To Appear in Image and Vision Computing, 2005

Generic vs. Person Specific Active Appearance Models

Ralph Gross, Iain Matthews, and Simon Baker

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Active Appearance Models (AAMs) are generative parametric models that have been successfully used in the past to model faces. Anecdotal evidence, however, suggests that the performance of an AAM built to model the variation in appearance of a single person across pose, illumination, and expression (a Person Specific AAM) is substantially better than the performance of an AAM built to model the variation in appearance of many faces, including unseen subjects not in the training set (a Generic AAM). In this paper we present an empirical evaluation that shows that Person Specific AAMs are, as expected, both easier to build and more robust to fit than Generic AAMs. Moreover, we show that: (1) building a generic shape model is far easier than building a generic appearance model, and (2) the shape component is the main cause of the reduced fitting robustness of Generic AAMs. We then proceed to describe two refinements to Generic AAMs to improve their performance: (1) a refitting procedure to improve the quality of the ground-truth data used to build the AAM and (2) a new fitting algorithm. For both refinements we demonstrate dramatically improved fitting performance. Finally, we evaluate the effect of these improvements on a combined model construction and fitting task.

1 Introduction

Active Appearance Models (AAMs) [4] are generative parametric models commonly used to model faces. Depending on the task at hand AAMs can be constructed in different ways. For example, we might build an AAM to model the variation in the appearance of a single person across pose, illumination and expression. Such a *Person Specific AAM* might be useful for interactive user interface applications that involve head pose estimation, gaze estimation, or expression recognition. Alternatively, we might attempt to build an AAM to model any face, including unseen subjects not in the training set. The most obvious use of such a *Generic AAM* would be face recognition.

Anecdotal evidence suggests that Person Specific AAMs perform substantially better than Generic AAMs. The performance of an AAM depends on two steps: (1) Modelling: How well is the AAM able to model (or generate) images in the class under consideration and (2) Fitting: How robustly can the AAM be fit to a novel input image? AAMs consist of two components: (1) a shape component, and (2) an appearance component. Natural questions then include: "is it harder to build a generic shape model that models the shape of any face well, or is building a generic appearance model harder?" and "what makes fitting harder, a large shape model, or a large appearance model?

We begin in Section 4 by studying the construction of AAMs. We present an empirical evaluation that shows that Person Specific AAMs are indeed easier to build than Generic AAMs. Moreover, building a generic shape model is relatively easy, whereas building a generic appearance model is what makes building a Generic AAM hard.

We continue in Section 5 by studying the fitting of AAMs. We present an empirical evaluation that shows that Person Specific AAMs are indeed easier to fit than Generic AAMs. We also show that what makes fitting a Generic AAM hard is the increased size of the shape model, more so than the increased size of the appearance model.

In Section 6 we proceed to describe two refinements to Generic AAMs to improve their performance. We first propose a refitting procedure (closely related to our automatic AAM construction algorithm [3]) to improve the quality of the ground-truth data used to build the AAM (Section 6.1). We then introduce a new fitting algorithm, the Simultaneous Inverse Compositional Algorithm [1] (Section 6.2). For both refinements we demonstrate dramatically improved fitting performance on the same data used in Section 5.

The fitting experiments in Sections 5 and 6 were conducted on *seen data* (i.e. data in the training set.) The experiments were conducted this way to remove the effect of model construction from the model fitting experiments. If a fitting algorithm fails, it fails because it fell into a local minima, not because the AAM cannot model the input image. In Section 7 we combine the experiments in Sections 4–6 and perform an evaluation of how much the two improvements to Generic AAMs help the performance of AAM fitting on *unseen data* (i.e. data not in the training set.)

2 Background: Active Appearance Models

We begin with a brief review of Active Appearance Models (AAMs) [4]. We briefly define them, explain how they are constructed from training data, and describe our efficient "Project Out" fitting algorithm [10].

2.1 Definition and Model Construction

The *2D shape* of an AAM is defined by a 2D triangulated mesh and in particular the vertex locations of the mesh. Mathematically, we define the shape $\mathbf{s}$ of an AAM as the 2D coordinates of the n vertices that make up the mesh: $\mathbf{s} = (x_1, y_1, x_2, y_2, \ldots, x_n, y_n)^{\mathrm{T}}$. AAMs allow linear shape variation. This means that the shape matrix $\mathbf{s}$ can be expressed

as a base shape $\mathbf{s}_0$ plus a linear combination of m shape matrices $\mathbf{s}_i$:

$$\mathbf{s} \;=\; \mathbf{s}_0 + \sum_{i=1}^{m} p_i \, \mathbf{s}_i \tag{1}$$

where the coefficients p_i are the shape parameters. AAMs are normally computed from training data consisting of a set of images with the shape mesh (usually hand) marked on them [4]. The training shapes are then geometrically aligned using the *Procrustes* algorithm [6]. Principal Component Analysis (PCA) [8] is then applied to the aligned training meshes. The base shape $\mathbf{s}_0$ is the mean shape and the matrices $\mathbf{s}_i$ are the (reshaped) eigenvectors corresponding to the m largest eigenvalues.

The *appearance* of the AAM is defined within the base mesh $\mathbf{s}_0$. Let $\mathbf{s}_0$ also denote the set of pixels $\mathbf{u} = (u, v)^{\mathrm{T}}$ that lie inside the base mesh $\mathbf{s}_0$, a convenient abuse of terminology. The appearance of the AAM is then an image $A(\mathbf{u})$ defined over the pixels $\mathbf{u} \in \mathbf{s}_0$. AAMs allow linear appearance variation. This means that the appearance $A(\mathbf{u})$ can be expressed as a base appearance $A_0(\mathbf{u})$ plus a linear combination of l appearance images $A_i(\mathbf{u})$:

$$A(\mathbf{u}) \;=\; A_0(\mathbf{u}) + \sum_{i=1}^{l} \lambda_i \, A_i(\mathbf{u}) \tag{2}$$

where the coefficients λ_i are the appearance parameters. The appearance images A_i are usually computed by applying PCA to the shape normalized training images [4, 10].

2.2 Model Fitting

Fitting a AAM is usually formulated [10] as minimizing the sum of squares difference between the model instance $A(\mathbf{x}) = A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x})$ and the input image warped back onto the base mesh $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$:

$$\sum_{\mathbf{x} \in \mathbf{s}_0} \left[A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) \right]^2 \tag{3}$$

where the sum is performed over all of the pixels $\mathbf{x}$ in the base mesh $\mathbf{s}_0$. In this equation, the warp $\mathbf{W}$ is the piecewise affine warp from the base mesh $\mathbf{s}_0$ to the current AAM shape $\mathbf{s}$ defined by the vertices. Hence, $\mathbf{W}$ is a function of the shape parameters $\mathbf{p}$. For ease of notation, in this paper we have omitted mention of the 2D similarity transformation that is used to normalize the shape of an AAM. In [10] we showed how to include this warp into $\mathbf{W}$. The goal of AAM fitting is to minimize the expression in Equation (3) simultaneously with respect to the shape $\mathbf{p}$ and appearance $\boldsymbol{\lambda}$ parameters. The "Project-Out" Inverse Compositional Algorithm [2] and its extension to 2D AAMs was proposed in [10]. See Figure 1 for a summary. The algorithm performs the non-linear optimization of Equation (3) in two steps (similar to Hager and Belhumeur [7]). The shape parameters $\mathbf{p}$ are found through non-linear optimization in a subspace in which the appearance variation can be ignored. This is achieved by "projecting out" the appearance variation from the *steepest-descent images*:

$$\mathbf{SD}_{\mathrm{ic}}(\mathbf{x}) = \boldsymbol{\nabla} A_0 \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \tag{4}$$

The Project-Out Inverse Compositional Algorithm

Pre-Computation:

(P1) Evaluate the gradient of the base appearance $\boldsymbol{\nabla} A_0$

(P2) Evaluate the Jacobian of the warp $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $(\mathbf{x}; \mathbf{0})$

(P3) Compute the steepest descent images $\mathbf{SD}_{\mathrm{ic}}(\mathbf{x})$ (Eqn. (4))

(P4) Project out appearance from $\mathbf{SD}_{\mathrm{ic}}(\mathbf{x})$ (Eqn. (5))

(P5) Compute the Hessian matrix H_{po} (Eqn. (7))

Iterate:

(I1) Warp I with $\mathbf{W}(\mathbf{x}; \mathbf{p})$ to compute $I(\mathbf{W}(\mathbf{x}; \mathbf{p}))$

(I2) Compute the error image $E(\mathbf{x}) = I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - \mathbf{A_0}(\mathbf{x})$

(I3) Compute $\sum_{\mathbf{x}} \mathbf{SD}^{\mathrm{T}}_{\mathrm{po}}(\mathbf{x}) E(\mathbf{x})$

(I4) Compute $\Delta \mathbf{p} = -H^{-1}_{\mathrm{po}} \sum_{\mathbf{x}} \mathbf{SD}^{\mathrm{T}}_{\mathrm{po}}(\mathbf{x}) E(\mathbf{x})$

(I5) Update the warp $\mathbf{W}(\mathbf{x}; \mathbf{p}) \leftarrow \mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta \mathbf{p})^{-1}$

Optional Post-Computation of Appearance Parameters:

(A1) Compute $\lambda_i = \sum_{\mathbf{x} \in \mathbf{s}_0} A_i(\mathbf{x}) \cdot [I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - A_0(\mathbf{x})]$

Figure 1: The Project-Out Inverse Compositional Algorithm [10].

by computing:

$$\mathbf{SD}_{\mathrm{po}}(\mathbf{x}) = \mathbf{SD}_{\mathrm{ic}} - \sum_{i=1}^{m} \left[\sum_{\mathbf{x} \in \mathbf{s}_0} A_i(\mathbf{x}) \mathbf{SD}_{\mathrm{ic}}(\mathbf{x}) \right] A_i(\mathbf{x}). \tag{5}$$

Equation (5) requires the appearance images A_i to be orthonormal. In each iteration of the algorithm, the input image is warped with the current estimate of the warp to estimate $I(\mathbf{W}(\mathbf{x}; \mathbf{p}))$, the base appearance subtracted to give the error image $E(\mathbf{x}) = I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - A_0(\mathbf{x})$, and the incremental parameter updates computed:

$$\Delta \mathbf{p} \;=\; -H^{-1}_{\mathrm{po}} \sum_{\mathbf{x} \in \mathbf{s}_0} \mathbf{SD}_{\mathrm{po}}(\mathbf{x}) [I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - A_0(\mathbf{x})] \tag{6}$$

using the *Project-Out Hessian*:

$$H_{\mathrm{po}} \;=\; \sum_{\mathbf{x} \in \mathbf{s}_0} \mathbf{SD}_{\mathrm{po}}(\mathbf{x})^{\mathrm{T}} \mathbf{SD}_{\mathrm{po}}(\mathbf{x}). \tag{7}$$

The incremental warp $\mathbf{W}(\mathbf{x}; \Delta \mathbf{p})$ is then *inverted* and *composed* with the current estimate to give the new estimate $\mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta \mathbf{p})^{-1}$. In an optional post-computation step, the

appearance parameters $\boldsymbol{\lambda}$ can then be computed as:

$$\lambda_i \;=\; \sum_{\mathbf{x}\in\mathbf{s}_0} A_i(\mathbf{x}) \cdot [I(\mathbf{W}(\mathbf{x};\mathbf{p})) - A_0(\mathbf{x})]\,. \tag{8}$$

If there are n shape parameters, m appearance parameters, and N pixels in the base appearance A_0, the pre-computation takes time $\mathrm{O}(n^2 \cdot N + m \cdot N)$ where the slowest step is the computation of the Hessian in Step P5 which alone takes time $\mathrm{O}(n^2 \cdot N)$. The online cost per iteration is just $\mathrm{O}(n \cdot N + n^3)$ and the post-computation cost is $\mathrm{O}(m \cdot N)$. In all cases we iterate the algorithm until convergence or for a sufficient (fixed) number of times. A implementation of this algorithm in "C" runs at 230 frames per second on a dual 3GHz Pentium 4 Xeon for typical values of n, m and N [10].

3 Evaluation Datasets

We assembled three datasets which separately vary illumination, pose and identity. For the illumination dataset we recorded a single subject in a static frontal pose and neutral expression while smoothly changing the position of a lamp illuminating the face. We randomly selected 100 images from this sequence for the experiments. This data is used to construct Person Specific AAMs. It contains a large amount of appearance variation, but little or no shape variation. In the second dataset the same subject was recorded under constant illumination and neutral expression while smoothly changing head pose. We randomly selected 100 images from the sequence for evaluation. Again, this data is used to build Person Specific AAMs. Unlike the illumination dataset, the pose dataset contains a lot of shape variation. Due to the non-uniform lighting and the presence of specularities, some appearance variation is visible as well. For the identity dataset, we chose 100 different subjects from the **fa** subset of the FERET database [11]. This data is used to build Generic AAMs. This dataset contains both shape and appearance variation due to the variation across face identities. Even though the images were taken from the same subset of FERET, differences in facial expression, face pose and illumination are also present. Figure 2 shows three example images from each dataset. To ground-truth the data the 68 vertex locations of the shape mesh for all 300 images were marked by hand.

4 Evaluating Model Construction

How many training images does it take to build a truly Generic AAM? Which requires more training data, building the shape component or building the appearance component? In order to quantify AAM construction performance we experimentally evaluate how well an AAM can model *unseen* data based on a training set of the same type. In the experiments we separately evaluate each dataset and the shape and appearance components of the AAMs.

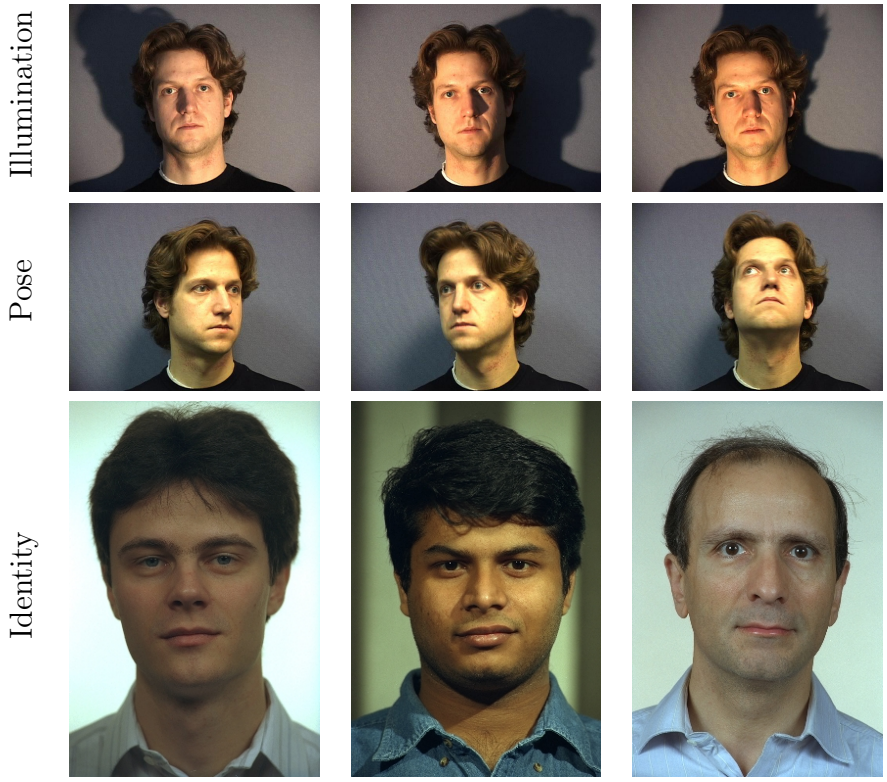
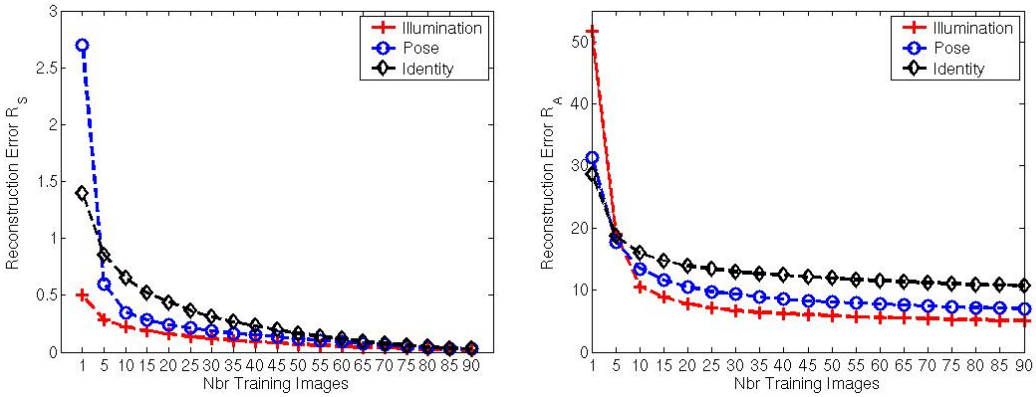


Figure 2: Datasets. Top Row: Illumination Dataset. The subject was recorded with constant frontal pose and neutral expression while smoothly changing the position of a lamp illuminating the face. Middle Row: Pose Dataset. The same subject was recorded under constant illumination while smoothly changing head pose. Bottom Row: Identity set: We randomly selected 100 subjects from the **fa** subset of the FERET database [11].

4.1 Experiment Description

We randomly select a varying number of training images from the dataset to build shape and appearance models. For all models we retain enough variance to explain 95% of the training data. (Retaining 95% of the variance is a standard procedure in the AAM literature. In Section 5.2 we employ a better method to choose the dimensionality of the models used in the fitting experiments. In the model construction experiments in this section, the best way to compare the datasets is to take the same amount of variance for each dataset.) We then evaluate the reconstruction error of a fixed number of images from an *independent* test set (although extracted from the same dataset.) In order to calculate the reconstruction error for a test shape $\mathbf{s}$ and appearance A, we compute the shape parameters $p_1, \ldots, p_m$ and the appearance parameters $\lambda_1, \ldots, \lambda_l$ by projecting $\mathbf{s}$ and A into the shape and appearance eigenspaces. The reconstruction errors are then defined by:

$$R_S = \left\| \mathbf{s} - (\mathbf{s_0} + \sum_{i=1}^{m} p_i \mathbf{s_i}) \right\|_2 \qquad R_A = \left\| A - (A_0 + \sum_{i=1}^{l} \lambda_i A_i) \right\|_2 \tag{9}$$



(a) Shape (b) Appearance

Figure 3: Shape and appearance reconstruction errors for the illumination, pose and identity datasets. We compute the reconstruction error by projecting independent test data into the eigenspace spanned by training sets of varying size, reconstructing the data using the eigenspace representation and measuring the Euclidean distance between original and reconstructed data. See Equation (9).

where $\|\cdot\|_2$ is the Euclidean L2 Norm. R_S is measured in pixels^2 and R_A in grey $-$ levels^2.

4.2 Experiment Results

Figure 3(a) plots the shape reconstruction errors for all three datasets against a varying number of training images. The illumination dataset theoretically has zero shape variation. Hence the reconstruction error for a single training image (0.5 pixels) can be attributed to errors in ground-truthing. We use this threshold to determine how many training images are needed for the pose and identity sets to model unseen data. The error over the pose set falls below 0.5 after 5 training images, which is consistent with intuition and recent theoretical results showing that at most 6 2D shape vectors are needed to model a single rigid 3D face [12]. For the identity set 15 training images are needed to reach this level of modelling accuracy. We can therefore conclude that (1) it is possible to build a generic shape model and (2) as few as 15 training images are needed for a generic shape model. However, this only applies to generic shape models for frontal faces. More training images will be necessary to build a generic shape model for faces under varying poses.

Figure 3(b) plots the appearance reconstruction error. Again we can use the reconstruction error over the illumination set as guideline to determine when the models only explain noise due to errors in (shape) ground-truthing and appearance model interpolation. This holds since faces under fixed pose but varying illumination can be modelled using a low dimensional subspace [9]. The reconstruction error over the pose set is actually slightly higher than over the illumination set, possibly due to the more difficult ground-truthing and the non-uniform illumination. The reconstruction error over the identity set always stays well above the level of either the illumination or the pose set. This observation holds even if we

Table 1: Appearance reconstruction error R_A for the illumination, pose and identity datasets for 90 and 190 training images. Even using 190 training images in the identity set, the reconstruction error stays well above the error for the illumination and pose sets.

	Illumination (90)	Pose (90)	Identity (90)	Identity (190)
R_A	5.06	6.99	10.67	9.14

expand the training set to 190 images. See Table 1 for numerical results. Overall we can conclude that it is much more difficult to build a generic appearance model than a generic shape model. 190 images are not enough, even for the frontal images in FERET which contain little expression variation.

4.3 Experiment Conclusions

Our experimental results show that it is relatively easy to build a generic shape model for frontal faces. With as few as 15 training images, the shape model is able to model unseen faces with accuracy comparable to that for a Person Specific illumination model with essentially no shape variation. Using the argument in [12], 15×6 images should be sufficient to model the shape of all neutral faces across pose. A few more images may be needed to model expression variation, but overall building a Generic shape model is probably possible with a few hundred images. However, the same does not hold for a Generic appearance model. 200 images are nowhere near enough to build a Generic model that approaches the quality of a Person Specific appearance model. Although we cannot provide an upper bound on how many images will be required, we predict that many thousands will be required.

5 Evaluating Model Fitting

In this section we quantify model fitting performance by empirically evaluating how well an AAM can be fit to an image using the "Project-Out" algorithm described in Section 2.2. Specifically we evaluate the fitting performance on *seen data*; i.e. images in the training set used to build the AAM. Although this choice may seem strange, the fitting algorithm is deliberately tested on the training image to separate the effects of model construction and fitting. By conducting the experiments in this way, we know that the AAM is able to model the input image it is being fit to. If it fails to fit, it is due to the difficulty of the fitting process rather than the inability of the AAM to model the image. In Section 7 we present results on *unseen data*; i.e. images not in the training set. The results in Section 7 investigate the trade-off between the difficulty in modeling and fitting.

Our evaluation methodology is similar to the one used in [10]. For a given AAM and test image we randomly perturb the (hand-marked) ground truth shape and similarity transform parameters. We then use these perturbed parameter values as the initial parameter estimates for the fitting algorithm. We then run the fitting algorithm on a large number of such trials

and record the *average frequency of convergence* by calculating how often the algorithm converges after 20 iterations. In [10], we found that the "Project-Out" algorithm typically converges well within 20 iterations, if it converges at all.

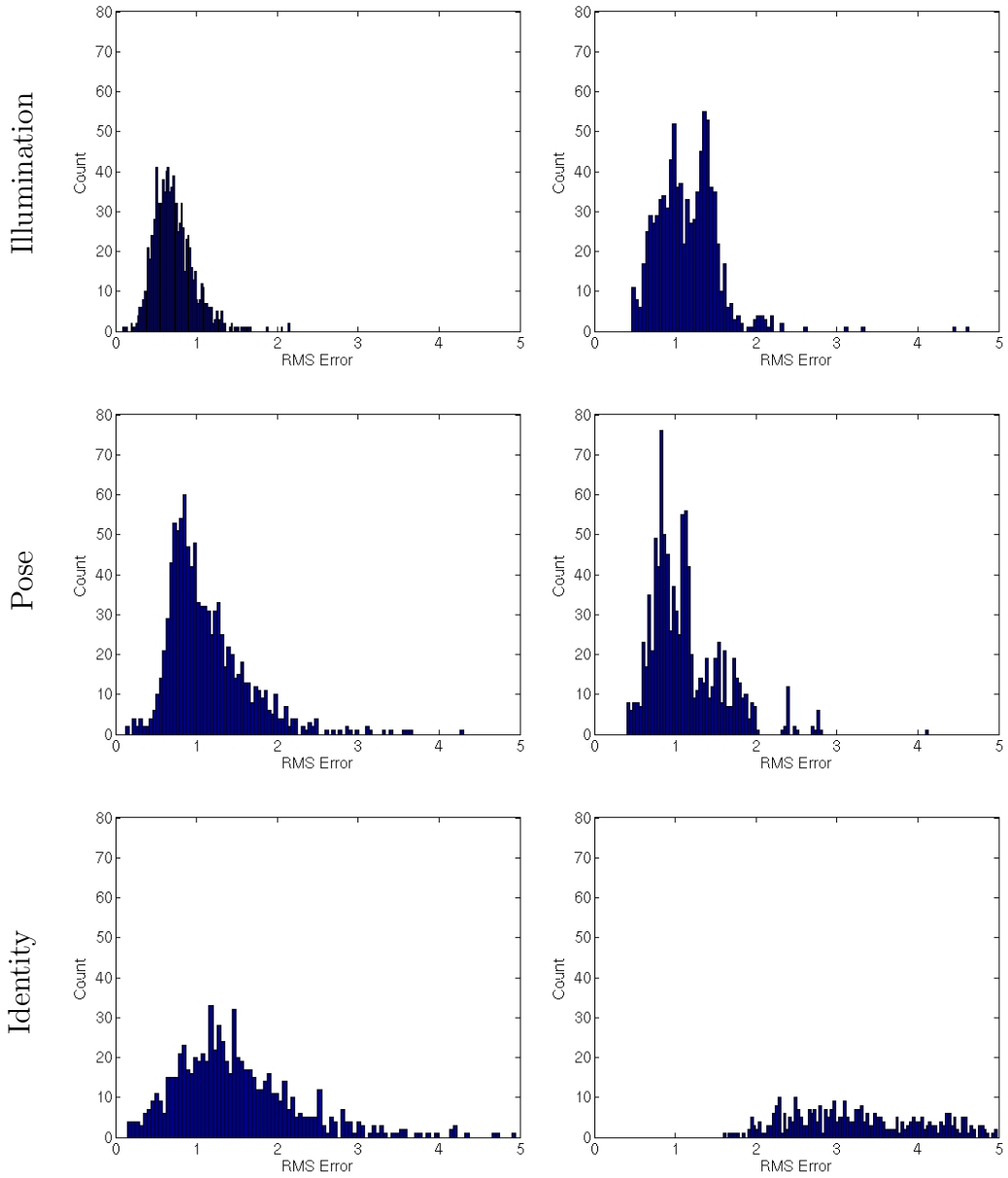
In [10] we generated trials for perturbations of varying magnitude and plot graphs of the frequency of convergence against the magnitude of the perturbation. In this paper we perturb the parameters by a large, fixed magnitude so that we can then vary the size of the model. See Section 5.3. Specifically, we perturb the shape parameters by 0.8 and the similarity parameters by 4.0. Before we provide the details of how we vary the size of the model, we first describe how we choose the convergence criterion and the model dimensionalities.

5.1 Determining The Convergence Criterion

There are no obvious or established choices for the selection of the convergence criterion. In previous work we simply chose a value and verified that it corresponds to "good" convergence. Generally we chose a threshold of between 1.0 and 2.0 pixels on the RMS error between the final mesh vertex locations and the ground-truth mesh vertex locations.

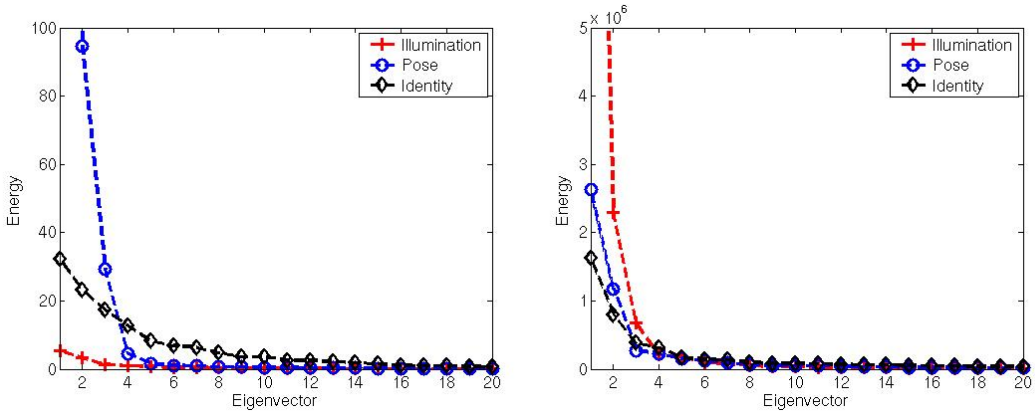
To choose the threshold in a more principled manner, we conducted the following experiments. We constructed three small AAMs for the three datasets (Illumination, Pose, and Identity.) In particular, we constructed AAMs from just 10 training images. We then ran 1000 perturbation experiments (in total) fitting the AAMs to the 10 training images. In Figure 4(a) we plot histograms of the final RMS error between the final mesh vertex locations and the ground-truth mesh vertex locations. We also repeated this experiment but with three larger models constructed from 100 images from each of the three datasets. We also ran 1000 perturbation experiments (in total) fitting the larger AAMs to the 100 training images. The corresponding histograms are shown in Figure 4(b).

The results in Figures 4(a) all show that most of the fitting trials for all three datasets end up with a RMS error between 0.0 and 3.0 pixels. Since the AAM used in this case is small and the initial perturbation not too large, it is reasonable to assume that most of these trials converged. We verified this by visually inspecting a number of results with final RMS errors in the 0.0 to 3.0 pixel range and confirmed that they look as though they have converged. The results in Figures 4(b) for the large AAMs are similar for the Illumination and Pose datasets, but not for the Identity dataset. For the Illumination and Pose AAMs, this is still a relatively easy fitting experiment, and most of the trials converge. For the Identity AAM, the fitting task is far harder. Most of the trials results in a final RMS value of over 3.0 pixels. We visually inspected a number of these results and found that when the final RMS error is somewhat larger than 3.0 pixels, the trials look as though they have diverged. Based on the results in Figure 4 we chose the convergence criterion to be 3.0 pixels. Naturally, this is still somewhat of an arbitrary decisions. But, note that whether we choose 2.0, 3.0, 4.0, or 5.0 pixels, the conclusion in Figure 4(b) is always the same. Most of the Illumination and Pose dataset trials converge, whereas very few of the Identity trials converge.



(a) 10 Images (b) 100 Images

Figure 4: Histograms of the final RMS fitting error between the final mesh vertex locations and the ground-truth mesh vertex locations for (a) small AAMs constructed with 10 training images and (b) larger AAMs constructed with 100 training images. We visual inspected many of the results and concluded, that for the small AAMs in (a), most of the trials for all three datasets converge, and that for the larger AAMs in (b), most of the trials for the Illumination and Pose datasets converge, but most of the trials for the Identity dataset diverge. Based on these results we chose the converge criterion to be 3.0 pixels. Note, however, than the main conclusion does not depend very much on the exact choice of this threshold.



(a) Shape (b) Appearance

Figure 5: Energy distribution of AAM shape (a) and appearance (b) eigenvectors.

5.2 Choosing the Model Dimensionalities

Next we address how to choose the dimensionality of the models. In most papers on AAMs, the dimensionality of the shape and appearance models is chosen by retaining a fixed percentage (typically 95%) of the variance in the eigenvalues. In our three datasets, the relative proportion of shape and appearance variation is very different. The Illumination dataset contains a lot of appearance variation, but not much shape variation. The Pose dataset is the other way around. See Figure 5 for plots of the PCA eigenvalues of the shape and appearance components of AAMs constructed from 100 images from each of the three datasets.

How, then, do we choose the dimensionalities of the models? We chose to base this decision on fitting performance, rather than on measures of the eigenvalues themselves. We chose the models to maximize the fitting robustness. In Figure 6 we include scatter plots of the percentage of trials converged against the fraction of shape variance retained in the model. Since the decision is 2D (we need to choose the dimensionality of both the shape and appearance models at the same time), when varying the size of the shape model we include results for a few different sizes of appearance models, and vice versa. In Figure 6(a) we include results for the shape component of the three datasets and in Figure 6(b) we include results for the appearance component.

First consider the shape component of the illumination model. Really there should be little shape variation. This manifests itself with the performance monotonically decreasing with the fraction of the shape variance retained. The more variance retained, the larger the shape model, but the more noise retained. Based on these results we chose the size of the shape component of the Illumination model to be 65%. On the other hand, the appearance component of the Illumination model should be large. There is a large amount of appearance variation in the dataset. This manifests itself with the performance generally increasing with the amount of appearance variation retained. At some point this levels off when all we add is noise. We therefore chose the size of the appearance model for the Illumination dataset to be 95%. Using similar arguments, we chose the size of the Pose AAM to be 98% for shape

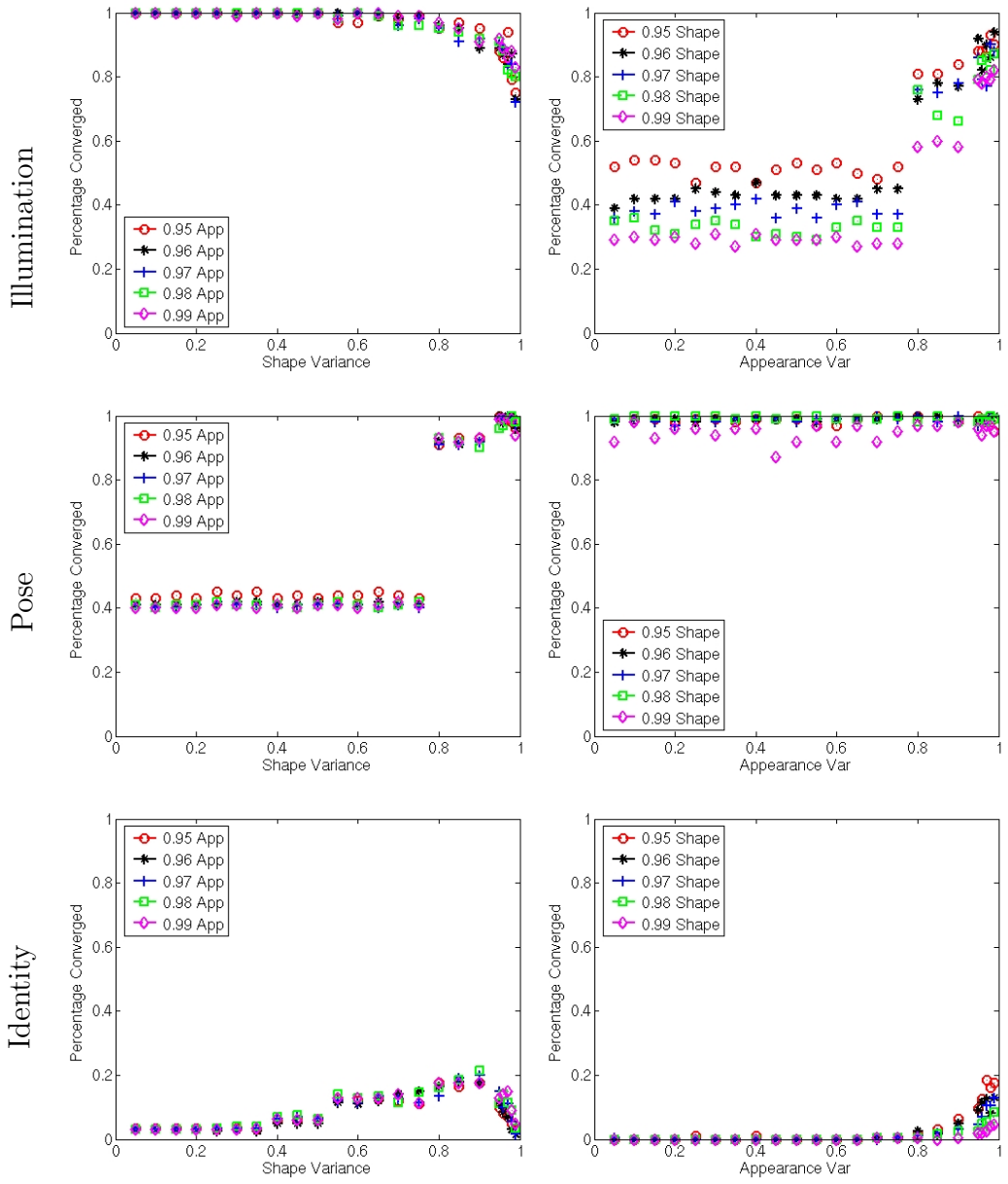


Figure 6: Scatter plots of the percentages of trials converged against the fraction of variance retained in the model for the three datasets for (a) the shape model and (b) the appearance model. From these plots we chose the size of the Illumination AAM to be 65% for shape and 95% for appearance, the size of the Pose AAM to be 98% for shape and 98% for appearance, and finally the size of the Identity AAM to be 90% for shape and 97% for appearance.

and 98% for appearance, and the size of the Identity AAM to be 90% for shape and 97% for appearance.

5.3 Experiment Description

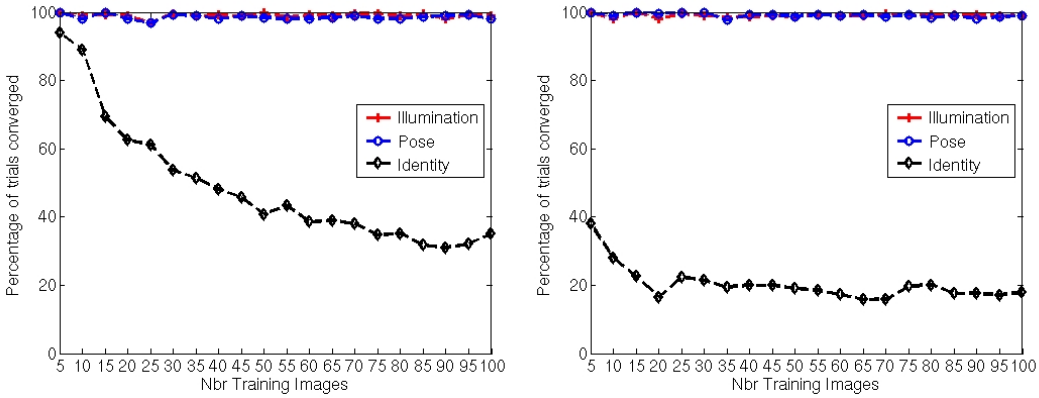
To compare the three datasets, we wish to generate fitting results for varying sizes of models. We also wish to separate the effects of shape and appearance. We do this by evaluating the fitting algorithm using shape models of varying size, but with a constant appearance model computed over the complete dataset. The shape models are computed by randomly choosing a fixed number (n) of training shapes and varying n between 5 and 100. For each n we build the shape component of the AAM using those n images (and retaining the same fraction of variance as described in Section 5.2 above). We then combine this shape component with the constant appearance model and evaluate the fitting performance of the resulting AAM on the n training images. The influence of the appearance model is determined in a similar fashion by combining varying size appearance models computed from n images with a constant shape model computed from the entire dataset, and then evaluating the fitting performance on the n training images. The main benefits of performing the experiments in this manner are:

- We evaluate the effect of varying shape and appearance model size separately. When varying shape, the appearance model is fixed, and vice versa.
- By fitting to the n training images, we know that the AAM we are using can model the data it is being fit to. If it diverges, it is because it fell into a local minima, not because it cannot explain the data.

5.4 Experiment Results

Figure 7 shows the results of these experiments. In Figure 7(a) we plot the frequency of convergence against the size of the shape model $n = 5, 10, \ldots, 100$ for a fixed appearance model and in Figure 7(b) we plot the frequency of convergence against the size of the appearance model $n = 5, 10, \ldots, 100$ for a fixed shape model. It is immediately apparent that fitting the Generic AAM is significantly harder than fitting either of the Person Specific AAMs. Note that the relative performance is entirely due to the data. Fitting the Generic AAM is inherently far harder than fitting the Person Specific AAM.

At this point, it is natural to ask what is the cause of this drastically different performance. Note the following results: (1) A Generic AAM built with 5 shape training images and 100 appearance training images operates about as well as similar Person Specific AAMs. See leftmost point in Figure 7(a). (2) A Generic AAM built with 100 shape training examples and 100 appearance training examples operates far, far worse than similar Person Specific AAMs. See leftmost point in Figure 7(b). (3) The performance of the Generic AAM drops rapidly with increasing size of the shape model. See Figure 7(a). But, the performance of all three datasets is much more independent of the size of the appearance model. See Figure 7(b).



(a) Shape (b) Appearance

Figure 7: Experimental results showing the frequency of convergence against (a) the size of the shape model $n = 5, 10, \ldots, 100$ for a fixed appearance model and (b) the size of the appearance model $n = 5, 10, \ldots, 100$ for a fixed shape model. These results (which use the Project-Out algorithm [10]) clearly show that fitting a Generic AAM is inherently harder than fitting the Person Specific AAM. They also show that the main cause of difficulty is the size of the shape model. See text for more explanation.

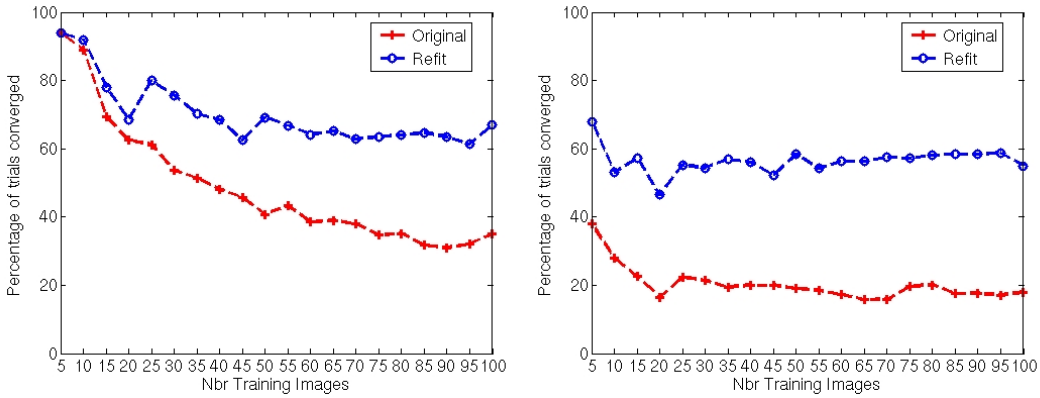
These results indicate that it is the shape component of the Generic AAM that is causing the problem. As further evidence of this argument, consider Figure 5 in which we plot the magnitudes of the eigenvectors for the shape and appearance models each computed with all 100 training images. The appearance eigenvectors for the 3 databases are all very similar. If anything, there is less appearance variation in the Generic AAM than the Person Specific AAMs. On the other hand, the shape eigenvectors of the Generic AAM are substantially larger than those of the Person Specific AAMs (at least after the first 4). The "effective" dimensionality of the shape component of the Generic AAM is far higher than the dimensionality of the Person Specific shape models.

5.5 Experiment Conclusions

Our experimental results confirm that fitting a Generic AAM is far harder than fitting a Person Specific AAM. The main reason for the extra difficulty appears to be that the effective dimensionality of the generic shape model is far higher than that of the Person Specific shape models. On the other hand, the performance is relatively independent of the size of the appearance model.

6 Improvements to Generic AAMs

Perhaps the main reason for performing the evaluation in Section 5 was to suggest possible methods of improving the performance of Generic AAMs. In the remainder of this section



(a) Shape (b) Appearance

Figure 8: Average rate of convergence for Generic AAMs computed using original and refitted labels. Results are shown for (a) AAMs with varying shape model sizes and fixed appearance model and (b) AAMs with varying appearance model sizes and fixed shape model.

we describe two such techniques: (1) refitting (Section 6.1) and (2) simultaneous fitting of shape and appearance (Section 6.2). These are by no means the only possibilities. We plan to cover several other possibilities in future papers.

6.1 Data Refitting

The vertex locations of the shape mesh used in the experiments above were all marked by hand. Due to the large amount of data involved (in total 16,800 mesh vertices were used) and the difficulty of the task, the quality of the labels is less than perfect. We used a "data refitting" algorithm to improve the ground truth data. Although this procedure does not improve consistently misplaced labels, it significantly improves model fitting performance.

First we construct a AAM with the original hand-marked labels. We then refit the AAM to the original training images and use the vertex locations of the fitted shape mesh as new landmark data. Cases where the fitting process diverges are detected automatically based on the fitting error.

Note that this procedure is very closely related to our automatic AAM construction algorithm [3]. In that algorithm, an AAM is automatically constructed by iterating model construction and model fitting; i.e. it operates: (1) build AAM, (2) refit AAM to training data, (3) rebuild AAM, (4) refit AAM to training data, etc. Although there are a few minor differences (such as the refitting algorithm assuming that the initial labels contain the ideal shape model plus noise whereas the automatic construction algorithm can extend the shape model), the refitting algorithm described above can be regarded as performing one iteration of the automatic construction algorithm, starting from the hand-marked labels. This fact provides a theoretical basis for the refitting algorithm.

Once we have refit the AAM to the training data and re-built the AAM, we need to decide how many shape and appearance dimensions to keep. We re-ran the experiments in

Simultaneous Inverse Compositional Algorithm

Pre-Computation:

(P1) Evaluate the gradients of ∇A_0 and ∇A_i for $i = 1, \ldots, m$

(P2) Evaluate the Jacobian of the warp $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $(\mathbf{x}; \mathbf{0})$

Iterate:

(I1) Warp I with $\mathbf{W}(\mathbf{x}; \mathbf{p})$ to compute $I(\mathbf{W}(\mathbf{x}; \mathbf{p}))$

(I2) Compute the error image $E_{\text{app}}(\mathbf{x})$ (Eqn. (16))

(I3) Compute the steepest descent images $\mathbf{SD}_{\text{sim}}(\mathbf{x})$ (Eqn. (12))

(I4) Compute the Hessian H_{sim} using Equation (15) and invert it

(I5) Compute $\sum_{\mathbf{x}} \mathbf{SD}^{\text{T}}_{\text{sim}}(\mathbf{x}) E_{\text{app}}(\mathbf{x})$

(I6) Compute $\Delta \mathbf{q} = -H^{-1}_{\text{sim}} \sum_{\mathbf{x}} \mathbf{SD}^{\text{T}}_{\text{sim}}(\mathbf{x}) E_{\text{app}}(\mathbf{x})$

(I7) Update $\mathbf{W}(\mathbf{x}; \mathbf{p}) \leftarrow \mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta \mathbf{p})^{-1}$ and $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \Delta \boldsymbol{\lambda}$

Figure 9: The Simultaneous Inverse Compositional Algorithm. Because the steepest descent images depend on the appearance parameters, Steps (I3-I4) must be performed in every iteration unlike the equivalent steps in the "Project Out" algorithm.

Section 5.2. The results (omitted) show that the performance peaks between 98% and 99% for both shape and appearance and for all three datasets. We therefore chose to retain either 98% or 99% in all cases. The fact that the optimal fitting performance occurs when we retain just less that 100% of the variance confirms that the refitting proceedure keeps most of the signal, and just removes (some of) the noise in the hand-labeled data.

The refitting procedure is evaluated by comparing the average rate of convergence of the Project-Out algorithm using Generic AAMs constructed from the original labels with the results obtained by fitting models constructed from the refitted labels. We follow the same evaluation methodology as used in Section 5. As shown in Figure 8 AAM fitting performance for the refitted labels is substantially better than fitting performance for the original labels. For AAMs with varying shape model sizes fitting performance improves on average from 48.7% of trials converged to 70.1% of trials converged (see Figure 8(a)). Similarly the fitting performance for AAMs with varying appearance model sizes improves on average from 20.2% of trials converged to 56.4% (see Figure 8(b)).

6.2 Simultaneous Fitting of Shape and Appearance

As described in Section 2.2, the goal of AAM fitting is usually formulated as minimizing the sum of squares difference between the model instance and the input image warped back onto the base mesh (see Equation (3)). Recently, we introduced the Simultaneous Inverse

Compositional Algorithm [1] which minimizes Equation (3) by performing a Gauss-Newton gradient descent optimization simultaneously on the warp parameters $\mathbf{p}$ and the appearance parameters $\boldsymbol{\lambda}$. The algorithm operates by iteratively minimizing:

$$\sum_{\mathbf{x}} \left[A_0(\mathbf{W}(\mathbf{x};\Delta\mathbf{p})) + \sum_{i=1}^{m} (\lambda_i + \Delta\lambda_i) A_i(\mathbf{W}(\mathbf{x};\Delta\mathbf{p})) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) \right]^2 \tag{10}$$

simultaneously with respect to $\Delta\mathbf{p}$ and $\Delta\boldsymbol{\lambda} = (\Delta\lambda_1, \ldots, \Delta\lambda_m)^{\mathrm{T}}$, and then updating the warp $\mathbf{W}(\mathbf{x};\mathbf{p}) \leftarrow \mathbf{W}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}(\mathbf{x};\Delta\mathbf{p})^{-1}$ and the appearance parameters $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \Delta\boldsymbol{\lambda}$.

To simplify the notation, denote:

$$\mathbf{q} = \begin{pmatrix} \mathbf{p} \\ \boldsymbol{\lambda} \end{pmatrix} \quad \text{and similarly} \quad \Delta\mathbf{q} = \begin{pmatrix} \Delta\mathbf{p} \\ \Delta\boldsymbol{\lambda} \end{pmatrix}; \tag{11}$$

i.e. $\mathbf{q}$ is an $n+m$ dimensional column vector containing the warp parameters $\mathbf{p}$ concatenated with the appearance parameters $\boldsymbol{\lambda}$. Denote the $n+m$ dimensional steepest-descent images as follows:

$$\mathbf{SD}_{\mathrm{sim}}(\mathbf{x}) = \left(\boldsymbol{\nabla} A \frac{\partial \mathbf{W}}{\partial p_1}, \ldots, \boldsymbol{\nabla} A \frac{\partial \mathbf{W}}{\partial p_n}, A_1(\mathbf{x}), \ldots, A_m(\mathbf{x}) \right) \tag{12}$$

where $\boldsymbol{\nabla}\mathbf{A}$ is defined as

$$\boldsymbol{\nabla}\mathbf{A} = \boldsymbol{\nabla} A_0 + \sum_{i=1}^{m} \lambda_i \boldsymbol{\nabla} A_i. \tag{13}$$

We can then compute the parameter update $\Delta\mathbf{q}$ as

$$\Delta\mathbf{q} = -H_{\mathrm{sim}}^{-1} \sum_{\mathbf{x}} \mathbf{SD}_{\mathrm{sim}}^{\mathrm{T}}(\mathbf{x}) E_{\mathrm{app}}(\mathbf{x}) \tag{14}$$

where:

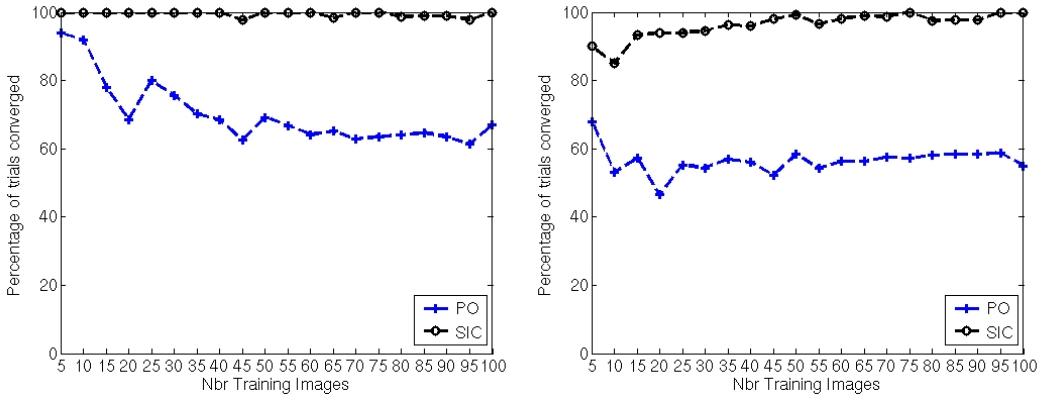
$$H_{\mathrm{sim}}^{-1} = \sum_{\mathbf{x}} \mathbf{SD}_{\mathrm{sim}}^{\mathrm{T}}(\mathbf{x})\, \mathbf{SD}_{\mathrm{sim}}(\mathbf{x}) \tag{15}$$

and E_{app} is defined as

$$E_{\mathrm{app}}(\mathbf{x}) = I(\mathbf{W}(\mathbf{x};\mathbf{p})) - \left[A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) \right] \tag{16}$$

Since the steepest descent images $\mathbf{SD}_{\mathrm{sim}}$ depend on the appearance parameters $\boldsymbol{\lambda}$ through Equation (13) they have to be re-computed in every iteration. The Simultaneous algorithm is therefore fairly (but not exceedingly) inefficient. Our implementation runs at about 1 frame per second in Matlab. The Simultaneous algorithm is summarized in Figure 9.

The Simultaneous Inverse Compositional Algorithm is defined for *independent AAMs* [10], which separately parameterize shape and appearance. It is different from the original AAM fitting algorithm defined for *combined AAMs* which jointly parameterize shape and appearance [4]. In the fitting algorithm in [4], the equivalent of the steepest descent images are assumed to be constant. On the other hand, the steepest descent images in the Simultaneous Inverse Compositional Algorithm are updated in each iteration of the algorithm. As the estimate of the appearance is updated, the way that the shape parameters are solved for



(a) Shape (b) Appearance

Figure 10: Average rate of convergence for Generic AAMs using the Simultaneous Inverse Compositional (SIC) and Project-Out (PO) algorithms using the refit labels used in Figure 8. Results are shown for (a) AAMs with varying shape model sizes and fixed appearance model and (b) AAMs with varying appearance model sizes and fixed shape model.

changes. The main reason for the performance improvement of the Simultaneous algorithm over the Project-Out algorithm is the fact that the steepest descent images are updated, not the coupling of the shape and appearance. In [10] we presented experimental results with a "coupled" version of the Project-Out algorithm and showed there is no performance increase over the separate optimization over the two sets of parameters.

We compared the performance of the Simultaneous Inverse Compositional and Project-Out algorithms by empirically evaluating the average rate of convergence for Generic AAMs for varying shape and appearance model sizes. Here we use AAMs constructed using refitted labels as described in Section 6.1. As shown in Figure 10, the Simultaneous Inverse Compositional Algorithm performs significantly better than the Project-Out algorithm. For AAMs with varying shape model sizes the fitting performance improves on average from 70.1% of trials converged to 99.5% (Figure 10(a)). For AAMs with varying appearance model sizes the fitting performance improves on average from 56.4% to 96.3% (Figure 10(b)).

7 Evaluating Model Construction and Fitting Combined

In order to separate the effects of model building and model construction we reported results of fitting experiments on *seen* data in Section 5; i.e. data in the training set. In order to evaluate the combined effects of model building and construction in this section we show results of fitting experiments on *unseen* data; i.e. data not in the training set.

7.1 Experiment Description

As in Section 5.3 we wish to generate fitting results for varying sizes of models, separating the effects of shape and appearance. Here however, we exclude a fixed set of 10 images from model training in order to evaluate fitting performance on unseen data. As in Section 5.3 we build a constant appearance model using 90 training images and combine it with a shape model computed by randomly choosing a fixed number (n) of training shapes and varying n between 5 and 90 in order to evaluate the influence of the shape model. The influence of the appearance model is determined in a similar fashion by combining appearance models computed from n training images with a constant shape model computed over all 90 training images. In either case fitting performance is evaluated on the 10 test images not in the training set.

7.2 Experiment Results

Figure 11 shows the results of the fitting experiments. In Figure 11(a) we plot the frequency of convergence against the size of the shape model $n = 5, \ldots, 90$ for a fixed appearance model and in Figure 11(b) we plot the frequency of convergence against the size of the appearance model $n = 5, \ldots, 90$ for a fixed shape model. We show results for all three datasets using original labels and the Project-Out algorithm (Figure 11, first row), using refitted labels and the Project-Out algorithm (Figure 11, second row), and using refitted labels and the Simultaneous Inverse Compositional Algorithm (Figure 11, third row). Overall fitting performance for Person Specific AAMs is excellent across all conditions. Small performance improvements can be seen in going from 5 to 10 training images for pose data (for increasing shape model size) and for illumination data (for increasing appearance models sizes). After 10 training images, the performance stays roughly constant as we add more images.

The largest difference however is in the performance using the Generic AAM. While only a very small percentage of trials converge for original labels and the Project-Out algorithm, the majority of trials converge for refitted labels and the Simultaneous algorithm. In the experiments using refitted labels the performance of the Generic AAMs improves significantly with increasing numbers of training images, for both the Simultaneous and Project-out algorithms.

As Sections 4 and 5 have shown, there is an inherent trade-off between model construction and fitting: the larger the model, the more likely it is to model previously unseen data (see Figure 3), but also the more likely it is for the model to get stuck in local minima during fitting (see Figure 7). This effect is visible to a small extent in the fitting performance of the Project-Out algorithm using refitted labels on the illumination dataset (see Figure 11, second row) and the performance of the Project-Out algorithm using the original labels on the identity dataset (see Figure 11, top row). On the other hand, with the refit labels and the Simultaneous algorithm, the performance of the Generic AAM consistently improves with increasing numbers of training images. This indicates that for Generic AAMs, the difficulty in modeling currently outweighs the difficulty in fitting.

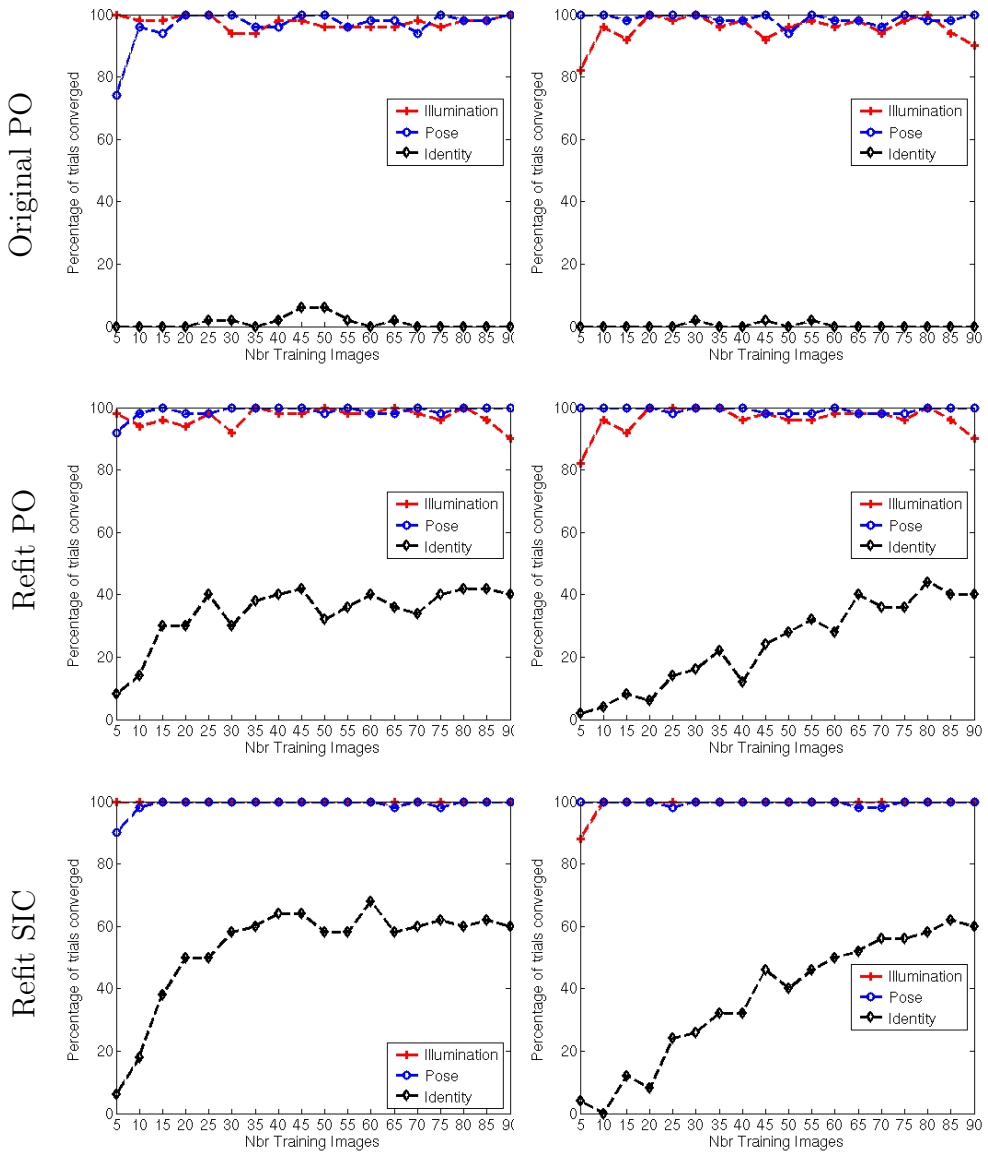


Figure 11: Average rate of convergence for both Person Specific and Generic AAMs on unseen data. Shown are results for different labels (original and refitted) and different fitting algorithms (Project-Out and Simultaneous).

7.3 Experiment Conclusions

The results in Figure 11 confirm the results of Section 6: fitting performance of Generic AAMs increases significantly if models are built using refitted labels and if the simultaneous fitting algorithm is used instead of the project-out algorithm. The fact that the performance of Generic AAMs generally increases with the amount of training data (and hence the size of the model) indicates that the difficulty in modeling currently outweighs the difficulty in fitting.

8 Discussion

In this paper we empirically compared Generic and Person Specific Active Appearance Models (AAMs). In Section 4 we showed that building a generic shape model is comparatively easy, whereas building a generic appearance model requires far more training data. We then demonstrated in Section 5 that fitting a Generic AAM appears to be harder than fitting a Person Specific AAM mainly due to the higher effective dimensionality of the shape model. In Section 6 we discussed two refinements to Generic AAMs: (1) label refitting and (2) the Simultaneous Inverse Compositional Algorithm. Finally, in Section 7 we evaluated the trade-off between the difficulties of model construction and fitting. For the Project-Out algorithm, there is an intermediate model size beyond which the reduction in fitting performance outweighs the gains from the improved representational power of the model. For the Simultaneous algorithm, however, the larger the model the better the overall performance (for the datasets we tried); i.e. the gains from the improved representational power outweigh the losses from the worse fitting performance. These results indicate that for Generic AAMs on unseen data, the difficulty in modeling (appearance) currently outweighs the difficulty in fitting (shape). The performance improvement due to the two refinements described in Section 6 on both seen and unseen data is summarized in Figure 12.

In Section 5 we showed that fitting an AAM with a complicated shape model is difficult. One possible refinement to help solve this problem is to incorporate shape priors into the fitting algorithm as suggested in [2, 5].

While the Simultaneous Inverse Compositional Algorithm performs significantly better than the Project-Out algorithm (see Figure 10), it is also fairly slow [1]. Our implementation of the Simultaneous algorithm runs at about 1 frame per second in Matlab. Another possible area for future work is to combine the two algorithms. For example, the Simultaneous algorithm might be used on the first image of a sequence for a high quality fit. The mean appearance image of the AAM is then updated with the extracted face appearance and the remainder of the sequenced tracked with the Project-Out algorithm. This "mean-update" algorithm is a special case of the "Template Update" algorithm for AAMs proposed in [10] and may partially alleviate the difficulty in building an appearance model general enough to model all faces.

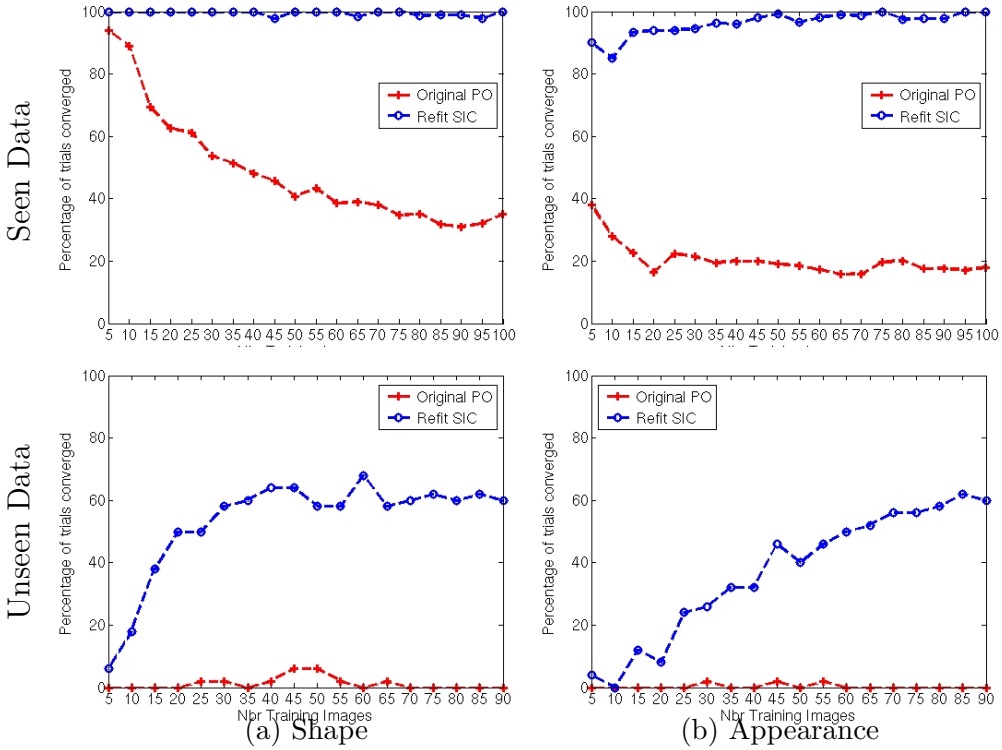


Figure 12: Average rate of convergence for Generic AAMs using the Simultaneous Inverse Compositional (SIC) Algorithm on refitted labels and the conventional Project-Out (PO) Algorithm on the original hand-marked labels. Top Row: The seen data of Section 5. Bottom Row: The unseen data of Section 7. These graphs illustrate the dramatic fitting performance improvement obtained by combining the two techniques described in Section 6.

9 Acknowledgments

The research described in this paper was supported in part by ONR contract N00014-00-1-0915, U.S. DoD contract N41756-03-C4024, National Institute of Mental Health grant R01 MH51435, and Denso Corporation.

References

[1] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 3. Technical Report CMU-RI-TR-03-35, Carnegie Mellon University Robotics Institute, 2003.

[2] S. Baker, R. Gross, and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 4. Technical Report CMU-RI-TR-04-14, Carnegie Mellon University Robotics Institute, 2004.

[3] Simon Baker, Iain Matthews, and Jeff Schneider. Automatic construction of active appearance models as an image coding problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1380–1384, October 2004.

[4] 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, June 2001.

[5] T.F. Cootes and C.J. Taylor. Constrained active appearance models. In *Proceedings of the International Conference on Computer Vision*, pages 748–754, 2001.

[6] I.L. Dryden and K.V. Mardia. *Statistical Shape Analysis*. Wiley & Sons, 1998.

[7] G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[8] M. Kirby and L. Sirovich. Application of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE PAMI*, 12(1):103–108, 1990.

[9] K.-C. Lee, L. Ho, and D. Kriegman. Nine points of light: Acquiring subspaces for face recognition under variable lighting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 519–526, 2001.

[10] I. Matthews and S. Baker. Active Appearance Models revisited. *International Journal of Computer Vision*, 60(2):135–164, 2004.

[11] P. J. Phillips, H. Wechsler, J. Huang, and P. Rauss. The FERET database and evaluation procedure for face recognition algorithms. *Image and Vision Computing*, 16(5):295–306, 1998.

[12] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2D+3D active appearance models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 668–675, 2004.