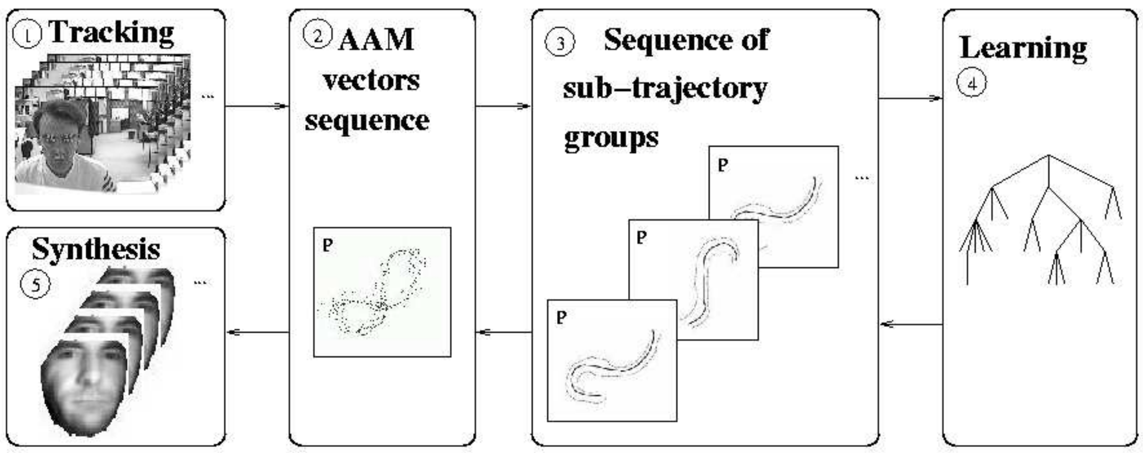


Figure 1: Overview of the components of the model. P is the appearance parameter space. Arrows from left to right represent the learning and arrows from right to left represent the generation. Each face from the frames of the original video sequence corresponds to a point in the space P. The trajectory of the points in P is modelled by a sequence of sub-trajectory groups. The temporal relationship of the groups is learnt by a variable length Markov model.

The aim of the segmentation is to extract some nodes that will split the trajectory into several sub-trajectories. The nodes will form the beginnings and ends of the sub-trajectories. The sub-trajectories are computed in order to be grouped later in the process, so similar sub-trajectories should also have similar beginnings and ends respectively. Furthermore, we would like to find the points where different behaviours split or converge together. Finding points of high density in the appearance parameter space is a good way of meeting these requirements. In order to find the high density points, we use the sample mean shift, described by Comaniciu and Meer [4]. We iteratively modify our current estimate of the local maxima of density by moving to the mean of the n closest points of the current estimate. The process converges to the position of the local maximum density.

We initialise the mean shift algorithm at each point of the trajectory in turn. Running the algorithm to convergence finds all the nearly local maxima in the density estimate. The trajectory points nearest to each local maxima are defined to be the nodes splitting the full path into sub-trajectories. In practice, we only do this for every m points in the training sequence. This improves efficiency with a negligible effect on the result.

3.4 Grouping similar sub-trajectories

3.4.1 The model of a group of sub-trajectories

We model each sub-trajectory using a linear statistical model, assuming a Gaussian distribution. We want each sub-trajectory to be described by a vector, which is a simple concatenation of the sub-trajectory points. A sub-trajectory group is a set of vectors, on which we can apply a principal component analysis. In order to be able to perform the principal component analysis on a group of sub-trajectories, it is required that all the sub-trajectories are encoded with the same number of points. Therefore, we interpolate all the sub-trajectories by cubic splines and homogeneously re-sample them to a given number of points. The resulting points after re-sampling do not have the same timings as the original ones. We need to keep track of this information. The vector used to describe a sub-trajectory is therefore extended with the timing information. Each extra coordinate represents the time necessary to reach a point from the first point of the sub-trajectory. Each sub-trajectory s, along with the corresponding timings δ, is approximated by:

$$\begin{pmatrix} s \\ \delta \end{pmatrix} = \begin{pmatrix} \overline{s} \\ \overline{\delta} \end{pmatrix} + Q_{s,\delta} b_{s,\delta} + R_{s,\delta} \qquad (2)$$

where $(\overline{s}, \overline{\delta})$ is a vector representing the mean sub-trajectory of the group along with this timings, $Q_{s,\delta}$ is a matrix computed by the principal component analysis and describing how the data varies, $R_{s,\delta}$ are the residuals and $b_{s,\delta}$ is the vector of parameters for that particular sub-trajectory. The probability density function of the set of parameters $b_{s,\delta}$ is modelled by a Gaussian, by computing the mean and variance of $b_{s,\delta}$ for all sub-trajectories (s, δ) in the group.

Generation of sub-trajectories is therefore a simple sampling from the distribution of $b_{s,\delta}$. Applying equation 2 gives a sub-trajectory s which is then re-sampled using cubic splines, and timing information δ.

3.4.2 The grouping algorithm

The grouping algorithm first assumes that each sub-trajectory given by the segmentation initially forms a group by itself. The timings δ are not taken into account for the grouping. Let $\mathcal{S}$ be this initial set of groups.

We want to know how to merge groups so that the resulting groups are sufficiently coherent that a Gaussian model is a good representation.

For every pair (g_i, g_j) of elements of $\mathcal{S}$, we compute the variance of the group $g_{i\cup j}$ that is built using the sub-trajectories from both g_i and g_j. We select the pair of groups (g_a, g_b) that gives the lowest variance and merge those two groups to form only one group. We delete g_a and g_b from $\mathcal{S}$ and insert $g_{a\cup b}$ instead. We iterate the process until we reach a given number of clusters.

More details on the algorithm and its performance can be found in [1].

3.5 Learning temporal relationships between groups

In order to learn the structure of the sequence of sub-trajectory groups, we use a variable length Markov model introduced by Ron *et al.* [13]. More details are given in [1]. The algorithm basically constructs a decision tree that stores probabilities of common sub-trajectory group sequences.

3.6 Generating new sequences

A new video sequence of faces can be generated from the model as follows. First, given a history of generated sub-trajectory groups, one can find the longest sequence encoded in the variable length Markov model tree. Thus the probability of generating a new group can be read directly from the tree, if it is encoded in the tree. The sequences not encoded in the tree have small probabilities, that we can approximate by an uniform distribution. After having fetched the probabilities of generation of each sub-trajectory groups, we sample from this set of probabilities to generate the next sub-trajectory group. We then generate new parameters by sampling from a Gaussian distribution. The new sub-trajectory can then be generated as described in section 3.4.1. A linear model is chosen for the residuals $R_{s,\delta}$ so that the beginning of the generated sub-trajectory matches the end of the previous generated sub-trajectory to avoid perceptible jumps in the generated video. All the sub-trajectories generated are then concatenated. This gives a sequence of appearance parameters. The video sequence can then be generated by synthesising those parameters into a face as described in section 3.1.

An example of generated trajectory is given by Figure 4(c). The three dimensions represent the three first modes of variation of the active appearance model used. Another example of generated video can be seen on Figure 3. The original video sequence is shown on Figure 2.

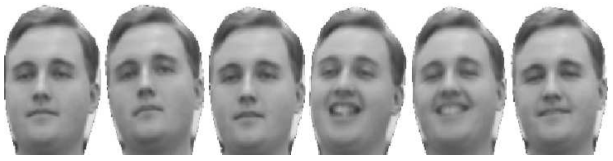


Figure 2: Frames taken every 4 seconds from the original long video sequence.

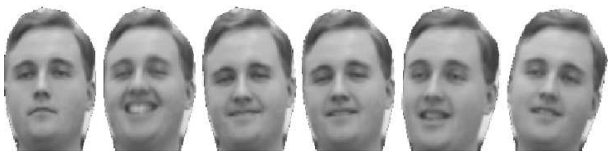


Figure 3: Frames taken every 4 seconds from the video sequence generated with our model.

4. Campbell's model

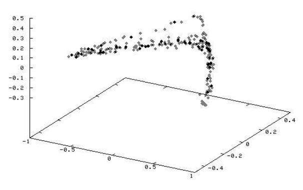
In [3], Campbell *et al.* introduce another way of generating video sequences of faces based on an existing video clip without direct reuse of the original frames. They encode frames from the original sequence in a way similar to our method. An active appearance model is used to model the face through the video sequence and a trajectory is obtained in the appearance parameter space. This trajectory is then used to train a second order autoregressive process.

An autoregressive process is simply trying to predict the position of a point y_k in the appearance parameter space, given the two previous points y_{k-1} and y_{k-2} where k represents the frame number. This prediction is produced by the equation:

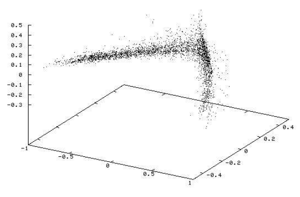
$$y_k - \overline{y} = A_2\left(y_{k-2} - \overline{y}\right) + A_1\left(y_{k-1} - \overline{y}\right) + B_0 w_k \quad (3)$$

where $\overline{y}$ is the limit of the mean value of y_k as k tend to infinity, w_k is a white noise ($w_k \sim \mathcal{N}(0,1)$), A_2, A_1 and B_0 are parameter matrices. $\overline{y}$, A_2, A_1 and B_0 can be learnt from the original data set. The learning method used in this work is due to Reynard *et al.* and is described in [12].

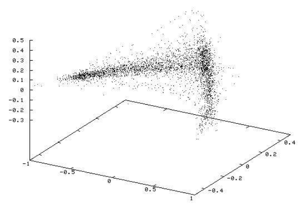
Given two initial points in the parameter space, a new trajectory can be generated by applying equation 3 repetitively. This new trajectory in the parameter space can then be synthesised back to a video sequence of a face in a similar way to ours. Figure 5 shows an example of a video clip generated by an autoregressive process.



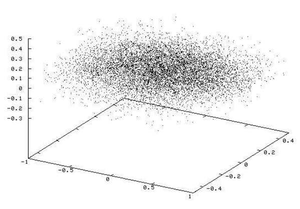
(a) Training sequence of a face gesturing ”no”.



(b) Generated sequence with our model without using the residuals.



(c) Generated sequence with our model and a linear model for the residuals.



(d) Generated sequence with an autoregressive process.

Figure 4: Figure 4(a) represents points on the trajectory that correspond to the original video sequence. Figure 4(b) represents points on the trajectory generated by our model if the residuals used are set to zero. Figure 4(c) represents the equivalent with a linear model for residuals. Figure 4(d) represents points on the trajectory generated by an autoregressive process.

5. Comparison of the two models

5.1 Method of comparison

In order to compare the two models, we need some measure of behavioural similarity between two video sequences. Our approach is to compare the distribution of the generated points in the parameter space with the distribution of points extracted from the original video sequence.

We construct two dimensional histograms to approximate the distribution of points in the parameter space for each pair of dimensions. In order to compare the original and generated sequences of points using histograms, particular care has to be taken on the selection of the bin width used to compute those histograms. Indeed, a too large bin width will act as a smoothing effect on the original data while a too small bin width will result in an over-fitting of the data. In order to solve this problem, and for reproducibility of the comparison method, we used Scott’s rule to select the bin size [17]. The bin size for each dimension

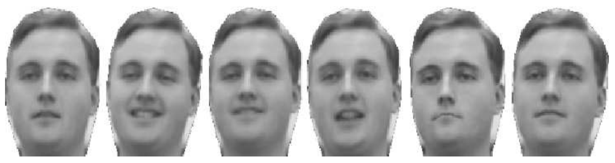


Figure 5: Frames taken every 4 seconds from the video sequence generated with an autoregressive process.

is given by the formula

$$h = 3.5\xi N^{-\frac{1}{3}} \tag{4}$$

where N is the total number of points in the original sequence and ξ is the standard deviation of the original data computed with respect to the selected dimension.

The two dimensional histograms of a reference and a generated set of points are then compared using the mean μ of the Bhattacharyya overlap computed for each pair of dimensions. The standard error σ can also be computed to represent our confidence in the result. This measure represents how close the point distributions of the generated and the reference sequences are. A value of 1 corresponds to a perfect match of the distributions. A value of 0 corresponds to two totally different distributions.

The facial behaviour models are assessed with respect to the original video sequence used to create those models. The distributions of points in the appearance parameter space should match with the original one if the model performs its task correctly.

5.2 Experimental results and discussion

Table 1 presents the results of the comparisons for three different generated videos. The first one is trained using a highly structured clip of a face gesturing ”no” (317 frames). The second one is trained on a structured clip of a face showing different expressions one after the other (1069 frames). Finally the third video is trained on a long sequence of a person in a free dialog (20000 frames), which is relatively unstructured. Each training video is compared with each model using the measure described in section 5.1 for both position and speed data. The bold labels represent lines where our model is significantly better than the autoregressive process. We can see that our model outperforms the autoregressive process for structured videos, while maintaining good results for the unstructured video.

A visual inspection of the generated sequences also shows that our model performs best when we use the linear model for the residuals. Indeed, it produces smooth video sequences that exhibit realistic behaviour. If the residuals are set to zero, then the generated videos contain perceptible jumps of the face, thus giving a worse overall effect.

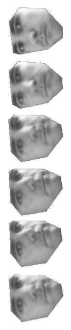
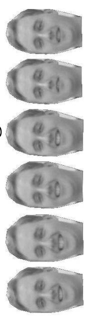
No		ARP	position :	$\mu = 0.898, \sigma = 0.010$
			speed :	$\mu = 0.863, \sigma = 0.008$
		M	**position :**	$\mu = 0.965, \sigma = 0.003$
			speed :	$\mu = 0.910, \sigma = 0.004$
		M’	**position :**	$\mu = 0.944, \sigma = 0.004$
			speed :	$\mu = 0.896, \sigma = 0.005$
Expressions		ARP	position :	$\mu = 0.828, \sigma = 0.011$
			speed :	$\mu = 0.782, \sigma = 0.011$
		M	position :	$\mu = 0.829, \sigma = 0.009$
			speed :	$\mu = 0.901, \sigma = 0.005$
		M’	**position :**	$\mu = 0.865, \sigma = 0.006$
			speed :	$\mu = 0.897, \sigma = 0.005$
Dialog		ARP	position :	$\mu = 0.896, \sigma = 0.003$
			speed :	$\mu = 0.870, \sigma = 0.002$
		M	**position :**	$\mu = 0.931, \sigma = 0.001$
			speed :	$\mu = 0.864, \sigma = 0.002$
		M’	position :	$\mu = 0.813, \sigma = 0.006$
			speed :	$\mu = 0.903, \sigma = 0.002$

Table 1: Comparison of the autoregressive process (ARP) with our model (M & M’). M is not using residuals while M’ is using a linear model for residuals.

Finally, the autoregressive process produces a video with a lot of perceptible jitters, due to the noise present in equation 3, which also give a worse effect.

6. Summary and Conclusions

We have presented a generative model of visual facial behaviour that is based on the assumption that people repeat facial expressions over time. It has been shown that this model is able to reproduce simple behaviours.

A measure of similarity between video sequences of facial behaviour has been developed and used to assess the model as well as an alternative based on autoregressive process. Our model performs better on highly structured video sequences such as nodding where expressions are repeated over time. It also produces a better distribution of speed. The model is capable of synthesising convincing facial behaviour.

We are currently investigating extending the work to include interactive behaviour between two people as they talk.

References

[1] F. Bettinger, T. F. Cootes, and C. J. Taylor. Modelling facial behaviours. In *BMVC*, 2002.

[2] C. Bregler. Learning and recognizing human dynamics in video sequences. In *CVPR*, 1997.

[3] N. W. Campbell, C. Dalton, D. Gibson, and B. Thomas. Practical generation of video textures using the auto-regressive process. In *BMVC*, 2002.

[4] D. Comaniciu and P. Meer. Mean shift analysis and applications. In *ICCV*, 1999.

[5] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *ECCV*. Springer, 1998.

[6] T. F. Cootes and C. J. Taylor. Statistical models of appearance for medical image analysis and computer vision. In *Proc. SPIE Medical Imaging*, 2001.

[7] V. E. Devin and D. C. Hogg. Reactive memories: An interactive talking-head. In *BMVC*, 2001.

[8] A. Galata, N. Johnson, and D. Hogg. Learning variable-length Markov models of behaviour. *CVIU*, 81(3):398–413, March 2001.

[9] C. A. Hack and C. J. Taylor. Modelling ’talking head’ behaviour. In *BMVC*, 2003.

[10] N. Johnson, A. Galata, and D. Hogg. The acquisition and use of interaction behaviour models. In *CVPR*, 1998.

[11] J. Ng and S. Gong. Learning intrinsic video content using Levenshtein distance in graph partitioning. *Lecture Notes in Computer Science*, 2353:670–684, 2002.

[12] D. Reynard, A. Wildenberg, A. Blake, and J. A. Marchant. Learning dynamics of complex motions from image sequences. In *ECCV*, 1996.

[13] D. Ron, Y. Singer, and N. Tishby. The power of amnesia. In *Advances in Neural Information Processing Systems*, volume 6, 1994.

[14] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa. Video textures. In *SIGGRAPH*, 2000.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. *CVPR*, 1997.

[16] M. Walter, A. Psarrou, and S. Gong. Auto clustering for unsupervised learning of atomic gesture components using minimum description length. In *ICCV-RATFG-RTS*, 2001.

[17] M. P. Wand. Data-based choice of histogram bin width. *The American Statistician*, 51(1):59, February 1997.