Practical Avatar Signing in British Sign Language

Report on UEA activity between 25th October 2004 and 24th January 2005

Overview

UEA work in this period saw the bringing together of data streams from the markerless facial tracker and from the optical body tracker as will be required to generate avatar animation of manual signing with realistic facial animation.

Introduction

Progress was made to bring together data streams from the markerless facial tracker and from the optical body tracker to generate avatar animation of manual signing with facial expression. This synthesis revealed a number of outstanding issues that need to be tackled. Although the current animation falls short of an acceptable solution, we are in a better position to plan the steps that will lead to a practical avatar signing system.

The facial tracker reported in earlier periods seems robust when there is no occlusion of the face. No further work is proposed on the basic tracker framework unless there are difficulties in integrating with body animation. However, work is being undertaken to handle occlusion which causes the tracker to fail in some cases.

Markerless Facial Feature Tracker (Barry Theobald)

The facial feature tracker proves robust when the face is visible, but often fails under occlusion, either due to the face being obscured by the hands or being turned so that only a profile is available for matching.

Failure occurs because the traditional evaluation function used to assess the closeness of fit is unduly influenced by outliers in the data, such as areas that cannot be matched due to occlusion.

A number of different evaluation functions have been implemented and assessed under controlled conditions. Estimation is used to decide whether a pixel is occluded or not. Since the test data has been manipulated artificially, ground truth is available to test the effectiveness of the candidate algorithms

Results of work in progress are reported in the document “Robust Error Functions” attached.

Avatar Research Platform Toolkit (Vince Jennings)

Work is progressing on a skeleton building tool for the ARP Toolkit that will allow the creation of skeletons with the same bone hierarchy as that used by Kaydara, instead of relying on the structure provided by 3D Studio Max‘s Character Studio, which has non-standard linking of the clavicle and the legs and a limited number of bones. At present, the skeletons in avatars exported from Max are re-linked in the ARP Toolkit to achieve the correct structure before import into Kaydara, but still have differences that cannot be resolved by further processing.

Facial animation is implemented via a set of morph targets that are used to deform the facial mesh in response to changes in landmarks placed by the tracker. The provious partial set of facial morphs for the ARP avatar has been completed. While some of the tongue and eye movements would benefit from further refinement, the morphs now allow facial tracker data to drive the avatar facial animation adequately.

Motion Capture and Integration (Judy Tryggvason and Vince Jennings)

Work in the previous period used mainly *ad hoc* approaches to begin the integration of body and face animation. In this period a more systematic approach is being taken to extract information

automatically. Results are shown using some movie files indicating that progress has been made, although the system is well short of an acceptable solution at this stage.

Body animation.

The earlier problem of repeated frames in body animation exported from Kaydara Mocap was found to be due to a corrupt installation and has cleared following a complete re-installation of the software. It is unclear whether the root cause was incompatible software versions or bugs now fixed.

Early body animation data was provided by the BBC in TRC format but the most recent files have been in HTR format after processing to correct erroneous marker assignments, HTR data was mapped onto the avatar skeleton using Mocap 4.03, by plotting the animation from one character to another and then exporting the animation in ARP format for driving an ARP avatar.

The resulting animations have all exhibited a similar problem - the finger joints in the hands have rotations in axes other than their natural bending axis. Due to the complexity of the display of the hands in Mocap (rotation controllers for each joint are displayed that completely obscure the finger animations) it was not possible to see how the fingers were animating until the resulting animation was played on an avatar.

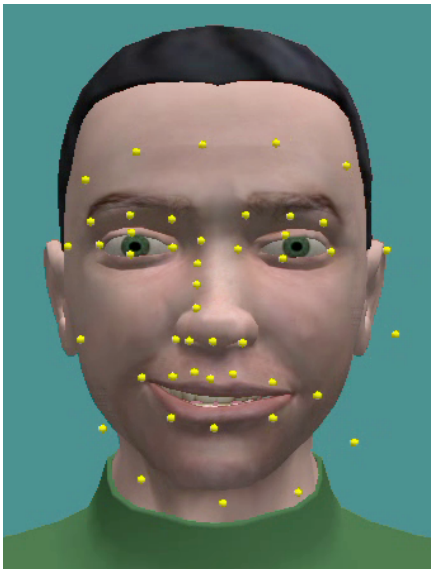
To check the source of the errors, the HTR file was imported into Mocap and then saved, without alteration, in Mocap's FBX format. This allowed it to be imported into Maya, where the bones of the hand are displayed far more clearly. The same errors in finger rotation are still evident which suggests that the errors are in the original HTR file. Errors include finger rotations that jump by 180^{o} and finger joints rotating in the wrong plane.

Face animation

As reported in the previous period, simple combination of facial and body data is ineffective because the 2D face model is effectively coding pose information by distorting the shape of the facial image. To illustrate this, a video is provided of the raw face data with no body animation. The face is deformed using information from the tracker markers. The image below shows the effect of the face being turned and rotated.

Video: raw face data 10fps with markers.avi

For combining with the body data, a head on view is required. Simple techniques were applied that would accommodate rotation of the head around a front to back axis. However the appearance appears unnatural when turning the head left to right, nodding up and down, or moving forwards or backwards. A video is provided applying a simplistic pose correction algorithm. The results are a considerable improvement on the raw data but there is still some spurious movement that corresponds to pose changes that have not been accommodated.



Video: face only pose corrected 10fps with markers.avi

There also appears to be some “jitter” in the data. This could be due to noise in the recorded data. Attempts will be made to smooth the data without losing crucial information.

Since no sequence in the HTR format was available with synchronised video, facial data was combined with body data from a different capture session. It was thought that the lack of synchronisation might be contributing to the unnatural appearance, but when a new sequence was acquired, it became apparent that further work on pose will be required.

Video: raw face data 10fps with markers.avi Face only animated with raw marker data.

Video: face only pose corrected 10fps with markers.avi Face only animated with corrected data.

John Glauert
24 May 2005

Robust Error Functions

The aim of an ordinary least-squares (OLS) fit is to take a linear model with unknown parameters, β, of the form,

$$\mathbf{y} = \beta\mathbf{x} + \epsilon, \tag{1}$$

and estimate the parameters from observations of the dependent variable, $\mathbf{x}$ and the corresponding measurements, $\mathbf{y}$. The vector ϵ is a random error vector. A single noisy observation in $\mathbf{y}$ can generate an inaccurate fit as the objective function, $\sum_i r_i^2$, increases indefinitely with the square of the residuals. Outlying data points often have the largest residual and consequently influence the fit more than the clean (true) data.

M-Estimators

To overcome the problem of outliers significantly influencing the least-squares optimisation, Huber introduced *M-estimators*. The least-squares objective function is replaced by an objective function of the form $\rho(r;\sigma)$, which increases less rapidly than the square of the residual. The ideal error function should always down-weight *all* outliers and always place more significance on *all* inliers in the estimate of the parameter updates. Desirable properties on the form of $\rho(r;\sigma)$ include a function that:

- does not alter the influence of data points with zero error ($\rho(0) = 0$),
- is non-negative ($\rho(r) > 0 \; \forall \; r$),
- is a symmetric ($\rho(r) = \rho(-r)$),
- is a monotonically increasing ($\rho(r_i) > \rho(r_j)$) for $|r_i| > |r_j|$),
- is a monotonically decreasing ($\rho(r_i) < \rho(r_j)$ for ($|r_i| < |r_j|$,
- is a piecewise differentiable — the derivative, $\psi(r) = \rho'(r)$, of the error function is used to calculate a weight for each sample that represents the importance, or reliability, of the sample (*W-estimator*.

Several forms of robust error function have been applied to the method of iteratively re-weighted least-squares (IRLS) fitting. Here we consider four: the Huber, Talwar, Tukey Bisquare and Cauchy functions.

Evaluation of M-Estimators

The task of fitting a robust AAM in an IRLS framework is slightly different from the robust estimation process. During the fit, the robust error function has only a single observation (the error image) from which it must estimate the outliers. Obtaining a single estimate of the scale by analysing the residuals of the elements across the entire error image is not ideal as the distribution of each element is likely to differ in both location and scale. Instead the scale estimate can be pre-computed for each pixel from known "good" data and used during the fit.

The AAM used in this experiment was constructed from 30 hand-labelled images of an individual and contained approximately 10,000 pixels (the appearance images $A(\mathbf{x})$ contain approximately 30,000 elements as the model constructed from colour images). The project-out AAM fitter was then used to fit this model to 300 novel images and the resultant fit checked to ensure the fit was accurate. From the 300 labelled images two scale estimates were calculated for each pixel. Firstly the scaled median of absolute deviations (MAD) as is common in robust

regression, and secondly the variance of the data. The MAD provides a robust estimate of the standard deviation of the residuals, however since we know that none of the (training) images contain occlusion, there is no reason why the variance cannot be used as a scale estimate.

To test the robust error functions, 900 images (none of which were used to build the model or estimate the scale) were randomly selected from a sequence of a person talking (the same individual used in training the model). The test proceeded as follows:

1. Automatically label the face using the project-out AAM fitter[1].
2. Warp the image from fitted landmarks to the base shape.
3. Compute the appearance parameters, $\lambda_i = A_i(\mathbf{x})' \left[I(\mathbf{W}(\mathbf{x}, \mathbf{p})) - A_0(\mathbf{x}) \right]$.
4. Generate the template image $T(\mathbf{x}) = A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x})$.
5. Randomly select N% of the pixels in $I(\mathbf{W}(\mathbf{x}, \mathbf{p}))$ and replace them with pixels randomly selected from an image of scenery. We place no constraints on the distance between the values of the pixels $I(\mathbf{W}(\mathbf{x}, \mathbf{p}))$ and those used to overwrite the them. If a pixel is overwritten with another of a similar colour then the pixel will unlikely be considered occluded — we overcome this by performing ten iterations at several percentage levels across 900 images and average the results. We consider a number of percentage levels, from 1% to 91% in steps of 10%.
6. Compute the error $E(\mathbf{x}) = [I(\mathbf{W}(\mathbf{x}, \mathbf{p})) - T(\mathbf{x})]$. We use $T(\mathbf{x})$ and not $A_0(\mathbf{x})$ to compute the error since the robust fitter must solve for the appearance parameters. If the model converges to the true minimum, then $T(\mathbf{x})$ during the fit will be approximately $I(\mathbf{x})$ (given the reconstruction error).
7. For each of the four loss functions, detect the occluded pixels using the two scale estimates.

The results are given in Section .

Results

The results of occlusion detection using *M-estimators* and the two estimates of scale are shown in Figure 1. The percentage of correctly identified occluded pixels is plotted against the total number of pixels that were occluded. The number of pixels selected to occlude ranged from 1% to 91% in steps of 10% (the total number of pixels in the model is 10,000). It is clear that the performance does not vary as function of the degree of occlusion. This is to be expected since the decision as to whether a pixel is an outlier or not is made on a per-pixel basis using estimates of scale computed from known clean observations. Estimating the scale in this way has over come the problem of the 0.5 breakdown point common in standard robust estimation schemes — algorithms based on the median are only robust up to 50% outliers. Note, since not all of the *M-estimators* assign zero weight for outliers (e.g. the Huber and Cauchy functions), we make the decision that a pixel is unoccluded if it has a corresponding weight $w_i > 0.75$, and occluded if it is less than 0.75..

It is also clear from Figure 1 that significantly better results are obtained when the variance of the data (computed at each individual pixel) rather than the MAD is used to estimate the scale, and of the four *M-estimators* tested here, the Cauchy function was, on average, most reliable.

[1] Again all images were checked to ensure the fit was accurate.

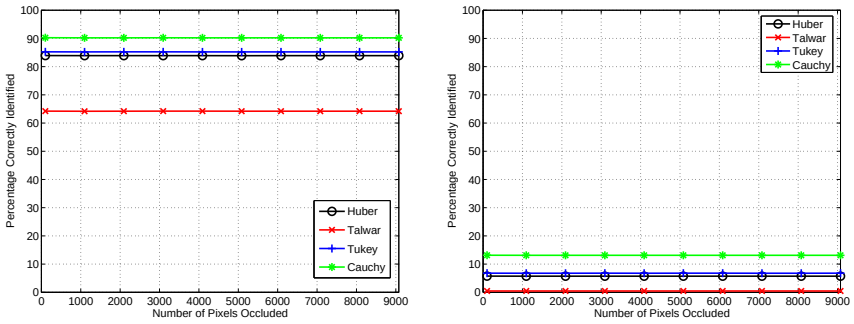


Figure 1: Detection of occluded pixels randomly selected in the image. A) The percentage of correctly identified pixel plotted against the total number of pixels occluded for the *M-estimators* described in Section using the variance as a scale estimate. B) as A) but using MAD/0.6745 as an estimate of the scale.