An Algorithm for Tuning an Active Appearance Model to New Data

T.F. Cootes and C.J. Taylor
Imaging Science and Biomedical Engineering,
The University of Manchester, M13 9PT, UK
t.cootes@manchester.ac.uk

Abstract

Active Appearance Models [5] are widely used to match statistical models of shape and appearance to new images rapidly. They work by finding model parameters which minimise the sum of squares of residual differences between model and target image. Their efficiency is achieved by pre-computing the Jacobian describing how the residuals are expected to change as the parameters vary. This leads to a method of predicting the position of the minima based on a single measurement of the residuals (though in practise the algorithm is iterated to refine the estimate).

However, the estimate of the Jacobian from the training set will only be an approximation for any given target image, and may be a poor one if the target image is significantly different from the training images.

This paper describes a simple method of updating a representation of the Jacobian as the search progresses. This allows us to tune the AAM to the current example. Though useful for matching to a single image, it is particularly powerful when tracking objects through sequences, as it gives a method of tuning the AAM as the search progresses. We demonstrate the power of the technique on a variety of datasets.

1 Introduction

Active Appearance Models (AAMs) [5] are widely used to match statistical models of shape and appearance to new images rapidly. Given a suitably annotated set of example images of a class of objects, we can construct statistical models of their shape and their patterns of intensity (texture) [5]. Such models can reconstruct synthetic images using small numbers of parameters. The AAM algorithm is an efficient method of estimating the parameters of the model which synthesize an image as close as possible to a new target image - it matches the model to the image.

Fast matching is achieved by assuming that the relationship between the residual differences and the parameter displacements is linear, with a Jacobian which can be estimated in advance from the training set. A good estimate of the best dispacement to make can be computed by a simple matrix multiplication of the current residual vector (see below for details).

However, the assumption that the Jacobian is fixed is only an approximation, which can lead to poor convergence for some images, especially those significantly different from the model mean.

The key idea in this paper is that each time we re-compute the residuals during AAM search we obtain information which can be used to update our estimate of the Jacobian. This makes subsequent iterations more likely to converge quickly and accurately, as the Jacobian becomes 'tuned' to the current image data. The resulting search algorithm turns out to have an elegant form, requiring only a series of simple linear operations. These are straightforward to add to any existing AAM implementation, with only a modest increase in computational cost.

The approach is closely related to the quasi-Newton methods for solving least squares problems without derivatives developed by Broyden [2].

The algorithm is also well suited to tracking applications, as it can tune an initial generic AAM to the image sequence as tracking progresses.

In the following we will give an overview of the original AAM approach and related literature. We will outline the derivation of the updating algorithm, and demonstrate how it can be implemented using fast rank-1 updates. We then show results of an extensive series of experiments on several databases of face images, demonstrating that the new algorithm is more accurate, robust and stable than the original AAM.

2 Active Appearance Models

2.1 Statistical Appearance Models

Statistical appearance models are generated by combining a model of shape variation with a model of the texture variation [8, 3].

An appearance model has parameters, $\mathbf{c}$, controlling the shape,$\mathbf{x}$, and texture,$\mathbf{g}$, (in the model frame) according to

$$\begin{aligned} \mathbf{x} &= \bar{\mathbf{x}} + \mathbf{Q}_s \mathbf{c} \\ \mathbf{g} &= \bar{\mathbf{g}} + \mathbf{Q}_g \mathbf{c} \end{aligned} \tag{1}$$

where $\bar{\mathbf{x}}$ is the mean shape, $\bar{\mathbf{g}}$ the mean texture in a mean shaped patch and $\mathbf{Q}_s$,$\mathbf{Q}_g$ are matrices describing the modes of variation derived from the training set.

A shape in the image frame, $\mathbf{X}$, can be generated by applying a suitable global transformation (such as a similarity transformation) to the points in the model frame. The texture in the image frame is generated by applying a scaling and offset to the intensities generated in the model frame.

A full reconstruction is given by generating the texture in a mean shaped patch, then warping it so that the model points lie on the image points, $\mathbf{X}$.

2.2 Interpretation through synthesis

The AAM seeks to minimise a sum-of-squares problem of the form

$$F(\mathbf{p}) = |\mathbf{r}(\mathbf{p})|^2 = \mathbf{r}^T \mathbf{r} \tag{2}$$

where $\mathbf{p}$ contains the t model parameters (the appearance parameters $\mathbf{c}$, together with global pose parameters - see [5]), and $\mathbf{r} = \mathbf{r}(\mathbf{p})$ is a function returning the n residual differences between model and data for parameters $\mathbf{p}$.

Suppose we evaluate the residual at $\mathbf{p}$, with $\mathbf{r} = \mathbf{r}(\mathbf{p})$. To a first order approximation (using a Taylor series), near $\mathbf{p}$ we have

$$\mathbf{r}(\mathbf{p} + d\mathbf{p}) = \mathbf{r} + \mathbf{J}d\mathbf{p} \tag{3}$$

where $\mathbf{J}$ is the (nxt) Jacobian of $\mathbf{r}(\mathbf{p})$ ($J_{ij} = \frac{\partial r_i}{\partial x_j}$).

In this case

$$\begin{aligned} F(\mathbf{p} + d\mathbf{p}) &= (\mathbf{J}d\mathbf{p} + \mathbf{r})^T(\mathbf{J}d\mathbf{p} + \mathbf{r}) \\ &= d\mathbf{p}^T\mathbf{J}^T\mathbf{J}d\mathbf{p} + 2d\mathbf{p}^T\mathbf{J}^T\mathbf{r} + const \end{aligned} \tag{4}$$

Differentiating w.r.t. $\mathbf{p}$ gives gradient

$$0.5\frac{\partial F}{\partial \mathbf{p}} = (\mathbf{J}^T\mathbf{J})d\mathbf{p} + \mathbf{J}^T\mathbf{r} \tag{5}$$

and equating the gradient to zero to find the minimum, gives

$$(\mathbf{J}^T\mathbf{J})d\mathbf{p} = -\mathbf{J}^T\mathbf{r} \tag{6}$$

Thus by solving Equation 6 we can find the displacement, $d\mathbf{p}$, from the current position, $\mathbf{p}$, to take us to the minimum. The key to the efficiency of the AAM is the assumption that $\mathbf{J}$ is constant and can be estimated from the training set as $\mathbf{J}_0$. The solution to Eq.6 is then given by the matrix multiplication $d\mathbf{p} = -\mathbf{R}\mathbf{r}$ where $\mathbf{R}$ is the Pseudo-inverse of $\mathbf{J}_0$, $\mathbf{R} = (\mathbf{J}_0^T\mathbf{J}_0)^{-1}\mathbf{J}_0$.

This leads to the basic AAM search algorithm as follows:

Basic AAM Algorithm

Initialise:		Set $\mathbf{p}_0$, $\mathbf{r}_0 = \mathbf{r}(\mathbf{p}_0)$, $s = 0$, $k_t = 0.125$
Loop:	1)	$d\mathbf{p}_s = -\mathbf{R}\mathbf{r}_s$
	2)	$k = 1.0$
	3)	$\mathbf{p}_{s+1} = \text{Update}(\mathbf{p}_s, kd\mathbf{p}_s)$
	4)	$\mathbf{r}_{s+1} = \mathbf{r}(\mathbf{p}_{s+1})$
	5)	if $\lvert\mathbf{r}_{s+1}\rvert^2 > \lvert\mathbf{r}_s\rvert^2$ and $k > k_t$ then $k = 0.5k$, go to step 3
	6)	s=s+1
Until $k < k_t$ or $s > s_{max}$		

The function 'Update' updates the current estimate of the parameters. Though the combined parameters $\mathbf{c}$ are usually updated linearly, $\mathbf{c} \rightarrow \mathbf{c} + d\mathbf{c}$, the global pose should be updated by composition - see [5]. Matthews and Baker [11] demonstrate that ideally the shape parameters should also be composed, but this isn't always possible.

In practise the algorithm is run in a multi-resolution framework. Note that steps 3-5 implement a crude line search, which can be replaced by a more sophisticated approach. The updating method described below automatically incorporates an efficient line search mechanism implicitly. [1]

[1] We have implemented more sophisticated line searches. Though they can give modest improvements to the basic AAM in some cases, results are a little ambiguous, and they too are outperformed by the new updating algorithm.

2.3 AAM Variants

The AAMs have been widely adopted, and many applications and modifications suggested. Notable amongst these are the Shape-AAMs [4], in which the algorithm drives the shape parameters rather than the appearance parameters, the Inverse-Compositional AAMs [11], in which it is demonstrated that the shape update should be implemented as a composition rather than a simple linear addition, the use of non-linear texture features rather than normalised intensities [10], the use of robust approaches to deal with occlusion efficiently [9] and various natural extensions to 3D.

Bataur and Hayes [1] have also observed that the assumption that the Jacobian is fixed is unsatisfactory, and proposed Adaptive AAMs, in which the Jacobian varies as a linear function of the position in parameter space. This can lead to more robust and accurate convergence, particularly when dealing with examples where there is significant texture variation (such as faces under differing lighting conditions).

The key idea in this paper is to observe that each new evaluation of the residual, $\mathbf{r}(\mathbf{p})$, gives us information which we can use to update our current estimate of $\mathbf{J}$.

3 Updating the Jacobian

Suppose we evaluate the residual at two positions, $\mathbf{p}_0$ and $\mathbf{p}_1$. Let $d\mathbf{p} = \mathbf{p}_1 - \mathbf{p}_0$ and $d\mathbf{r} = \mathbf{r}(\mathbf{p}_1) - \mathbf{r}(\mathbf{p}_0)$. If the Taylor approximation holds, then the i^{th} element of the latter vector,

$$dr_i = \mathbf{j}_i^T d\mathbf{p} = d\mathbf{p}^T \mathbf{j}_i \tag{7}$$

where $\mathbf{j}_i^T$ is the i^{th} row of $\mathbf{J}$. This defines a single linear constraint on this row of $\mathbf{J}$.

Now consider a set of $s+1$ observations, $\{\mathbf{p}_0..\mathbf{p}_s\}$. Let $d\mathbf{p}_j = \mathbf{p}_j - \mathbf{p}_{j-1}$, and $d\mathbf{r}_j = \mathbf{r}(\mathbf{p}_j) - \mathbf{r}(\mathbf{p}_{j-1})$. Let $\mathbf{X} = (d\mathbf{p}_1|...|d\mathbf{p}_s)$ and $\mathbf{R} = (d\mathbf{r}_1|...|d\mathbf{r}_s)$. These give s linear constraints on each row of $\mathbf{J}$,

$$\mathbf{X}^T \mathbf{j}_i = \mathbf{q}_i \tag{8}$$

where $\mathbf{q}_i^T$ is the i^{th} row of $\mathbf{R}$.

If $rank(\mathbf{X}) < t$ then this equation is underconstrained. However, with no additional constraints, we require $\mathbf{j}_i = \mathbf{j}_{0i}$, the i^{th} row of $\mathbf{J}_0$, the Jacobian learnt from the training set.

We can set up a quadratic form which uses $\mathbf{j}_{0i}$ as a regulariser,

$$f(\mathbf{j}_i) = \alpha|\mathbf{X}^T \mathbf{j}_i - \mathbf{q}_i|^2 + |\mathbf{j}_i - \mathbf{j}_{0i}|^2 \tag{9}$$

where α controls the strength of the regularisation.

Differentiating and equating to zero reveals $\mathbf{j}_i$ as the solution to the linear equation

$$(\mathbf{I}_t + \alpha\mathbf{X}\mathbf{X}^T)\mathbf{j}_i = \alpha\mathbf{X}\mathbf{q}_i + \mathbf{j}_{i0} \tag{10}$$

since $\mathbf{j}_i$ is a row of $\mathbf{J}$, and $\mathbf{q}_i$ is a row of $\mathbf{R}$, we have

$$(\mathbf{I}_t + \alpha\mathbf{X}\mathbf{X}^T)\mathbf{J}^T = \mathbf{J}_0^T + \alpha\mathbf{X}\mathbf{R}^T \tag{11}$$

which gives an equation for computing the new estimate of $\mathbf{J}$ given the initial estimate from the training set ($\mathbf{J}_0$) and the residuals sampled at new positions. Notice that with no samples this simplifies to $\mathbf{J} = \mathbf{J}_0$.

3.1 A More Efficient Version

Let

$$\begin{array}{rcl} \mathbf{A} & = & \mathbf{I}_t + \alpha \mathbf{X}\mathbf{X}^T \\ \mathbf{B} & = & \mathbf{J}_0 + \alpha \mathbf{R}\mathbf{X}^T \end{array} \tag{12}$$

Then $\mathbf{A}$ is a txt symmetric matrix, which is positive definite by construction (assuming $\alpha > 0$). Thus it can be efficiently and reliably inverted (eg using Cholesky decomposition).

Thus

$$\mathbf{J}^T = \mathbf{A}^{-1}\mathbf{B}^T \tag{13}$$

.

Suppose our last function evaluation was at $\mathbf{p}$, where $\mathbf{r} = \mathbf{r}(\mathbf{p})$. We can estimate the optimal update,$d\mathbf{p}$, using Eqn 6,

$$\begin{array}{rcl} (\mathbf{J}^T\mathbf{J})d\mathbf{p} & = & -\mathbf{J}^T\mathbf{r} \\ \left(\mathbf{A}^{-1}\mathbf{B}^T\mathbf{B}(\mathbf{A}^{-1})^T\right)d\mathbf{p} & = & -\mathbf{A}^{-1}\mathbf{B}^T\mathbf{r} \\ \left(\mathbf{B}^T\mathbf{B}(\mathbf{A}^{-1})^T\right)d\mathbf{p} & = & -\mathbf{B}^T\mathbf{r} \end{array} \tag{14}$$

Now, let $\mathbf{y} = (\mathbf{A}^{-1})^T d\mathbf{p}$ be the solution to the linear equation

$$\left(\mathbf{B}^T\mathbf{B}\right)\mathbf{y} = -\mathbf{B}^T\mathbf{r} \tag{15}$$

then the optimal update is given by

$$d\mathbf{p} = \mathbf{A}\mathbf{y} \tag{16}$$

thus we can estimate the optimal update using a few matrix multiplications and the solution to a single linear equation (15).

However, some of the multiplications are $O(nt^2)$, which may be expensive. Fortunately we can construct an incremental algorithm which only requires $O(nt)$ operations at each update step.

Suppose that $\mathbf{A}_s$ and $\mathbf{B}_s$ are the versions of $\mathbf{A}$ and $\mathbf{B}$ after adding s constraints (ie $\mathbf{X}$ is txs, and $\mathbf{R}$ is nxs). It is simple to show that if $d\mathbf{p}_s$ and $d\mathbf{r}_s$ are the new constraint vectors to be added, then

$$\begin{array}{rcl} \mathbf{A}_{s+1} & = & \mathbf{A}_s + \alpha_s d\mathbf{p}_s d\mathbf{p}_s^T \\ \mathbf{B}_{s+1} & = & \mathbf{B}_s + \alpha_s d\mathbf{r}_s d\mathbf{p}_s^T \\ \mathbf{C}_{s+1} & = & \mathbf{B}_{s+1}^T\mathbf{B}_{s+1} \\ & = & \mathbf{C}_s + \alpha_s \mathbf{B}_s^T d\mathbf{r}_s d\mathbf{p}_s^T + \alpha_s d\mathbf{p}_s d\mathbf{r}_s^T\mathbf{B}_s + \alpha_s^2 |d\mathbf{r}_s|^2 d\mathbf{p}_s d\mathbf{p}_s^T \end{array} \tag{17}$$

each of which can be done in $O(nt)$ or better, if care is taken in the order of matrix multiplication.

This leads to the following algorithm for AAM search starting from $\mathbf{p}_0$.

Algorithm

Initialise:		Set $\mathbf{p}_0$, $s=0$, $\mathbf{r}_0=\mathbf{r}(\mathbf{p}_0)$	
		$\mathbf{A}_0=\mathbf{I}_t$, $\mathbf{B}_0=\mathbf{J}_0$, $\mathbf{C}_0=\mathbf{B}_0^T\mathbf{B}_0$	
Loop:	1)	Solve $\mathbf{C}_s\mathbf{y}=-\mathbf{B}_s^T\mathbf{r}_s$ for $\mathbf{y}$	$O(t^3)$
	2)	$d\mathbf{p}_s=\mathbf{A}_s\mathbf{y}$	$O(t^2)$
	3)	$\mathbf{p}_{s+1}=\mathbf{p}_s+d\mathbf{p}_s$	
	4)	$\mathbf{r}_{s+1}=\mathbf{r}(\mathbf{p}_{s+1})$	
	5)	$d\mathbf{r}_s=\mathbf{r}_{s+1}-\mathbf{r}_s$	
	6)	$\mathbf{d}=\mathbf{B}_s^T d\mathbf{r}_s$	$O(nt)$
	7)	$\mathbf{A}_{s+1}=\mathbf{A}_s+\alpha_s d\mathbf{p}_s d\mathbf{p}_s^T$	$O(t^2)$
	8)	$\mathbf{B}_{s+1}=\mathbf{B}_s+\alpha_s d\mathbf{r}_s d\mathbf{p}_s^T$	$O(nt)$
	9)	$\mathbf{C}_{s+1}=\mathbf{C}_s+\alpha_s\mathbf{d}d\mathbf{p}_s^T+\alpha_s d\mathbf{p}_s\mathbf{d}^T$ $+\alpha_s^2\lvert d\mathbf{r}_s\rvert^2 d\mathbf{p}_s d\mathbf{p}_s^T$	$O(t^2)$
	10)	if $\lvert\mathbf{r}_{s+1}\rvert^2>\lvert\mathbf{r}_s\rvert^2$ then $\mathbf{p}_{s+1}=\mathbf{p}_s$, $\mathbf{r}_{s+1}=\mathbf{r}_s$	
	11)	s=s+1	
Until happy			

A natural test for convergence would be of the form $|d\mathbf{p}_s|^2<\varepsilon$ for some suitable ε, though a limit on the number of passes through the loop could also be used. Note that no line search step is required. The update to the matrices means that the algorithm does an implicit quadratic line search (ie if it does not go to the true minimum along the current update direction the first time, subsequent iterations will automatically correct for that).

It will usually be possible to solve the linear equation in step 1 using Cholesky decomposition, as $\mathbf{C}_s$ is symmetric and (usually) positive definite.

For larger numbers of parameters, t, it is worth storing $\mathbf{C}_s$ as its Cholesky decomposition. The update in step 9 can implemented as combination of rank-1 updates and downdates to the Cholesky representation. In that case the linear equations can be solved in $O(t^2)$ using a backsubstitution.

In the following we use $\alpha_s=(\delta+|d\mathbf{p}_s|^2)^{-1}$, where δ is small, included to avoid numerical instability after small steps. This gives a weight to the new measurement equal to the contribution from the training (since $\sqrt{\alpha}d\mathbf{p}_s$ would be a unit vector, and the estimate would be equivalent to numeric differentiation, displacing by ± 0.5 units along the chosen direction in parameter space).

4 Results of Experiments

We demonstrate the power of the algorithm by applying it to a variety of different face databases, and comparing its performance with that of the Basic AAM.

4.1 Static Face Images

4.1.1 People in the Model

We constructed an appearance model from 172 face images (each of a different person with no facial hair or glasses) from session 1 of the XM2VTS face database [12]. The model used 50 combined appearance modes. Our test set was 160 images of the same

Search algorithm	Mean Boundary Error (pixels)	Median (pixels)	90%-ile (pixels)
Basic	5.17 ± 0.08	2.98	10.08
Updating	4.23 ± 0.07	2.67	8.11

Table 1: Boundary errors when searching new images of people included in the model

Search algorithm	Mean Boundary Error (pixels)	Median (pixels)	90%-ile (pixels)
Basic	5.52 ± 0.07	3.26	13.31
Updating	3.70 ± 0.04	2.65	6.86

Table 2: Boundary errors when searching images of people not included in the model

people from session 2 of the database. For each image we systematically displaced the (mean) model from the known best position by [-20,-10,0,10,20] pixels in x and y (25 displacements in total) and performed a 3-level multi-resolution AAM search, allowing at most 10 iterations at each resolution. We then recorded the average point-to-boundary error (the distance between each model point and the curve passing through the point in the hand labelled data). Results are shown in Table 1. The updating approach leads to a considerable improvement in both mean, median and 90%-ile.

4.1.2 People not in the Model

We performed a similar experiment, this time arranging that the people in the test set were not in the training set. We constructed the model from 162 images of 82 people, again using examples from sessions 1 and 2 of the XM2VTS dataset. We then tested on 175 images of different people from the set.

We used the same displacement protocol as before. Results are shown in Table 2, and show a considerable improvement is obtained by using the updating approach.

To test sensitivity to initial conditions we performed another experiment on the same data. We initialised the search from a range of displacements in x, and for each we evaluated the median boundary error and the proportion of searches converging (defined as those with a final boundary error less than 5 pixels). Results are summarised in Figure 1). The updating algorithm leads to significantly better convergence rates.

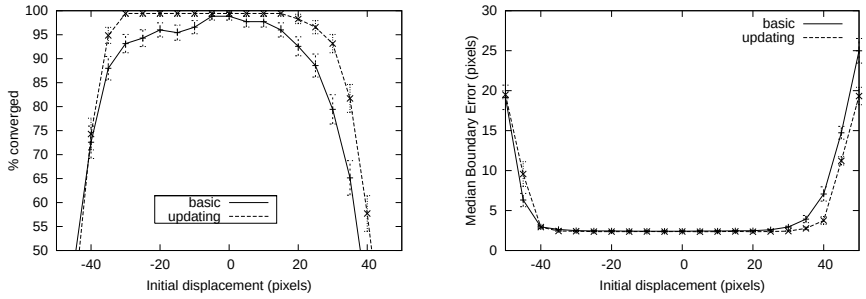

Figure 1: Plot of convergence and median error vs initial displacement

Search algorithm	Mean Boundary Error (pixels)	Median (pixels)	90%-ile (pixels)
Basic	7.96 ± 0.09	6.81	14.57
Updating	6.82 ± 0.09	5.08	10.99

Table 3: Boundary errors when searching images of a new individual under different conditions to those in the training set

The updating algorithm outperforms the original method considerably.

4.1.3 Fitting a Generic Face Model to an Individual

We trained a model from 480 images of 100 different people, including multiple expressions and head pose variations of each (examples are given in Figure 2). We then repeated the fitting experiments above on a dataset of 100 images of a new individual taken from a sequence of that person talking. This sequence was taken under different lighting conditions and with a different camera. Figure 3 shows the results of searching on two of the frames. In this case the model is unable to accurately follow the shape of the chin (due to the small training set and the truncation of the modes used). However, the internal facial features are quite accurately located, and the reconstruction is convincing.

Results of the displacement experiment are summarised in Table 3. and again demonstrate the superiority of the new algorithm.

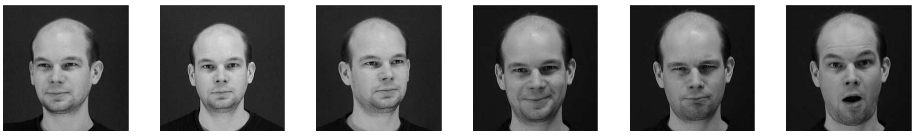

Figure 2: Examples of faces with varying viewpoint and expression

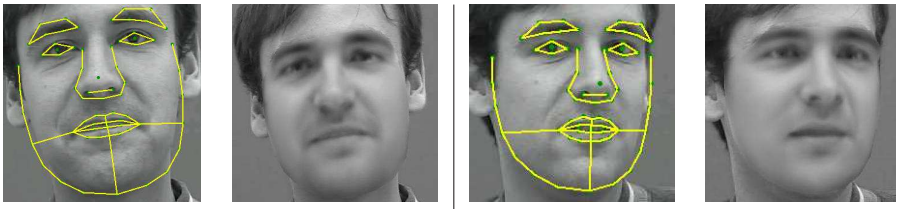

Figure 3: Search results and reconstructions on images of a new individual

4.2 Face Tracking Experiments

The face model described in section 4.1.3 was used to track the face of a new individual (Fig.3) with different versions of the AAM algorithm. The model was initialised by fitting it to the (manually annotated) points in the first frame. The search was performed on each

Search algorithm	Mean Boundary Error (pixels)	Median (pixels)	convergence failures (%)
Basic	5.81 ± 0.05	5.42	0.4%
Updating	3.69 ± 0.01	3.64	0.0%

Table 4: Results of tracking a 1000 frame sequence of a new individual

subsequent frame by initialising at the final result for the previous frame. The boundary error between the model and the annotated points was recorded. When the error rose above 10 pixels, the search was assumed to be diverging, and the model was re-initialised to the labelled points. Table 4 summarises the results for the different methods when tracking through 1000 frames. Figure 4 shows the boundary error for each frame for the methods. The updating algorithm gives much more accurate, stable and robust results than the original algorithm.

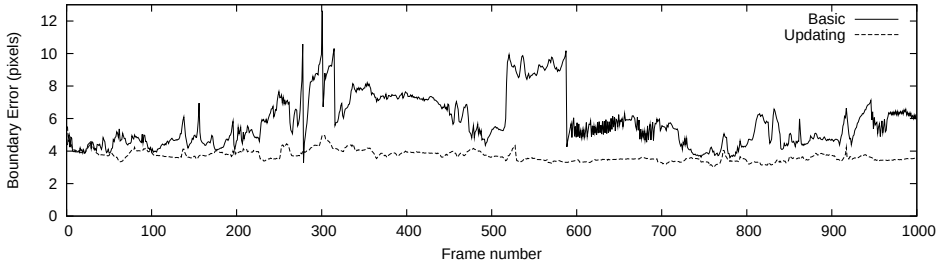

Figure 4: Boundary error against frame number when tracking a face

5 Discussion and Conclusions

We have presented an efficient algorithm for 'tuning' an AAM to a particular dataset. Every evaluation of the residual difference between model and image can be used to update our current estimate of the Jacobian of $\mathbf{r}(\mathbf{p})$, leading to more accurate estimates of updates in subsequent steps.

The approach described above is closely related to the quasi-Newton methods for solving least squares problems without derivatives developed by Broyden [2]. He proposed an algorithm which directly updates the pseudo-inverse of the Jacobian. More sophisticated variants are described by Xu [7]. In future work we will explore using such variants for the AAM matching problem. The advantage of working with a version of the Jacobian is that it allows us to incorporate priors in a natural extension of the work described in [6].

With our implementation, the updating algorithm took approximately twice the time of the basic algorithm, indicating it is dominated by the extra $O(nt)$ operations required to update $\mathbf{B}$ (step 8). It would be possible to significantly reduce this by updating only the columns of $\mathbf{J}$ associated with the more important modes.

It is possible that repeated steps $d\mathbf{p}_s$ do not adequately span the full parameter space, which could result in poorer estimates of the Jacobian in some directions. An interesting stochastic variant of the algorithm would be to occasionally replace step 1 (estimation

of $d\mathbf{p}$) with the generation of a small random perturbation. This would ensure that over time the full parameter space was spanned, and may be particularly useful in tracking applications.

The algorithm has been described for the basic version of the AAM, but is equally applicable to a most variants, including [4, 11, 10, 9], though there may be a significant loss of efficiency in those methods which work by projecting out the texture variation [4, 11].

Overall, the approach is simple and elegant. It is easy to add to any existing AAM framework. Our experience suggests that the new algorithm can give more accurate and stable results than the basic AAM. We encourage anyone currently using AAMs to implement it and explore its performance on their own data.

References

[1] A. Bataur and M. Hayes. Adaptive active appearance models. *IEEE Trans. Imaging Processing*, 14:1707–21, 2005.

[2] C. Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of Computation*, 30:1–17, 1976.

[3] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In H.Burkhardt and B. Neumann, editors, 5^{th} *European Conference on Computer Vision*, volume 2, pages 484–498. Springer, Berlin, 1998.

[4] T. F. Cootes, G. J. Edwards, and C. J. Taylor. A comparative evaluation of active appearance model algorithms. In P. Lewis and M. Nixon, editors, 9^{th} *British Machine Vision Conference*, volume 2, pages 680–689, Southampton, UK, Sept. 1998. BMVA Press.

[5] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[6] T. F. Cootes and C. J. Taylor. Constrained active appearance models. In 8^{th} *International Conference on Computer Vision*, volume 1, pages 748–754. IEEE Computer Society Press, July 2001.

[7] C.X.Xu. Hybrid method for nonlinear least-square problems without calculating derivatives. *Journal of Optimization Theory and Applications*, 65(3):555–574, 1990.

[8] G. Edwards, A. Lanitis, C. Taylor, and T. Cootes. Statistical models of face images - improving specificity. *Image and Vision Computing*, 16:203–211, 1998.

[9] R. Gross, I. Matthews, and S. Baker. Constructing and fitting active appearance models with occlusion. In *Proceedings of the IEEE Workshop on Face Processing in Video*, June 2004.

[10] I.M.Scott, T.F.Cootes, and C.J.Taylor. Improving appearance model matching using local image structure. In 18^{th} *Conference on Information Processing in Medical Imaging*, pages 258–269. Springer-Verlag, 2003.

[11] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135 – 164, November 2004.

[12] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. XM2VTSdb: The extended m2vts database. In *Proc. 2nd Conf. on Audio and Video-based Biometric Personal Verification*, pages 72–77. Springer Verlag, 1999.