A 3D Head Tracker for an Automatic Lipreading System

Gareth Loy*, Eun-Jung Holden, Robyn Owens****

* Department of Systems Engineering
Research School of Information Sciences and Engineering
Australian National University, Acton 0200
gareth@syseng.anu.edu.au

** Department of Computer Science
University of Western Australia, Nedlands 6907
{eunjung, robyn}@cs.uwa.edu.au

Abstract

A real world automatic lip reading system must be able to cope with movement of the speaker's head during operation. The observed mouth shape depends not only on the true shape of the mouth, but also the angle at which the mouth is viewed. As the speaker's head moves and rotates the viewing angle changes. The resulting distortion can lead to inaccurate mouth measurement and incorrect phoneme recognition. We have developed a system that robustly measures the dimensions of a speaker's mouth whilst the speaker's head is moving and exhibiting rotations of up to 30 degrees away from the camera. Our system tracks the pose of the speaker's head in 3D, detects the mouth by tracking unadorned lip contours and estimates the 3D locations of the upper and lower lip edges and the mouth corners. The system is demonstrated on a person speaking whilst moving his head in 3D, and the mouth height and width are corrected over 9 seconds of 25Hz video footage.

1 Introduction

Automatic lip reading has important applications such as audio-visual speech recognition [Bregler and Omohundro., 1995] [Harvey *et al.*, 1997] and graphical speech synthesis [Yamamoto *et al.*, 1998]. A number of techniques have been reported to extract mouth features for visual lip reading. Active contour models [Kass *et al.*, 1987] have been used to detect lip contours [Bregler and Omohundro., 1995] [Chiou and Hwang, 1994] and [Li *et al.*, 1995] applied the eigensequence approach that is often used in facial expression recognition.

These systems, however, assume the speaker is directly facing the camera, and do not allow any head movement that would distort the lip shape in the captured images. For example, turning the head to the side makes the lip width and height ratio appear differently on images even when the speaker's mouth shape does not change.

As people talk, their heads naturally move about as they gesture and follow natural conversation cues. It is necessary for an automatic lip-reading device to be robust with respect to this behavior; to be able to detect, monitor and account for 3D movement of a speaker's head.

We have developed a lip tracking system that allows the speaker's head to move in 3D and rotate up to 30 degrees away from the camera. Our system performs the following tasks:

- Tracking the speaker's head in 3D;
- Tracking the top, bottom and corners of the mouth in the 2D input image;
- Estimating the 3D locations of the top, bottom and corners of the mouth; and
- Determining the mouth dimensions from the 3D mouth information.

Firstly, this paper presents the 3D head tracker used to determine the head pose in each frame. Secondly, it explains the detection of the mouth features, the estimation of their locations in 3D and the calculation of the corrected mouth dimensions. Lastly, it demonstrates

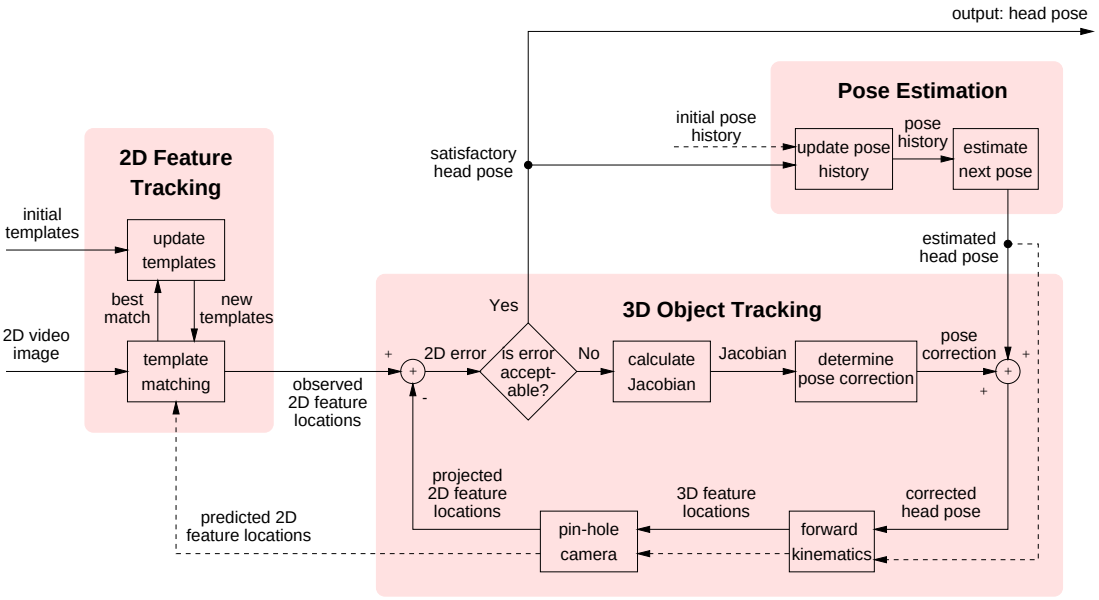

Figure 1: Block diagram of the 3D head tracking system.

the usefulness of this technique with a practical experiment. A subject enunciates 5 phonemes displaying the 5 mouth shapes of visual speech while moving and turning his head in 3D.

2 3D Head Tracker

The head tracking system uses template matching to locate features on the face. A rigid 3D model of these feature points is then iteratively rotated and translated until the projection of the 3D model points on the image plane closely matches the 2D feature locations observed via the template matching. The pose correction for each rotation and translation of the 3D model is determined using Lowe's object tracking algorithm [Lowe, 1991].

The head tracking system can be broken into three subsystems dealing with pose estimation, 2D feature tracking and 3D object tracking. Figure 1 shows a block diagram of the system with the three subsystems clearly identified.

The overall operation of the system is summarised as follows. The pose estimator supplies an estimated pose to the 3D object tracker (indicated by the dotted line in Figure 1). Using this pose, projected feature locations are predicted and passed to the 2D feature tracker which searches for the features in the vicinity of these locations. These observed 2D feature locations then form the target state of the object tracker. The object tracker performs up to ten iterations per frame until a satisfactory pose is found that places the projected 2D feature points sufficiently close to the observed feature locations (errors of up to 4 pixels per feature are tolerated). If no satisfactory pose is found in twelve consecutive frames the tracking is assumed to be lost and the system aborts.

Each of the subsystems is discussed in detail below.

2.1 Pose Estimation

The pose in the next frame is estimated by assuming that the translational and angular velocities between the current frame and the next are equal to the average of the velocities over the previous 5 frames, that is

$$\hat{p}[k+1] = \vec{p}[k] + \sum_{i=1}^{n} \frac{\vec{p}[k-i+1] - \vec{p}[k-i]}{n}$$

where n = 5. Initially, the pose history is assumed to be $\vec{0}$ (this initial pose history is shown as dotted lines in Figure 1).

2.2 2D Feature Tracking

The motion of the head in the video sequence is tracked using template matching of three feature points. For our experiment the outer corners of each eye and the inner corner of one nostril were chosen, but any three (or more) distinctive points that are rigidly attached to the head can be used.

The features to be tracked were manually selected in the first frame of the sequence and the initial templates $\mathcal{T}_i[0]$ created.

In every new frame, each template is correlated across a search area centred about the 2D feature location that was predicted by the projection of the estimated pose $\hat{p}$. Once the best match is found (and provided the correlation coefficient is greater than 0.75) the template $\mathcal{T}_i[k]$

(for the i^{th} feature in the k^{th} frame) is updated to become the weighted average of the initial template and the image region $\mathcal{R}_i[k]$ in the new frame that best matches the current template, that is

$$\mathcal{T}_i[k+1] = \frac{1}{3}\mathcal{T}_i[0] + \frac{2}{3}\mathcal{R}_i[k],$$

where $\mathcal{T}_i[0]$ is the initial template.

Updating templates in this fashion makes the template matching robust to feature distortion resulting from head rotation. Such distortion typically occurs gradually over a series of frames, thus allowing the templates time to adapt. The new template is defined as a combination of the current and the initial feature appearance in order to prevent template drift that results from simply updating templates to be the current feature appearance $\mathcal{T}_i[k]$. In the latter case, any error in the template matching will never be corrected, since the new template will be chosen off target. Even if there is no incorrect matching such templates will do a random walk about the image over time due to the quantisation error (up to half a pixel) present in each template match. Grounding each template with a portion of the initial template prevents this problem.

The computational load of the template matching is significantly reduced by performing the search in two stages: an initial sparse search to localise the feature, followed by a localised detailed search. The sparse search consists of correlating the image with the template centred at every second pixel in every second row within a large 41×29 search window. The point that returns the highest correlation in the sparse search becomes the centre of the detailed search. The detailed search has a radius of only 2 pixels and the template is correlated at every location in this small local region, the point of highest correlation giving the new feature location.

2.3 3D Object Tracking

As can be seen in Figure 1, the object tracking system is effectively a feedback control system continually correcting the pose until the error between the projected and observed 2D feature locations is sufficiently small. The forward kinematics and pin-hole camera determine the 2D feature locations resulting from a particular pose. If these 2D locations are insufficiently close to the observed (target) quantities the Jacobian is calculated and a pose correction factor is determined. Another component of the object tracking system that is not shown in Figure 1 is the 3D model of the feature points. This is essential for the calculation of the forward kinematics and the Jacobian.

The 2D feature locations $(\vec{q}_x, \vec{q}_y)$ are a function of the pose $\vec{p}$. The 3D object is tracked using Newton's iterative method, which approximates this function as locally linear for small incremental changes in $\vec{p}$. That is, for a pose $\vec{p}[k]$ at frame k

$$\vec{q}(\vec{p}[k+1]) \approx \vec{q}(\vec{p}[k]) + \frac{d\vec{q}(\vec{p}[k])}{d\vec{p}} q(\vec{c}),$$

where $\vec{q} = (q_{x_1}, q_{y_1}, q_{x_2}, q_{y_2}, q_{x_3}, q_{y_3})^T$ is a concatenation of the elements of $\vec{q}_x$ and $\vec{q}_y$, $\frac{d\vec{q}(\vec{p}[k])}{d\vec{p}}$ is the Jacobian and $\vec{c} = (\vec{p}[k+1] - \vec{p}[k])$ is the pose correction factor.

The pose of the head $\vec{p} = (x, y, z, \theta_x, \theta_y, \theta_z)^T$ is defined as the pose of the *head reference frame* that is rigidly attached to the head, with respect to the *initial head reference frame* as seen in the first image of the sequence.

The individual components of the 3D object tracking system are described below.

3D Model

The 3D model M is a manually constructed rigid model of the three feature points used for head tracking (namely the outer eye corners and one nostril).

$$M = \begin{pmatrix} \vec{m}_1 & \vec{m}_2 & \vec{m}_3 \end{pmatrix},$$

where $\vec{m}_i = (x_i, y_i, z_i, 1)^T$ contains the homogeneous coordinates of the i^{th} feature point in the head reference frame.

Forward Kinematics

The forward kinematics determine the 3D locations $\vec{b}_i$ of the i^{th} feature point in the world coordinate frame (located at the camera origin). For a given head pose $\vec{p}$

$$\vec{b}_i = A_{H \to W} T(\vec{p}) \vec{m}_i.$$

T is the homogeneous transformation matrix representing the pose $\vec{p}$ and $A_{H \to W}$ is the transformation matrix from the head to the world coordinate frames.

Pin-hole Camera Projection

The pin-hole camera projection performs a true perspective projection of the 3D feature points from the world coordinate frame onto the image plane perpendicular to the z-axis. A 3D feature point $\vec{b} = (b_x, b_y, b_z)^T$ is projected onto the image plane at

$$\begin{pmatrix} q_x \\ q_y \end{pmatrix} = -\frac{f}{b_z} \begin{pmatrix} b_x \\ b_y \end{pmatrix},$$

where f is the focal length of the camera.

Jacobian Calculation

The Jacobian J is calculated from the following equation

$$J = \frac{d\vec{q}(\vec{p})}{d\vec{p}} = \begin{pmatrix} \frac{\partial q_1}{\partial p_1} & \frac{\partial q_1}{\partial p_2} & \cdots & \frac{\partial q_1}{\partial p_6} \\ \frac{\partial q_2}{\partial p_1} & \frac{\partial q_2}{\partial p_2} & \cdots & \frac{\partial q_2}{\partial p_6} \\ \vdots & \vdots & & \vdots \\ \frac{\partial q_6}{\partial p_1} & \frac{\partial q_6}{\partial p_2} & \cdots & \frac{\partial q_6}{\partial p_6} \end{pmatrix}$$

where

$$\frac{\partial q_{i\in odd}}{\partial p_j} = \frac{-f}{z_i}\left(\frac{\partial x_i}{\partial p_j} - \frac{x_i}{z_i}\frac{\partial z_i}{\partial p_j}\right)$$
$$\frac{\partial q_{i\in even}}{\partial p_j} = \frac{-f}{z_i}\left(\frac{\partial y_i}{\partial p_j} - \frac{y_i}{z_i}\frac{\partial z_i}{\partial p_j}\right)$$
$$x_i = \vec{A}_1 \cdot (A_{H\to W}\vec{m}_i)$$
$$y_i = \vec{A}_2 \cdot (A_{H\to W}\vec{m}_i)$$
$$z_i = \vec{A}_3 \cdot (A_{H\to W}\vec{m}_i)$$

$\vec{A}_i$ is the i^{th} row of the homogeneous transformation matrix $A = A_{H\to W}T(\vec{p})$, $\vec{p}$ is the pose, f the camera focal length, $A_{H\to W}$ the transformation matrix from the head to the world coordinate frames, and $\vec{m}_i$ the model coordinates of the i^{th} feature.

Calculating Pose Correction

The pose correction factor $\vec{c}$ is calculated using Lowe's algorithm [Lowe, 1991] as follows:

$$\vec{c} = (J^T J + \lambda W^T W)^{-1}(J^T \vec{e} + \lambda W^T W \vec{s}),$$

where J is the Jacobian matrix of $\vec{q}$, $\vec{e}$ is the error vector containing the difference between the observed and projected 2D feature locations, W is a diagonal matrix whose diagonal elements are $W_{ii} = \frac{1}{\sigma_i}$ with σ_i being the standard deviation of the change in parameter $\vec{p}_i$ from one frame to the next, s_i is the desired default value for parameter $\vec{p}_i$, and λ is a scalar weight.

With each iteration of the object tracking loop the above equation is driven to minimise the difference between the measured error and the sum of all the changes in the error resulting from the parameter corrections. The stabilisation technique uses the addition of a small constant to the diagonal elements of $J^T J$ in order to avoid the possibility of this matrix becoming close to singular.

In this algorithm, the standard deviation of parameter changes in consecutive frames represents the limit on the acceleration of each parameter from one frame to the next. For translation parameters, a limit of up to 10 pixels (within the image size of 384×284) is used as the standard deviation, and for rotational parameters 0.1 radians is used. The scalar λ can be increased to strengthen the weight of stabilisation whenever divergence occurs, however, a constant scalar of 0.64 is used in this system to maintain stability throughout the iterations.

3 Mouth Detection and Correction for Pose

Template matching is used to track the corners, top and bottom of the mouth. The 3D pose determined from the head tracker is then used to determine the mouth feature locations in 3D from which the true width and height of the mouth can be determined.

3.1 Mouth Template Matching

The mouth corners are tracked in the same manner as the feature points used for head tracking (see Section 2.2) with the exception that the template search areas are centred at the feature locations determined in the previous frame. The new mouth corner points are not estimated using the method described in Section 2.1 since they are not rigid with respect to the 3D model, and are better localised by their previous locations.

Once the mouth corners are located, the upper mouth edge is searched for along the line joining the mid-point of the mouth corners to the nose template position, and the bottom mouth edge is searched for on the line perpendicularly bisecting the line joining the mouth corners. Locating the mouth edges on these lines avoids the potential problem of the templates drifting along the top and bottom mouth edges, and the computational requirement for template matching is drastically reduced by only searching along a line rather than a 2D region. As the head turns to the side and tilts the mid-point of the line joining the mouth corners no longer corresponds to the centre of the mouth, however, this approximation proved adequate for estimating the height of the mouth.

Initial templates are manually located on the first frame of the sequence, and they are updated every frame in the same manner as the head tracking templates. This updating is essential as the mouth corners look very different when the mouth is open as opposed to when it is closed.

3.2 Determining Mouth Width and Height

Since the mouth is constantly changing shape during speech it cannot be modelled as a rigid 3D object, so the technique used above in Section 2.3 to track the head cannot be applied to find the 3D locations of the mouth corners and edges. Instead the mouth is assumed to lie flat on a plane parallel to the front of the face. The observed 2D mouth feature locations are projected from the image plane onto this face plane, the orientation of which is given by the head pose (as determined by the 3D head tracker). Thus the 3D locations of the mouth features are determined.

The mouth height and width are then calculated as the 3D Euclidean distances between the mouth edges and the mouth corners respectively.

4 Experimentation

The system was tested on a 234 frame video sequence taken at 25Hz. Each frame was 8-bit grey-scale and 384×284 pixels. Templates were 12×12 pixels.

Figures 2 shows six snap shots of the system whilst tracking, the animated face illustrates the current pose and mouth dimensions of the subject.

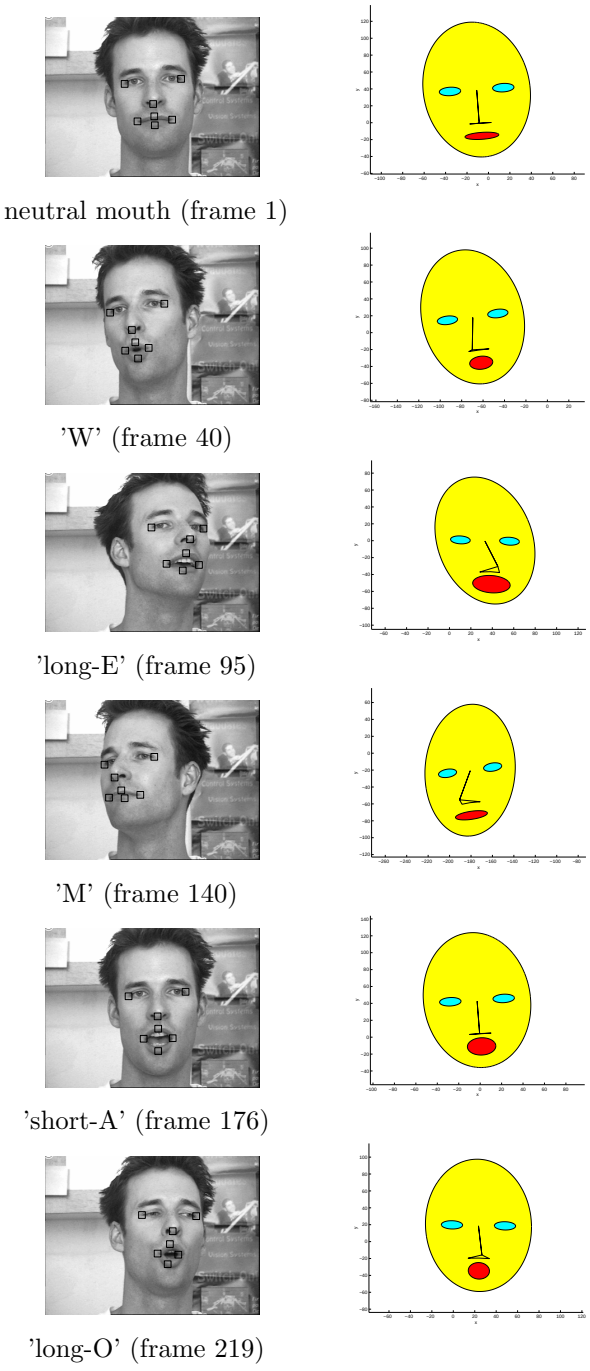

Figure 2: Some snap shots of the system in operation. The phoneme being pronounced and the frame number are indicated.

The quantitative output of the system over the full video sequence is presented in Figure 3. This shows a record over the test sequence of the 3D head pose, the uncorrected mouth dimensions and the corrected mouth dimensions. As expected there is a strong correlation between the amount the mouth width is corrected and

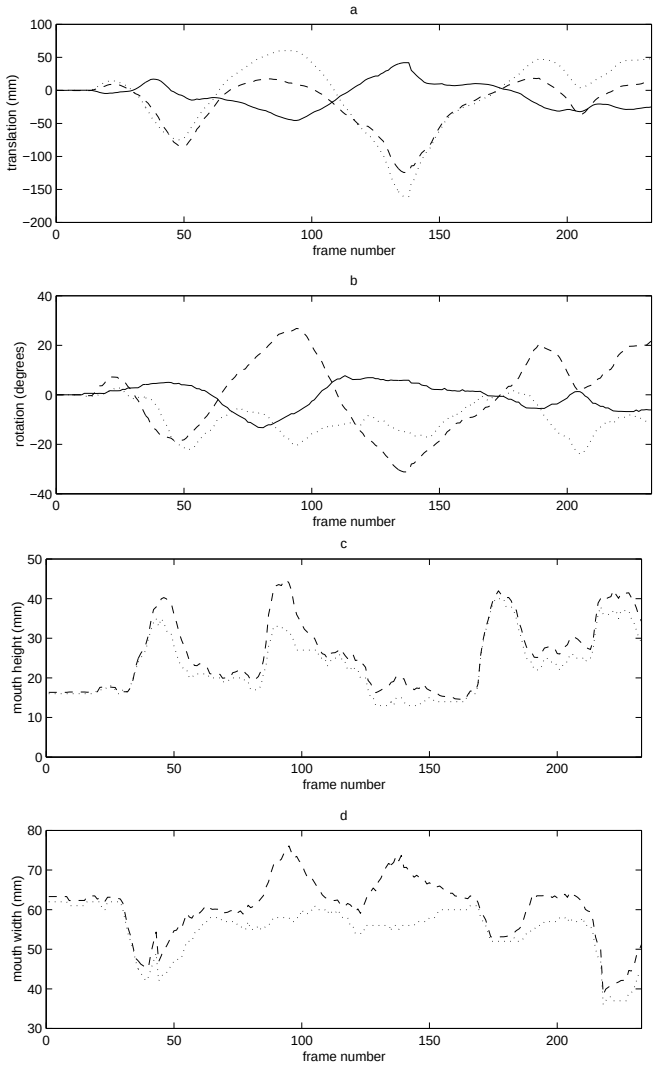

Figure 3: Results of the head and mouth tracking system. (a) Translational movement of head, x, y and z translations are shown as dotted, dashed and solid lines respectively. (b) Head rotation, rotations about x, y and z axis are shown as dotted, dashed and solid lines respectively. (c) Mouth height, uncorrected measurement shown as dotted lines and corrected as dashed lines. (d) Mouth width, uncorrected measurement shown as dotted lines and corrected as dashed lines.

the head's rotation about the vertical y-axis.

The result shows the 3D head tracker recovering the head pose, and the mouth shapes being effectively corrected throughout the sequence. For example, at the frames 40 and 95, the phonemes 'W' and 'long-E' were spoken with the speaker's head nodding. Thus the detected mouth heights for their surrounding frames were corrected as shown in Figure 3(c). In frames 95 and 176 (phonemes 'long-E' and 'M' respectively) the speaker's

head was turned to the side and it was necessary to correct the mouth width. Enunciating these phonemes causes the mouth to widen relative to the neutral mouth width. Figure 3(d) shows that the uncorrected mouth widths for these frames were similar to the neutral mouth width, and that the correction is successfully made to enlarge the widths to represent the actual mouth shape corresponding to these phonemes.

5 Future Work

This paper has presented a technique to track robustly the dimensions of the mouth whilst the speaker's head is moving in 3D. This is an important step in the development of an automatic lip reading system designed for practical application. However, using only mouth width and height, it is possible to distinguish only a small subset of the mouth shapes corresponding to the 28 visual phonemes. Work is currently underway to extend this system to detect the appearances of the teeth and tongue within the mouth area of an image. Linear Predictive Coding (LPC) is being employed to analyse changes of the lip area image over time, while the higher order local autocorrelation feature extraction technique is being used to extract the visual features from the LPC analysed images [Holden and Owens, 2000]. We are currently working on combining the LPC approach with the mouth area extraction technique presented in this paper to detect the appearance of the teeth and tongue within the mouth area. The aim is to build a system capable of recognising the full set of visual phonemes and which is robust to movement of the speaker's head.

Currently, the feature points for head and mouth tracking are manually initialised and we are investigating techniques to locate these features automatically. We also plan to perform a detailed error analysis of the tracking system using an animated mannequin for which a ground truth can be measured.

References

[Bregler and Omohundro., 1995] C. Bregler and S. M. Omohundro. Nonlinear manifold learning for visual speech recognition. In *Proc. of 5th International Conference on Computer Vision*, pages 494–499, 1995.

[Chiou and Hwang, 1994] G. I. Chiou and J. N. Hwang. Image sequence classification using a neural network based active contour model and a hidden markov model. In *Proc. of International Conference on Image Processing*, pages 926–930, 1994.

[Harvey *et al.*, 1997] R. Harvey, I. Matthews, J. A. Bangham, and S. Cox. Lip reading from scale-space measurements. In *IEEE*, pages 582–587, 1997.

[Holden and Owens, 2000] E. Holden and R. Owens. Visual speech recognition using the linear predictive coding analysis and the higher order local autocorrelation feature extraction. In *Proc. of Departmental Conference*. Department of Computer Science, University of Western Australia, 2000.

[Kass *et al.*, 1987] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proc. of IEEE First International Conference on Computer Vision*, pages 259–269, 1987.

[Li *et al.*, 1995] N. Li, S. Dettmer, and M. Shah. Lipreading using eigensequences. In *Proc. of Workshop on Automatic Face and Gesture Recognition*, pages 30–34, 1995.

[Lowe, 1991] D. G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, 1991.

[Yamamoto *et al.*, 1998] E. Yamamoto, S. Nakamura, and K. Shikano. Lip movement synthesis from speech based on hidden markov models. In *Proc. of IEEE International Conference on Face and Gesture Recognition*, pages 154–159, 1998.