Optimization problems for fast AAM fitting in-the-wild

Georgios Tzimiropoulos
1. School of Computer Science
University of Lincoln, U.K.
2. Department of Computing
Imperial College London, U.K.
gtzimiropoulos@lincoln.ac.uk

Maja Pantic
1. Department of Computing
Imperial College London, U.K.
2. University of Twente
The Netherlands
m.pantic@imperial.ac.uk

Abstract

We describe a very simple framework for deriving the most-well known optimization problems in Active Appearance Models (AAMs), and most importantly for providing efficient solutions. Our formulation results in two optimization problems for fast and exact AAM fitting, and one new algorithm which has the important advantage of being applicable to 3D. We show that the dominant cost for both forward and inverse algorithms is a few times mN which is the cost of projecting an image onto the appearance subspace. This makes both algorithms not only computationally realizable but also very attractive speed-wise for most current systems. Because exact AAM fitting is no longer computationally prohibitive, we trained AAMs in-the-wild with the goal of investigating whether AAMs benefit from such a training process. Our results show that although we did not use sophisticated shape priors, robust features or robust norms for improving performance, AAMs perform notably well and in some cases comparably with current state-of-the-art methods. We provide Matlab source code for training, fitting and reproducing the results presented in this paper at `http://ibug.doc.ic.ac.uk/resources`.

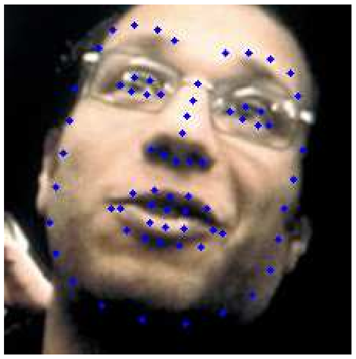


Figure 1. An example of a face in-the-wild taken from the LFPW database [3]. Landmarks were detected by fitting an AAM. The appearance model of the AAM was built using raw un-normalized pixel intensities as features. Neither sophisticated shape priors or robust norms were used during fitting nor robust image features were employed to build the AAM. Even without such sophisticated enhancements, AAM fitting produced satisfactory accuracy in landmark localization. To obtain these results, we simply trained the AAM in-the-wild (on the same database) and additionally for fitting and we used Fast-Forward algorithm, an exact but fast simultaneous algorithm.

1. Introduction

Active Appearance Models (AAMs) have been around in computer vision research for more than 15 years [5]. They are statistical models of shape and appearance that can generate instances of a specific object class (e.g. faces) given a small number of model parameters which control shape and appearance variation. Fitting an AAM to a new image entails estimating the model parameters so that the model instance and the given image are *"close enough"* typically in a least-squares sense. Recovering the shape parameters is important because it implies that the location of a set of landmarks (or fiducial points) has been detected in the new image. Landmark localization is of fundamental significance in many computer vision problems like face and medical image analysis. Hence, fitting AAMs robustly to new images has been the focus of extensive research over the past years.

AAM fitting is an iterative process at each iteration of which an update of the current model parameters is estimated. Typically, the update is a function of the error between the image and the model measured in the canonical reference frame of the model. There are two main lines of research for modeling this function. The first is to learn it via regression which was also the approach proposed in the original AAM paper [5]. Regression-based approaches are fast but approximate. For example in [5], the relationship between the error image and the update is assumed linear and independent of the current model parameters. A notable departure from [5] is the work of [21] in which a nonlinear regressor is learned via boosting. Other discriminative methods for fitting AAMs have been proposed in [11, 22, 20].

The second line of research for fitting AAMs is through non-linear least-squares [16]. AAM fitting is formulated as a Lukas-Kanade (LK) problem which can be solved iteratively using Gauss-Newton optimization. However, standard gradient descend algorithms when applied to AAMs are inefficient. This problem is addressed in the seminal work of Matthews and Baker [16] which extends the classical Lukas-Kanade algorithm [13] and the appearance-based tracking framework of Hager and Belhumeur [9] for the case of AAMs and deformable models. One of the major contributions of [16] is the so-called project-out inverse compositional algorithm (POIC). The algorithm is coined project-out because it decouples shape from appearance by projecting out appearance variation and inverse compositional because the warp update is estimated in the model coordinate frame and then composed to the current warp estimate (this is in contrast to the standard LK algorithm in which the warp parameters are updated in a forward additive fashion). This combination results in an algorithm which is as efficient as regression-based approaches and is now considered the standard choice for fitting person-specific AAMs (i.e. AAMs trained for fitting face images of a specific subject which is known in advanced). Its main disadvantage is its limited capability of generalizing well to unseen variations, the most notable example of which is the case of generic AAMs (i.e. AAMs trained for fitting face images of various subjects not known in advance).

In contrast to POIC, the simultaneous inverse compositional (SIC) algorithm, proposed in [1], has been shown to perform robustly for the case of generic fitting [7]. However, the computational cost of the algorithm is almost prohibitive for most applications. Let n and m denote the number of the shape and appearance parameters of the AAM. Then, the dominant cost per iteration of SIC is on the order of $(n+m)^2N$, where N is the number of pixels in the reference frame. Note that the cost of POIC is only $O(nN)$. For generic fitting $m \gg n$ and hence the huge difference in computational cost has either ruled out SIC from most papers/studies that depart from the person-specific case or made the authors resort in approximate solutions (please see [20] for an example).

Some attempts to reduce the cost of SIC have been occasionally reported in AAM literature. A notable example is the normalization algorithm [1]. However, the performance of the normalization algorithm has been reported to be closer to that of POIC rather than that of SIC. Finally, other techniques for reducing the cost to some extend via pre-computations have been reported in [2, 17].

Main results. We show that the cost for solving the exact AAM non-linear least squares problem with no approximations for *both forward and inverse* is significantly less than $O((n+m)^2N)$. Let f be a function that is no necessarily convex. Then a standard result from optimization theory is [4]

$$\min_{x,y} f(x,y) = \min_{x}[\min_{y} f(x,y)]. \quad (1)$$

As we show later on, using (1) reduces the dominant cost for both forward and inverse algorithms to nmN. Especially for $m \gg n$, which is the case for generic face alignment, the cost is reduced to a few times mN which is the cost of projecting an image onto the appearance subspace. Hager and Belhumeur made use of the above result in [9], however without explicitly referring to (1). To the best of our knowledge, the only AAM paper that acknowledges the optimization strategy described in [9] is [18]. Here, we provide an alternative derivation based on (1) which has the advantage of producing the exact form of the optimization problem that SIC solves. Hence, our derivations shed further light on the different optimization problems that POIC and SIC solve. Additionally, the authors of [18] investigated only the inverse case. As it is well known, the inverse compositional approach cannot be applied to 3D AAMs [23]. One of our main contributions is to show that (1) can be used to derive a forward additive update scheme and hence can be readily applied to 3D. Finally, we believe that the proposed framework is readily applicable to recently published papers on AAMs [14, 15].

Our second main contribution is to train AAMs in-the-wild using the well-known LFPW database [3] and then fit using the proposed fast forward and inverse simultaneous algorithms, with the goal of investigating whether AAMs benefit from such a training process. Indeed, it turns out that this is the case: the obtained fittings are in many cases as good as the ones produced by current state-of-the art methods (please see Fig. 1 for a fitting example). These results are notable given that no shape prior was used, the employed appearance model was built using raw pixel intensities and no attempt to use more sophisticated image features (like Gabor filter responses as in [14] or SIFT features [12] as in [3]) was made.

2. AAMs

An AAM is defined by the shape, appearance and motion models. Learning the shape model requires consistently annotating a set of u landmarks $[x_1, y_1, \dots x_u, y_u]$ across D training images $\mathbf{I}_i(\mathbf{x}) \in \mathcal{R}^N$ (e.g. faces). These points are said to define the shape of each object. Next, Procrustes Analysis is applied to remove similarity transformations from the original shapes and obtain D similarity-free shapes. Finally, PCA is applied on these shapes to obtain a shape model defined by the mean shape and n shape eigenvectors $\{\mathbf{s}_0, \mathbf{S} \in \mathcal{R}^{\{2u,n\}}\}$. The model typically captures shape variation due to identity, pose and expression. Assume that we are given a new similarity-free shape $\mathbf{s}$. Then, the model can be used to represent $\mathbf{s}$ as

$$\hat{\mathbf{s}} = \mathbf{s}_0 + \mathbf{S}\mathbf{p}, \quad \mathbf{p} = \mathbf{S}^T(\mathbf{s} - \mathbf{s}_0). \tag{2}$$

Finally, in this work, to model similarity transforms the shape matrix $\mathbf{S}$ is appended with 4 similarity eigenvectors [16], all eigenvectors are re-orthonormalized, and then (2) is applied.

Learning the appearance model requires removing shape variation from the texture. This can be achieved by first warping each $\mathbf{I}_i$ to the reference frame defined by the mean shape $\mathbf{s}_0$ using motion model $\mathbf{W}$. Finally, PCA is applied on the shape-free textures, to obtain the appearance model defined by the mean appearance and m appearance eigenvectors $\{\mathbf{A}_0, \mathbf{A} \in \mathcal{R}^{\{N,m\}}\}$. The model captures appearance variation for example due to identity and illumination. The model can be used to represent a shape-free test texture $\mathbf{I}$ as

$$\hat{\mathbf{I}} = \mathbf{A}_0 + \mathbf{A}\mathbf{c}, \quad \mathbf{c} = \mathbf{A}^T(\mathbf{I} - \mathbf{A}_0). \tag{3}$$

We used piecewise affine warps $\mathbf{W}(\mathbf{x};\mathbf{p})$ as the motion model in this work. Briefly, to define a piecewise affine warp, one first needs to triangulate the set of vertices of the given shapes. Then, each triangle in $\mathbf{s}$ and the corresponding triangle in $\mathbf{s}_0$ can define an affine warp. The collection of all affine warps defines a piecewise affine warp which is parameterized with respect to $\mathbf{p}$.

Finally, a model instance is synthesized to represent a test object by warping $\hat{\mathbf{I}}$ from the mean shape $\mathbf{s}_0$ to $\hat{\mathbf{s}}$ using the piecewise affine warp define by $\mathbf{s}_0$ and $\hat{\mathbf{s}}$. Please see [16, 5] for a detailed coverage of AAMs.

3. Fitting AAMs

Our approach to fitting AAMs is based on non-linear least-squares [16]. Assume that we are given a test image $\mathbf{I}$. Fitting an AAM to the image entails estimating the model parameters so that the ℓ_2 norm of the error between the model instance and the given image is minimized

$$\arg\min_{\mathbf{p},\mathbf{c}} ||\mathbf{I}(\mathbf{W}(\mathbf{x};\mathbf{p})) - \mathbf{A}_0 - \mathbf{A}\mathbf{c}||^2. \tag{4}$$

Because (4) is a non-linear function of $\mathbf{p}$, the standard approach to proceed is to linearize with respect to the shape parameters $\mathbf{p}$ and then optimize iteratively in a Gauss-Newton fashion. Linearization of (4) with respect to $\mathbf{p}$ can be performed in two coordinate frames. In the *forward* case, the test image $\mathbf{I}$ is linearized around the current estimate $\mathbf{p}$, a solution for a $\Delta\mathbf{p}$ is sought using least-squares, and $\mathbf{p}$ is updated in an additive fashion $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$. In the *inverse* case, the model $\{\mathbf{A}_0, \mathbf{A}\}$ is linearized around $\mathbf{p} = \mathbf{0}$, a solution for a $\Delta\mathbf{p}$ is sought using least-squares, and $\mathbf{p}$ is updated in a compositional fashion $\mathbf{p} \leftarrow \mathbf{p} \circ \Delta\mathbf{p}^{-1}$, where $\circ$ denotes the composition of two warps. Note that applying the inverse compositional approach for piecewise affine warps is by no means straightforward. Please see [16] for a principled way of applying the inverse composition to AAMs.

Following the seminal work of [16], inverse algorithms have gained increased popularity. The two most popular inverse algorithms are SIC and POIC. At each iteration SIC linearizes with respect to both $\mathbf{c}$ and $\mathbf{p} = \mathbf{0}$, and hence (4) becomes

$$\arg\min_{\Delta\mathbf{p},\Delta\mathbf{c}} ||\mathbf{I} - \mathbf{A}_0 + \mathbf{J}_0\Delta\mathbf{p} - \sum_{i=1}^{m}(c_i + \Delta c_i)(\mathbf{A}_i + \mathbf{J}_i\Delta\mathbf{p})||^2, \tag{5}$$

where $\mathbf{J}_i$ is the $N \times n$ Jacobian built as follows: Its $k-$th row corresponds to pixel $\mathbf{x}_k$ and contains the $1 \times n$ vector $[\mathbf{A}_{i,x}(k) \;\; \mathbf{A}_{i,y}(k)]\frac{\partial \mathbf{W}(\mathbf{x}_k;\mathbf{p})}{\partial \mathbf{p}}$. $\mathbf{A}_{i,x}(k)$ and $\mathbf{A}_{i,y}(k)$ are the x and y gradients of $\mathbf{A}_i$ for the $k-$th pixel and $\frac{\partial \mathbf{W}(\mathbf{x}_k;\mathbf{p})}{\partial \mathbf{p}} \in \mathcal{R}^{2\times n}$ is the Jacobian of the piecewise affine warp. Please see [16] for calculating and implementing $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$. All of these are defined in the model coordinate frame for $\mathbf{p} = \mathbf{0}$ and can be pre-computed. Finally, with some abuse of notation we denote by $\mathbf{J}_i = [\mathbf{A}_{i,x} \;\; \mathbf{A}_{i,y}]\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ the process of constructing $\mathbf{J}_i$ for all N rows.

An update for $\Delta\mathbf{c}$ and $\Delta\mathbf{p}$ can be obtained only after second order terms are omitted as follows

$$\arg\min_{\Delta\mathbf{p},\Delta\mathbf{c}} ||\mathbf{I} - \mathbf{A}_0 - \mathbf{A}\mathbf{c} - \mathbf{A}\Delta\mathbf{c} - \mathbf{J}\Delta\mathbf{p})||^2, \tag{6}$$

where $\mathbf{J} = \mathbf{J}_0 + \sum_{i=1}^{m} c_i\mathbf{J}_i$. In [1], the update for SIC was derived as

$$[\Delta\mathbf{p}; \Delta\mathbf{c}] = \mathbf{H}_{sic}^{-1}\mathbf{J}_{sic}^T(\mathbf{I} - \mathbf{A}_0 - \mathbf{A}\mathbf{c}), \tag{7}$$

where $\mathbf{J}_{sic} = [\mathbf{A}; \mathbf{J}] \in \mathcal{R}^{N\times(m+n)}$ and $\mathbf{H}_{sic} = \mathbf{J}_{sic}^T\mathbf{J}_{sic}$ are the SIC Jacobian and Hessian respectively. SIC is slow because the cost for calculating $\mathbf{H}_{sic}$ is $O((n+m)^2N)$ [1].

POIC reduces this cost dramatically by decoupling shape and appearance by solving (6) in the subspace orthogonal to $\mathbf{A}$. Let us define the projection operator $\mathbf{P} = \mathbf{E} - \mathbf{A}\mathbf{A}^T$, where $\mathbf{E}$ is the identity matrix. Then, $||\mathbf{I} - \mathbf{A}_0 - \mathbf{A}\mathbf{c}||^2_{\mathbf{P}} = ||\mathbf{I} - \mathbf{A}_0||^2_{\mathbf{P}}$, and hence an update for $\Delta\mathbf{p}$ can be computed by optimizing

$$\arg\min_{\Delta\mathbf{p}} ||\mathbf{I} - \mathbf{A}_0 - \mathbf{J}_0\Delta\mathbf{p})||^2_{\mathbf{P}}, \tag{8}$$

the solution of which is given by

$$\Delta\mathbf{p} = \mathbf{H}_{poic}^{-1}\mathbf{J}_{poic}^T(\mathbf{I} - \mathbf{A}_0), \tag{9}$$

where the projected-out Jacobian $\mathbf{J}_{poic} = \mathbf{P}\mathbf{J}_0$ and Hessian $\mathbf{H}_{poic} = \mathbf{J}_{poic}^T\mathbf{J}_{poic}$ can be pre-computed. This reduces the cost to $O(nN)$, only [16].

4. Fast algorithms for fitting AAMs

Solving the exact problem in a simultaneous fashion as described above is not the only way for fitting AAMs. Below we describe two algorithms, Fast-SIC and Fast-Forward for achieving the same result by applying (1). The solution of the inverse problem was originally proposed in [18]. Here we provide an alternative derivation based on (1), which has the advantage of producing the exact form of the optimization problem that Fast-SIC solves. Hence, our derivations shed further light on the different optimization problems that POIC and SIC solve. Finally, to the best of our knowledge, Fast-Forward is described for the first time in AAM literature in this work.

4.1. Fast-SIC

Using (1), we can optimize (6) with respect $\Delta\mathbf{c}$, and then plug in the solution (which will be a function of $\Delta\mathbf{p}$) back to (6). Then, we can optimize (1), with respect to $\Delta\mathbf{p}$. Setting the derivative of (6) with respect to $\Delta\mathbf{c}$ equal to $\mathbf{0}$ gives the update of $\Delta\mathbf{c}$

$$\Delta\mathbf{c} = \mathbf{A}^T(\mathbf{I} - \mathbf{A}_0 - \mathbf{A}\mathbf{c} - \mathbf{J}\Delta\mathbf{p}) \tag{10}$$

Plugging the above into (6), we get the following optimization problem

$$\arg\min_{\Delta\mathbf{p}} ||\mathbf{I} - \mathbf{A}_0 - \mathbf{J}\Delta\mathbf{p})||^2_{\mathbf{P}}. \tag{11}$$

As we may see, the difference between POIC and Fast-SIC is that POIC uses $\mathbf{J}_0$ while Fast-SIC uses $\mathbf{J}$. This difference simply comes from the point at which we choose to linearize. Matthews and Baker [16] chose to project out first and then linearize. Fast-SIC first linearizes (the appearance model), and then projects out. Another way to interpret Fast-SIC is to solve the original SIC problem of (6) in the subspace defined by $\mathbf{P}$. This has the effect that the appearance terms $\mathbf{A}\mathbf{c}$ and $\mathbf{A}\Delta\mathbf{c}$ immediately vanish. However, the Jacobian $\mathbf{J}$ does not vanish as assumed by POIC. Hence, POIC is only an approximation to Fast-SIC (and hence to SIC).

The solution of Fast-SIC is readily given by

$$\Delta\mathbf{p} = \mathbf{H}^{-1}_{fsic}\mathbf{J}^T_{fsic}(\mathbf{I} - \mathbf{A}_0), \tag{12}$$

where the projected-out Jacobian and Hessian are given by $\mathbf{J}_{fsic} = \mathbf{P}\mathbf{J}$ and $\mathbf{H}_{fsic} = \mathbf{J}^T_{fsic}\mathbf{J}_{fsic}$, respectively. Because $\mathbf{J}$ is a function of $\mathbf{c}$, it needs to be re-computed per iteration.

Calculating $\mathbf{J}_{fsic}$ has dominant cost nmN. To see this we first note that for a matrix $\mathbf{X} \in \mathcal{R}^{N\times l}$, we can calculate $\mathbf{P}\mathbf{X} = \mathbf{X} - \mathbf{A}(\mathbf{A}^T\mathbf{X})$ with cost lmN. Let us also denote by $\mathbf{A}_x = [\mathbf{A}_{0,x} \dots \mathbf{A}_{m,x}] \in \mathcal{R}^{N\times(m+1)}$, the matrix the columns of which are the gradients of the model along the x-axis and $\mathbf{A}_y$ the matrix the columns of which are the gradients along the y-axis. Hence $\mathbf{J}_{fsic}$ can be computed in nmN from

$$\mathbf{J}_{fsic} = \mathbf{P}[\mathbf{A}_x\mathbf{c}' \;\; \mathbf{A}_y\mathbf{c}']\frac{\partial\mathbf{W}}{\partial\mathbf{p}}, \tag{13}$$

where $\mathbf{c}' = [1; \mathbf{c}] \in \mathcal{R}^{m+1}$. Note that $\frac{\partial\mathbf{W}}{\partial\mathbf{p}}$ above is evaluated at $\mathbf{p} = \mathbf{0}$ and can be pre-computed. Finally, the cost for calculating $\mathbf{H}_{fsic}$ is n^2N. Hence, the complexity of Fast-SIC per iteration is mainly due to the computation of $\mathbf{J}_{fsic}$.

4.2. Fast-Forward

The coordinate frame of the model is not the only frame that (4) can be solved. Rather than linearizing the model, we can linearize the test image in a standard Lukas-Kanade fashion [13]

$$\arg\min_{\{\Delta\mathbf{p},\mathbf{c}\}} ||\mathbf{I} + \mathbf{J}_I\Delta\mathbf{p} - \mathbf{A}_0 - \mathbf{A}\mathbf{c}||^2, \tag{14}$$

where $\mathbf{J}_I$ is the Jacobian of $\mathbf{I}$ and needs to be re-computed per iteration. Note that error above is already linear with respect to $\mathbf{c}$. Also there are no second terms that need to be omitted. Hence, we can directly optimize using (1). At each iteration, the optimal $\mathbf{c}$ is given by

$$\mathbf{c} = \mathbf{A}^T(\mathbf{I} + \mathbf{J}_I\Delta\mathbf{p} - \mathbf{A}_0). \tag{15}$$

Plugging the above into (14), we end up with the following optimization problem

$$\arg\min_{\Delta\mathbf{p}} ||\mathbf{I} + \mathbf{J}_I\Delta\mathbf{p} - \mathbf{A}_0||^2_{\mathbf{P}}, \tag{16}$$

the solution of which is readily given by

$$\Delta\mathbf{p} = -\mathbf{H}^{-1}_{ffw}\mathbf{J}^T_{ffw}(\mathbf{I} - \mathbf{A}_0), \tag{17}$$

where the projected-out Jacobian and Hessian are given by $\mathbf{J}_{ffw} = \mathbf{P}\mathbf{J}_I$ and $\mathbf{H}_{ffw} = \mathbf{J}^T_{ffw}\mathbf{J}_{ffw}$, respectively. Notice that in order to compute $\Delta\mathbf{p}$, the value of the optimal $\mathbf{c}$ is not required. Hence, the Fast-Forward iteratively applies (17), only.

Calculating $\mathbf{J}_{ffw}$ has dominant cost nmN. To readily see this notice that to calculate $\mathbf{J}_I$ the cost is nN, and hence the cost for calculating $\mathbf{P}\mathbf{J}_I$ is nmN. Alternatively, one could avoid calculating $\mathbf{P}\mathbf{J}_I$ directly because $\mathbf{J}^T_{fsic}(\mathbf{I} - \mathbf{A}_0) = \mathbf{J}^T_I\mathbf{P}(\mathbf{I} - \mathbf{A}_0)$, and $\mathbf{P}(\mathbf{I} - \mathbf{A}_0)$ has a cost of mN. However, the cost for calculating

$$\mathbf{A}^T\mathbf{J}_I = \mathbf{A}^T[\mathbf{I}_x \;\; \mathbf{I}_y]\frac{\partial\mathbf{W}}{\partial\mathbf{p}} \tag{18}$$

is nmN ($\mathbf{I}_x$ and $\mathbf{I}_y$ are the gradients of $\mathbf{I}$ evaluated at $\frac{\partial\mathbf{W}}{\partial\mathbf{p}}$) and calculating $\mathbf{A}^T\mathbf{J}_I$ is necessary if we wish to efficiently calculate $\mathbf{H}_{ffw}$ from

$$\mathbf{H}_{ffw} = \mathbf{J}^T_I\mathbf{J}_I - (\mathbf{A}^T\mathbf{J}_I)^T(\mathbf{A}^T\mathbf{J}_I). \tag{19}$$

Note that the cost for calculating $\mathbf{H}_{ffw}$ as above is n^2N and comes from the first term (this is because $\mathbf{A}^T\mathbf{J}_I \in \mathcal{R}^{m\times n}$). An additional cost for the forward additive formulation is that $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is evaluated at $\mathbf{p}$ and not at $\mathbf{p} = \mathbf{0}$, but the cost for doing this can be negligible.

An interesting observation following the above analysis is that, for both forward and inverse algorithms, the dominant computational cost comes from projecting out the appearance subspace when calculating the Hessian. If we choose not to do this, then the total cost is further reduced to $(n^2 + m)N$. Note however that, in this case, the resulting algorithms are approximate and not exact. We leave the evaluation of these approximate algorithms as interesting future work.

5. Fitting AAMs in-the-wild

Simultaneous AAM fitting algorithms are known to perform well but their performance has not been previously assessed on recently collected in-the-wild data sets. In this section and in the one that follows, we aim to address this gap in literature. In particular, we show that AAMs perform almost comparably to some state-of-the-art face alignment algorithms, even without using any priors (the fitting algorithms described above are used as is) and using raw pixel intensities as features.

One reason for not evaluating AAMs in-the-wild is that SIC as proposed in [16] is too slow to be employed, especially for $m \gg n$, which is the case for generic face alignment. However, as we showed above, the cost per iteration for Fast-SIC and Fast-Forward is on the order of a few times mN. This cost can be easily handled by current systems possibly allowing a close to real-time implementation.

Another reason for ruling out AAMs from unconstrained face alignment experiments is the fact that AAMs are not considered robust. All optimization problems considered in the previous sections are least-squares problems, and as it is well-known in computer vision least-squares combined with pixel intensities as features typically results in poor performance for data corrupted by outliers (e.g. sunglasses, occlusions, difficult illumination). Standard ways of dealing with outliers are robust features and robust norms. The problem with feature extraction is that it might slow down the speed of the fitting algorithm significantly especially when the dimensionality of the featured-based appearance model is large. The problem with robust norms is that scale parameters must be estimated (or percentage of outlier pixels must be predefined) and this task is not trivial.

We propose a third orthogonal direction for fitting AAMs in unconstrained conditions which is via training AAMs in-the-wild. In fact, this paper is one of the few that propose the combination of generative models plus training in-the-wild (plus robust optimization for model fitting). It turns out that this combination is very beneficial for unconstrained AAM fitting. Consider for example the image shown in Fig. 2 (a). This is a test image from the LFPW data set. This image was not seen during training, but similar images of unconstrained nature were used to train the shape and appearance model of an AAM. Fig. 2 (b) shows the reconstruction of the image from the appearance subspace. As we may see the appearance model is powerful enough to reconstruct the texture almost perfectly. Fitting with a robust algorithm (Fast-SIC in this case) gives the fitting result of Fig. 2 (a). This example illustrates why the results of the next section should not be considered too surprising.

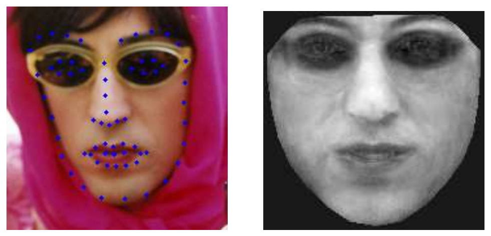
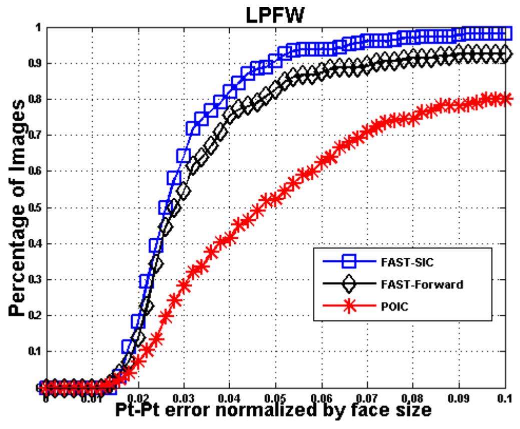

(a) (b)

Figure 2. (a) A face image from the test set of LFPW [3]. The image was not seen during training. Landmarks were detected by fitting the AAM using the Fast-SIC algorithm. (b) Reconstruction of the image from the appearance subspace. The appearance subspace is powerful because the AAM was built in the wild.

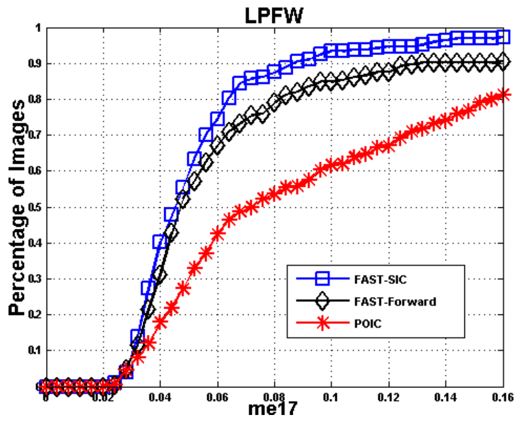
6. Results

The main target of our experiments was not to prove that AAM fitting is state-of-the-art in face alignment but to show that robust fitting plus training in-the-wild improves AAM fitting performance dramatically. For this reason, we did not attempt to use sophisticated shape priors for regularization, nor we employed robust features/appearance models or robust norms for improving performance. In all experiments, we used simple pixel intensities as features. Additionally, we note that we did not attempt to reproduce the results of any of current state-of-the-art methods because their implementation is not trivial and in most cases the code is not publicly available. Instead, we report the performance of AAMs using two very popular error measures and we refer the readers to these papers and the references therein for drawing comparisons with the algorithms described in this paper.

To facilitate fitting, we used a multi-resolution fitting approach with $m = 50, n = 3$ at the lowest level and $m = 200, n = 10$ at the highest. We found experimentally, that for robust fitting, m should be at least one order of magnitude greater than n. Note that in most AAM papers, m and n are chosen so that a fixed percentage of variance is fixed (most cases 95%). This may result in relatively large model sizes that are more difficult to optimize. Instead we chose the model parameters by running the al-



(a)



(b)

Figure 3. Fitting performance on LFPW. (a) mean point-to-point error (Euclidean) normalized by the face size vs percentage of test images. (b) me_{17} vs percentage of test images.

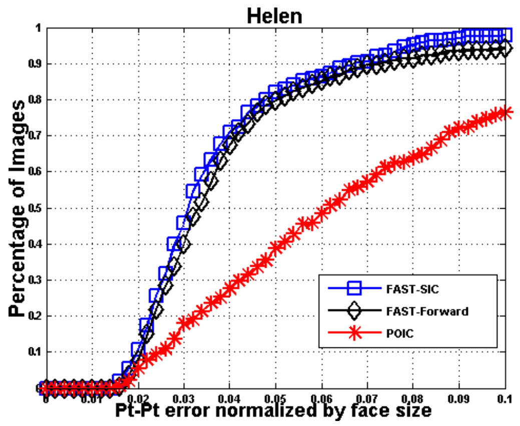


Figure 4. Fitting performance on Helen. The normalized mean point-to-point error (Euclidean) is plotted.

gorithms on a small validation set. For our experiments, we used the training set of LFPW to train the shape and appearance model of the AAM. The database consists of images from the web containing variations in pose, illumination, expression and occlusion. Because some URLs are no longer valid, about 800 out of 1.100 and 224 out of 300 images were available for training and testing, respectively. Because, the landmarks provided by LFPW are sparse, we annotated both training and test sets based on the point configuration of Multi-pie [8, 19]. In all cases, fitting was initialized by the face detector recently proposed in [24]. The first error measure that we used is the point-to-point error normalized by the face size as proposed in [24]. Similarly to [24], for this error measure, we produced the cumulative curve corresponding to the percentage of test images for which the error was less than a specific value. The second performance measure is the popular me_{17} [6]. Please see our publicly available implementation at `http://ibug.doc.ic.ac.uk/resources` for more details on our experimental setting (we provide source Matlab code for training, fitting and reproducing the results presented in this paper).

Fig. 3 (a) shows the obtained results. As we may observe Fast-SIC appears to perform better than Fast-Forward. One reason for this might be that in Fast-SIC the linearization is performed on the model which is smoother and less noisy than the test image. Both Fast-SIC and Fast-Forward largely outperform POIC. Finally, we note that in terms of landmark localization, Fast-SIC and Fast-Forward also outperform [24] but we did not include these results because [24] was trained on a different data set. Additionally, by comparing our results in Fig. 3 (b) with those in [3] (please note that we did not use exactly the same landmarks), we may observe that although at error 0.05 [3] performs better, at error 0.1 performance is similar although [3] used SIFT features [12] and more than twice as many training images as we did. Overall, although fair comparison is impossible, we believe that the proposed AAMs perform notably well and almost comparably with current systems. Finally, we performed a very challenging cross-database experiment: we annotated the test set of Helen [10] and fitted our AAMs that were trained on LFPW. Although both databases are in-the-wild, the faces of Helen seem to be much more natural, with more shape and appearance variation, and hence are more challenging to fit. Fig. 4 shows the obtained results. As expected, performance drops but still the fittings are satisfactory for the majority of images. Examples of fittings for both LFPW and Helen can be seen in 5.

7. Conclusions

We described a very simple framework based on (1) for deriving the optimization problems and solutions for fast AAM fitting in both inverse (Fast-SIC) and forward (Fast-Forward) coordinate frames. Based on the proposed framework, exact AAM fitting is no longer computationally prohibitive. Then, we proposed a new direction for employing AAMs in unconstrained conditions by means of

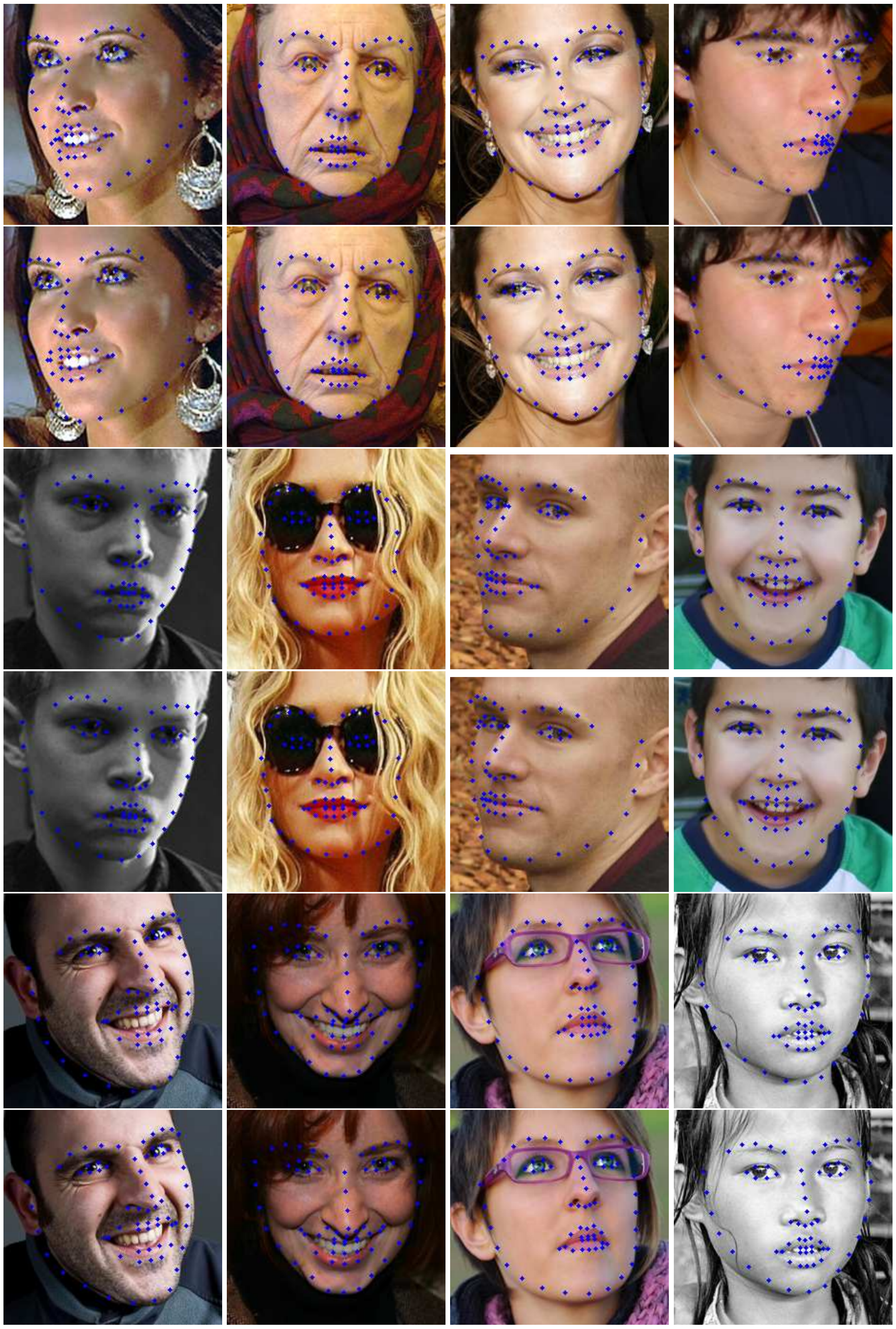


Figure 5. Examples of fittings from LFPW and Helen. Odd rows: Fast-SIC. Even rows: Fast-Forward.

training AAMs in-the-wild, and fitting using the proposed fast and exact algorithms. Our results show that although we did not use sophisticated shape priors, robust features or robust norms for improving performance, AAMs perform almost comparably with current state-of-the-art methods. We believe that our results are notable given that it is widely believed that (especially pixel-based) AAMs generalize poorly to unseen variations let alone images in-the-wild.

8. Acknowledgements

This work has been funded by the European Community 7th Framework Programme [FP7/2007-2013] under grant agreement no. 288235 (FROG). The work of Maja Pantic was originally funded by the European Research Council under the ERC Starting Grant agreement no. ERC-2007-StG-203143 (MAHNOB). The authors would also like to thank Epameinondas Antonakos for the many fruitful discussions on AAMs.

References

[1] S. Baker, R. Gross, and I. Matthews. Lucas-kanade 20 years on: Part 3. *Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-03-35*, 2003.

[2] A. U. Batur and M. H. Hayes. Adaptive active appearance models. *IEEE TIP*, 14(11):1707–1721, 2005.

[3] P. Belhumeur, D. Jacobs, D. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *CVPR*, 2011.

[4] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[5] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE TPAMI*, 23(6):681–685, 2001.

[6] D. Cristinacce and T. Cootes. Feature detection and tracking with constrained local models. In *BMVC*, 2006.

[7] R. Gross, I. Matthews, and S. Baker. Generic vs. person specific active appearance models. *Image and Vision Computing*, 23(12):1080–1093, 2005.

[8] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.

[9] G. D. Hager and P. N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE TPAMI*, 20(10):1025–1039, 1998.

[10] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang. Interactive facial feature localization. In *ECCV*. 2012.

[11] X. Liu. Generic face alignment using boosted appearance model. In *CVPR*, 2007.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[13] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *7th International Joint Conference on Artificial Intelligence*, 1981.

[14] S. Lucey, R. Navarathna, A. Ashraf, and S. Sridharan. Fourier lucas-kanade algorithm. *IEEE TPAMI*, 35(6):1383–1396, 2013.

[15] P. Martins, R. Caseiro, and J. Batista. Generative face alignment through 2.5 d active appearance models. *CVIU*, 2012.

[16] I. Matthews and S. Baker. Active appearance models revisited. *IJCV*, 60(2):135–164, 2004.

[17] K. Netzell and J. E. Solem. Efficient image inner products applied to active appearance models. In *ICPR*, 2008.

[18] G. Papandreou and P. Maragos. Adaptive and constrained algorithms for inverse compositional active appearance model fitting. In *CVPR*, 2008.

[19] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. A semi-automatic methodology for facial landmark annotation. In *CVPR Workshops*, 2013.

[20] J. Saragih and R. Gocke. Learning aam fitting through simulation. *Pattern Recognition*, 42(11):2628–2636, 2009.

[21] J. Saragih and R. Goecke. A nonlinear discriminative approach to aam fitting. In *ICCV*, 2007.

[22] H. Wu, X. Liu, and G. Doretto. Face alignment via boosted ranking model. In *CVPR*, 2008.

[23] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2d+3d active appearance models. In *CVPR*, 2004.

[24] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark estimation in the wild. In *CVPR*, 2012.