To appear at International Journal of Computer Vision (IJCV):
volume 128, issue 12, December 2020, pages 2828-2848.

Gradient Shape Model

Pedro Martins · João F. Henriques · Jorge Batista

Abstract For years, the so-called Constrained Local Model (CLM) and its variants have been the gold standard in face alignment tasks. The CLM combines an ensemble of local feature detectors whose locations are regularized by a shape model. Fitting such a model typically consists of an exhaustive local search using the detectors and a global optimization that finds the CLM's parameters that jointly maximize all the responses. However, one major drawback of CLMs is the inefficiency of the local search, which relies on a large amount of expensive convolutions. This paper introduces the Gradient Shape Model (GSM), a novel approach that addresses this limitation. We are able to align a similar CLM model without the need for any convolutions at all. We also use true analytical gradient and Hessian matrices, which are easy to compute, instead of their approximations. Our formulation is very general, allowing an optional 3D shape term to be seamlessly included. Additionally, we expand the GSM formulation through a cascade regression framework. This revised technique allows a substantially reduction in the complexity/dimensionality of the data term, making it possible to compute a denser, more accurate, regression step per cascade level. Experiments in several standard datasets show that our proposed models perform faster than state-of-the-art CLMs and better than recent cascade regression approaches.

Keywords Facial landmark localization · Face alignment · Constrained Local Model · CLM.

1 Introduction

Nonrigid face registration, sometimes known as facial alignment, plays a fundamental role in many computer vision tasks. Typical applications include visual tracking, recognition (identity and facial expression), biometric security, video compression, head pose estimation and many others. In general, the main goal of face registration consists of locating, with accuracy, the semantic structural facial landmarks (fiducial points) such as eyes, nose, mouth, chin, eyebrows, etc. Although this problem has been studied for years, it still is challenging to locate and consistently track subjects with previously unseen appearances and under unconstrained acquisition conditions (p.e. changes in pose, lighting, occlusion, resolution and focus).

Much has been done in the past, but since the introduction of the Active Shape Model (ASM) (Cootes et al, 1995) and shortly after with the Active Appearance Model (AAM) (Cootes et al, 2001) (Matthews and Baker, 2004) (Alabort-i-Medina and Zafeiriou, 2017) (Tzimiropoulos and Pantic, 2017), it became immensely popular to align faces by finding the parameters of a linear shape model.

Later, discriminative based extensions, namely the Constrained Local Model (CLM) (Cristinacce and Cootes, 2008) (Wang et al, 2008) (Saragih et al, 2010) (Cootes et al, 2012) (Asthana et al, 2013) (Martins et al, 2012) (Martins et al, 2014), have been proposed. These approaches improve the model's representation, moving

P. Martins
Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal. E-mail: pedromartins@isr.uc.pt

J. Batista
Department of Electrical and Computer Engineering, University of Coimbra, Coimbra, Portugal.
Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal. E-mail: batista@isr.uc.pt

J. F. Henriques
Visual Geometry Group, University of Oxford, Oxford, United Kingdom. E-mail: joao@robots.ox.ac.uk

away from the holistic texture representation, by only accounting for local correlations between features (around facial landmarks). In this architecture, both shape and appearance are combined by constraining an ensemble of local feature detectors to lie within the subspace spanned by a shape model. In practice, the CLM implements a two step fitting stage: (1) a local search and (2) a global optimization. The first step performs an exhaustive local search using an expert feature detector, obtaining response maps for each landmark (the data term). Afterwards, a global optimization strategy finds the model's parameters that jointly maximize all responses simultaneously (regularization term).

The most popular CLM optimization strategies propose to replace the true response maps by simple parametric forms: Weighted Peak Responses (Cootes et al, 1995), Gaussians Responses (Wang et al, 2008), Mixture of Gaussians (Gu and Kanade, 2008) or nonparametrically, using the mean-shift algorithm (Saragih et al, 2010), and perform a global optimization over these forms instead of the original response maps. Bayesian CLM had also been proposed (Paquet, 2009) (Martins et al, 2012) (Martins et al, 2014), where the shape parameters are inferred in a *Maximum A Posteriori* (MAP) sense. In BCLM (Paquet, 2009), a Gaussian inference was made using Gaussian assumptions in both likelihood/data and prior terms. The revised Bayesian CLM framework in (Martins et al, 2016) infers second order statistics of the parameters by formulating the overall alignment in terms of a Linear Dynamic System. The npBCLM (Martins et al, 2014) extends the previous by making non-parametric inference of the CLM's parameters.

Most of the previously CLM solutions established the core foundations of many other approaches. As example we highlight the use of enhanced shape models (Belhumeur et al, 2011) (Zhou et al, 2013b) (Zhu and Ramanan, 2012) (non-parametric and tree structured models, respectively), others like (Dantone et al, 2012) (Cristinacce and Cootes, 2007) (Valstar et al, 2010) that predict local landmark updates by regression, (Fanelli et al, 2013) (Tzimiropoulos et al, 2012) that use discriminative holistic appearances and (Tzimiropoulos and Pantic, 2014) that use a part-based generative appearance model.

Recently, a different paradigm has emerged, the so-called cascaded regression techniques (Cao et al, 2012) (Xiong and De la Torre, 2013) (Xiong and De la Torre, 2015) (Tzimiropoulos, 2015) (Burgos-Artizzu et al, 2013) (Kazemi and Sullivan, 2014) (Lee et al, 2015) (Jourabloo and Liu, 2015) (Zhu et al, 2015) (Sánchez-Lozano et al, 2018). These methods learn a series of averaged descent directions by performing offline simulations. This process is usually chained throughout a cascade (iterative regression), thus consisting of learning an ensemble of regressors, where each regressor relates the extracted features at a given image location with the updates to be made to the control parameters. Fitting such a cascaded model simply consists of applying recursively each regressor (from an initial estimate) and keep collecting the parameters updates. In general, these methods differ from each other by the way as the regression is accomplished, p.e. boosted regression (Cao et al, 2012) (Ren et al, 2014) (Burgos-Artizzu et al, 2013) (Kazemi and Sullivan, 2014), least-squares regression (Xiong and De la Torre, 2013) (Xiong and De la Torre, 2015) (Sánchez-Lozano et al, 2018) or Gaussian Processes regression (Lee et al, 2015).

Still under this multi-stage paradigm, it is worth mentioning some of the deep neural networks approaches, in particular, the Convolutional Neural Networks (CNNs) based methods. Several techniques have been proposed, we highlight just a few. In (Sun et al, 2013) a standard CNN based regression approach was used to locate facial landmarks. Later, in (Zhang et al, 2014a), successive stacked autoencoders were arranged in a coarse-to-fine strategy. The Tasks-Constrained Deep Convolutional Network (TCDCN) (Zhang et al, 2014b) (Zhang et al, 2016) proposes a multi-task learning framework for joint facial landmark localization and attribute classification, such as gender, expression and pose. The detection of such atributes was used to assist in the face alignment procedure. In (Zhou et al, 2013a), a coarse-to-fine CNN based framework was used to iteratively refine a subset of facial landmarks (in local regions defined by previous network levels). Extensions of this work (Huang et al, 2015) (Fan and Zhou, 2016), made the foundations for the Face++ (Face++, 2018) comercial software. The Mnemonic Descent Method (MDM) (Trigeorgis et al, 2016) combines cascaded regression with feature learning (using CNNs). This method extends SDM (Xiong and De la Torre, 2013) with a Recurrent Neural Network (RNN). Deep learning extensions of CLMs and Deformable Parts Models (DPM) (Zhu and Ramanan, 2012) have also been proposed (Zadeh et al, 2017) and (Songsri-in et al, 2018), respectively. The first approach, referred as the Convolutional Experts Constrained Local Model (CE-CLM) (Zadeh et al, 2017) combines several CNNs structures per landmark (acting as multiple local detectors) and the global CLM like shape regularization. The Face Alignment Network (FAN) (Bulat and Tzimiropoulos, 2017b) applies convolutional heatmap regression using a stack of Hourglass networks (Newell et al, 2016). Finally, in (Bulat and Tzimiropoulos, 2017a), the same authors explored binarized Hourglass-like Convolutional Network structures to improve the computational performance.

Despite all previous achievements, in our understanding, the CLM still remains one of the most influential techniques to locate facial features. However, the major drawback of all CLM methods is the inefficiency of the local search (data term), which relies on a large amount of expensive convolutions required for each landmark. Evaluating this data term consumes around of 90% of the total computational time.

The first part of this paper tackles the CLM's foundations and introduces the Gradient Shape Model (GSM), that addresses the previous limitation. The GSM is able to align a similar CLM model without the need for any convolutions at all. Moreover, we use true analytical gradient and Hessian matrices, which are easy to compute, instead of their approximations. We additionally show how an optional 3D shape term can be seamlessly included, effectively constraining the 2D model search and improving the overall fitting performance.

Later on, we revisit the cascade regression formulation. While such methods can be considered efficient concerning online model fitting, the learning stage (ensemble of regressors) is a different story. Learning each cascade level involves to manage massive data matrices (that holds the extracted features from every training image and every simulation made) and the regression process itself.

This paper also presents a cascade regression extension of the GSM, referred as GSM-CR. In this novel approach, which adopts a least-squares regression approach (Xiong and De la Torre, 2013), we show how to efficiently extract local gradient features and substantially reduce the dimensionality of the data term, overcoming the computational burden of such learning methods. Moreover, since the performance of the cascade regression methods is deeply related to the density of the simulations, our approach is therefore more accurate for the same computational cost.

1.1 Contributions

This paper makes the following contributions:

1. We introduce the Gradient Shape Model (GSM) that aims to replace the CLM's exhaustive local search (convolutions) with a fast local gradient estimate, that is trained discriminatively. We also show how second order gradients can easily be computed while designing an unifying Newton optimization.
2. A conceptually simpler cost function is proposed, combining the local cost with the log-likelihood of a Gaussian shape prior. Unlike previous work, we do not project shapes into a low dimensional subspace. Instead we rely on a simple regularization of the covariance to prevent over fitting.
3. True analytical gradient and Hessian matrices are computed, unlike previously proposed CLM approximations (section 3).
4. We highlight the extensibility of this formulation, by proposing an additional 3D Gaussian shape prior (section 4).
5. Finally, we expand the GSM approach into a cascade regression formulation (section 5). We show how our methodology leads to a substantially reduction in the complexity/dimensionality of the data term, making it possible to compute a denser, more accurate, regression step per cascade level.

The proposed methods are validated with extensive experiments (section 6), showing a significant speed-up (by at least an order of magnitude) over other CLM techniques. Similarly, the extended GSM-CR formulation outperforms other cascade regression based strategies in training times and overall accuracy.

1.2 Outline

The remaining of the paper is organized as follows: section 2 briefly reviews the CLM formulation. Our GSM formulation is presented in section 3. The 2D+3D GSM extension appears right after in section 4. The GSM-CR cascade regression strategy is described in section 5. Finally, sections 6 and 7 show the experimental results and the conclusions, respectively.

2 Revisiting of the Constrained Local Model

The Constrained Local Model (CLM) consist of a collection of v local landmark detectors, denoted here as $\{\mathbf{h}_i\}_1^v$, whose locations are regularized by a linear shape model. The following sections briefly describe the shape, the local detectors and some of the most popular fitting strategies.

2.1 Shape Model

The 2D shape is represented by the locations of a set of v landmarks with a $2v$ dimensional vector $\mathbf{s} = (x_1, \ldots, x_v, y_1, \ldots, y_v)^T$. The shape model itself is a Point Distribution Model (PDM) (Cootes and Taylor, 2004) that describes each shape by following a linear parametric model

$$\mathbf{s} \approx \mathcal{S}\left(\mathbf{s}_0 + \Phi \mathbf{b}_s, \boldsymbol{\theta}\right) \tag{1}$$

where $\mathbf{s}_0$ is the mean shape (also known as the base mesh), Φ is the shape subspace matrix holding n eigenvectors (that resulted from applying Principal Components Analysis on a set of normalized training shapes) and $\mathbf{b}_s \in \mathbb{R}^n$ is the shape parameters vector representing the mixing weights. Please refer to (Cootes and Taylor, 2004) for additional details on PDMs. The ability to model the 2D rigid pose is included by the similarity transformation $\mathcal{S}(\mathbf{s}, \boldsymbol{\theta})$ where each landmark point $\mathbf{s}_i = (x_i, y_i)^T$ is warped around the base mesh by

$$\mathcal{S}(\mathbf{s}_i, \boldsymbol{\theta}) = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} (\mathbf{s}_i - \mathbf{s}_m) + \begin{bmatrix} t_x \\ t_y \end{bmatrix} + \mathbf{s}_m \tag{2}$$

where $\boldsymbol{\theta} = [a,\, b,\, t_x,\, t_y]^T$ is the pose parameters vector with $a = s\cos(\theta)$, $b = s\sin(\theta)$ being combined scale and rotation and (t_x, t_y) the translations, all expressed w.r.t. $\mathbf{s}_0$[1]. The $\mathbf{s}_m = [\mathrm{mean}(\mathbf{s}_0^x),\ \mathrm{mean}(\mathbf{s}_0^y)]^T$ is the mean shape center of mass. The full CLM parameters are usually packed into a single set represented by $\mathbf{b} = [\mathbf{b}_s^T | \boldsymbol{\theta}^T]^T \in \mathbb{R}^{n+4}$.

2.2 Local Detectors

Different kinds of local detectors have been used within the CLM framework (Cristinacce and Cootes, 2006) (Wang et al, 2008) (Cootes et al, 2012) (Baltrušaitis et al, 2013) (Martins et al, 2012). The initial formulation used generative based templates (Cristinacce and Cootes, 2006) (Cristinacce and Cootes, 2008). Afterwards, the usage of discriminative based detectors were introduced (Wang et al, 2008) (Saragih et al, 2009), in particular, linear SVMs trained with aligned vs. misaligned patch examples.

Recently, correlation filters have also been employed (Martins et al, 2012) (Martins et al, 2014), in particular the MOSSE filter (Bolme et al, 2010). The MOSSE filter, when compared to the previous, has several advantages: it extends the linear SVM scalar labels with 2D maps of real valued labels (meaning that a large amount of virtual samples are included in the the training stage (Henriques et al, 2013)); it allows discriminative training using only aligned (positive) data; it maintains its linear nature, and finally, it performs better in some cases (Martins et al, 2016).

In this work, we follow the extended multidimensional MOSSE formulation, namely the Multi-Channel Correlation Filter (MCCF) (Galoogahi et al, 2013) (Boddeti et al, 2013). Briefly, finding each MCCF filter $\mathbf{h}_i$ (for the i^{th} landmark), can be posed by solving the following linear regression problem

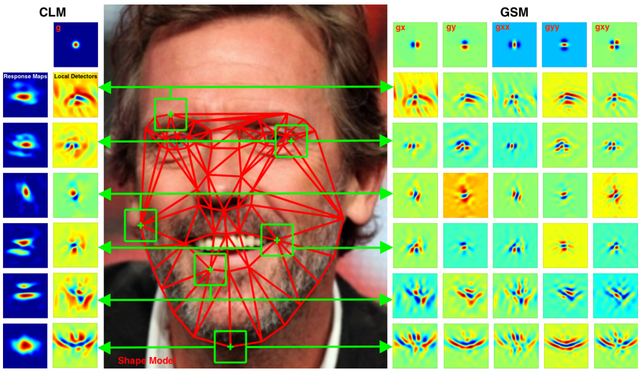


Fig. 1 Comparison between the proposed Gradient Shape Model (GSM) and the Constrained Local Model (CLM), shown in right and left, respectively. Fitting a CLM relies in a two step strategy: locally scan with the local detectors, producing response maps, and then jointly optimize the shape parameters that maximize all the responses. The GSM, in contrast, aims to estimate directly the local gradients (and Hessian) and thus avoids the costly exhaustive local search (no response maps are required).

$$\arg\min_{\mathbf{h}_i} \sum_{j=1}^{N} \sum_{k=1}^{D} \left(\mathbf{f}_j^{(k)} \circledast \mathbf{h}_i^{(k)} - \mathbf{g}_j \right)^2 + \epsilon \sum_{k=1}^{D} ||\mathbf{h}_i^{(k)}||^2 \tag{3}$$

where $(\circledast)$ is the correlation operator, $\mathbf{f}_j^{(k)}$ represents features extracted from the k^{th} image channel (D in total) of the j^{th} training patch (with $L \times L$ size), N is the total number of examples and ϵ is a regularization parameter.

In eq. 3, $\mathbf{g}_j$ stands for the desired target correlation in the j^{th} example (the labels), which is typically defined as a 2D Gaussian (when aiming to predict locations)

$$\mathbf{g}_j(x, y) = \exp\left(\frac{-\left(x^2 + y^2\right)}{2\sigma_h^2} \right) \tag{4}$$

where (x, y) are control points over the support region and σ_h is the standard deviation (that controls the ratio between positive and negative virtual samples). Eq. 3 can be efficiently solved in the Fourier domain (where convolutions become products), with the solution being

$$\mathbf{h}_i = \mathcal{F}^{-1} \left\{ (\epsilon \mathbf{I} + \sum_{j=1}^{N} \Xi_j^T \Xi_j)^{-1} \sum_{j=1}^{N} \Xi_j^T (\mathbf{1}_D \otimes \mathcal{F}\{\mathbf{g}_j\}) \right\}^* \tag{5}$$

where $\mathcal{F}$ represents the 2D Fourier transform, $\Xi_j = \left[\mathrm{diag}(\mathcal{F}\{\mathbf{f}_j^{(1)}\})^T, \ldots, \mathrm{diag}(\mathcal{F}\{\mathbf{f}_j^{(D)}\})^T \right]$ defines the Fourier

[1] It is worth mentioning that some authors use a slightly different pose parametrization ($\boldsymbol{\theta}' = [a-1,\, b,\, t_x,\, t_y]^T$) that allows to append to Φ a special set of 4 eigenvectors that linearly model the 2D pose (Matthews and Baker, 2004).

transform of the 'extended' data matrix, $\mathbf{1}_D$ is a D dimensional vector with ones, $\mathbf{I}$ is an identity matrix with appropriate dimensions and the symbols $(*), (^T), (\otimes)$ represent the complex conjugate, conjugate transpose and Kronecker product, respectively.

A closer look on Ξ_j shows that it is a heavily sparse matrix. According, an efficient solution can be found through variable re-ordering (please refer to (Henriques et al, 2013) (Galoogahi et al, 2013) for additional details).

Taking into account that most CLM approaches rely exclusively in single channel features (grey level), it is worth to mention that when $D = 1$, the eq. 5 reduces to

$$\mathbf{h}_i^{(1)} = \mathcal{F}^{-1}\left\{\frac{\sum_{j=1}^{N}\mathcal{F}\{\mathbf{g}_j\}\odot\mathcal{F}\{\mathbf{f}_j^{(1)}\}^*}{\sum_{j=1}^{N}\mathcal{F}\{\mathbf{f}_j^{(1)}\}\odot\mathcal{F}\{\mathbf{f}_j^{(1)}\}^*+\epsilon}\right\}^* \tag{6}$$

where $\odot$ symbol stands for the Hadamard product.

2.3 CLM Alignment

In general, CLMs seek to find the optimal set of parameters that maximize the cost

$$\hat{\mathbf{b}} = \arg\max_{\mathbf{b}} \sum_{i=1}^{v} D_i\left(\mathbf{I}(\mathbf{s}_i), \mathbf{b}\right) - \lambda_0\ \mathbf{b}^T \Sigma_{\mathbf{b}}^{-1}\mathbf{b} \tag{7}$$

where the first term (data term) is the summation of a measure of goodness in matching the appearance across landmarks and the second term (regularization) penalizes the shape model against large deformations. The $\Sigma_{\mathbf{b}}$ is the covariance of the shape parameters, assumed to be independent and Gaussian distributed (diagonal matrix with PCA eigenvalues). The parameter λ_0 is a scalar that controls the strength of the regularization. A higher lambda 'forces' the overall solution to follow a more rigid shape model and, by contrast, a lower lambda loosens the model. In theory, the value of λ_0 can be determined by computing the ratio between the data and the regularization term across the training data (evaluating local landmark detections at training shape locations), however, in practice a bit of tuning may be required (see section 6).

Normally, the landmark detectors are designed to operate at a given scale. The CLM framework deals with this by including a warp normalization step, in particular, a similarity transformation into the base mesh. In this sense, the data term can be defined as

$$D_i(\mathbf{I}(\mathbf{s}_i), \mathbf{b}) = \mathcal{S}^{-1}(\mathbf{I}(\mathbf{s}_i), \boldsymbol{\theta}) * \mathbf{h}_i, \tag{8}$$

where with some abuse of notation we denote $\mathcal{S}^{-1}(\mathbf{I}(\mathbf{s}_i), \boldsymbol{\theta})$ to be the i^{th} image patch, centred at $\mathbf{s}_i$, sampled at a image previously warped (inverse similarity with parameters $\boldsymbol{\theta}$). In this context, $\mathbf{s}_i$ follows eq. 1 without pose. Finally, $\mathbf{h}_i$ is the i^{th} local detector, defined in section 2.2. Please see figure 1. Note that around 90% of the total computational cost consists of evaluating the data term (convolutions).

The optimization, in eq. 7, is usually posed as an minimization problem

$$\min_{\mathbf{b}} - \sum_{i=1}^{v} D_i\left(\mathbf{I}(\mathbf{s}_i), \mathbf{b}\right) + \lambda_0\ \mathbf{b}^T \Sigma_{\mathbf{b}}^{-1}\mathbf{b} \tag{9}$$

and solved using Newton's methods by iterating

$$\mathbf{b}^{(\tau+1)} \leftarrow \mathbf{b}^{(\tau)} - \gamma \underbrace{\mathbf{H}^{-1}(\mathbf{b})\mathbf{J}(\mathbf{b})}_{\Delta\mathbf{b}} \tag{10}$$

where $\mathbf{J}(\mathbf{b}) \in \mathbb{R}^{2v\times(n+4)}$, $\mathbf{H}(\mathbf{b}) \in \mathbb{R}^{(n+4)\times(n+4)}$ and γ are the Jacobian, the Hessian matrix and the step size, respectively. In general, and to the best of our knowledge, only Gauss-Newton approximations to the Hessian have been used so far.

Several CLM alignment solutions have been proposed. The main difference between them can be essentially reduced to the way the data term is treated. In the following subsections we briefly describe some of the most popular strategies.

2.3.1 Active Shape Model (ASM)

The ASM (Cootes et al, 1995) simply takes as the solution the location where the response map D_i has its maximum score $\mu_i = \arg\max D_i$ and its uncertainty is set to be inversely proportional to the peak value $w_i^{-1} = \max D_i$, that leads to

$$\Delta\mathbf{b} = \mathbf{H}_{\mathrm{ASM}}^{-1}\left(\lambda_0\Sigma_{\mathbf{b}}^{-1}\mathbf{b} + \sum_{i=1}^{v} w_i\mathbf{J}_i^T(\mathbf{s}_i - \mu_i)\right) \tag{11}$$

where $\mathbf{J}_i$ is the Jacobian sub-matrix $(2\times(n+4))$ regarding the i^{th} landmark and $\mathbf{H}_{\mathrm{ASM}} = \lambda_0\Sigma_{\mathbf{b}}^{-1} + \sum_{i=1}^{v} w_i\mathbf{J}_i^T\mathbf{J}_i$ is the Hessian matrix.

2.3.2 Convex Quadratic Fitting (CQF)

The CQF (Wang et al, 2008) considers the response maps to be fully approximated by a Gaussian distribution. The problem, in this case, consists of fitting a 2D Gaussian to weighted data, $\mu_i = \mathbf{s}_i D_i$ and $\Sigma_{D_i} = \mathrm{cov}(\mathbf{s}_i D_i)$ where the update is

$$\Delta\mathbf{b} = \mathbf{H}_{\mathrm{CQF}}^{-1}\left(\lambda_0\Sigma_{\mathbf{b}}^{-1}\mathbf{b} + \sum_{i=1}^{v} \mathbf{J}_i^T\Sigma_{D_i}^{-1}(\mathbf{s}_i - \mu_i)\right) \tag{12}$$

with $\mathbf{H}_{\mathrm{CQF}} = \lambda_0\Sigma_{\mathbf{b}}^{-1} + \sum_{i=1}^{v} \mathbf{J}_i^T\Sigma_{D_i}^{-1}\mathbf{J}_i$.

2.3.3 Subspace Constrained Mean-Shifts (SCMS)

The SCMS (Saragih et al, 2010) approximates D_i by a non-parametric representation using a Kernel Density Estimator (KDE) (Silverman, 1986) (isotropic Gaussian kernel). The mean-shift algorithm (Comaniciu and Meer, 2002), with a decreasing annealing bandwidth schedule, was used to maximize over the KDE. The SCMS update is given by

$$\Delta \mathbf{b} = \mathbf{H}_{\text{SCMS}}^{-1} \left(\lambda_0 \Sigma_{\mathbf{b}}^{-1} \mathbf{b} - \mathbf{J}^T \nu \right) \qquad (13)$$

where each element ν_i is the i^{th} mean-shift landmark update ($\nu \in \mathbb{R}^{2v}$) and $\mathbf{H}_{\text{SCMS}} = \lambda_0 \Sigma_{\mathbf{b}}^{-1} + \mathbf{J}^T \mathbf{J}$.

Finally, we point out that a detailed computational efficiency comparison can be seen later, in section 6.3.

3 Gradient Shape Model

The Gradient Shape Model (GSM) aims to reduce the computational cost of aligning an image by avoiding the need for convolutions. This is accomplished by replacing the exhaustive scan (and the local optimization strategy) around each landmark by a gradient based search. Figure 1 highlights the difference between deformable model fitting with a standard CLM and with our proposed method (i.e. exhaustive local search vs gradient search).

3.1 The Alignment Goal

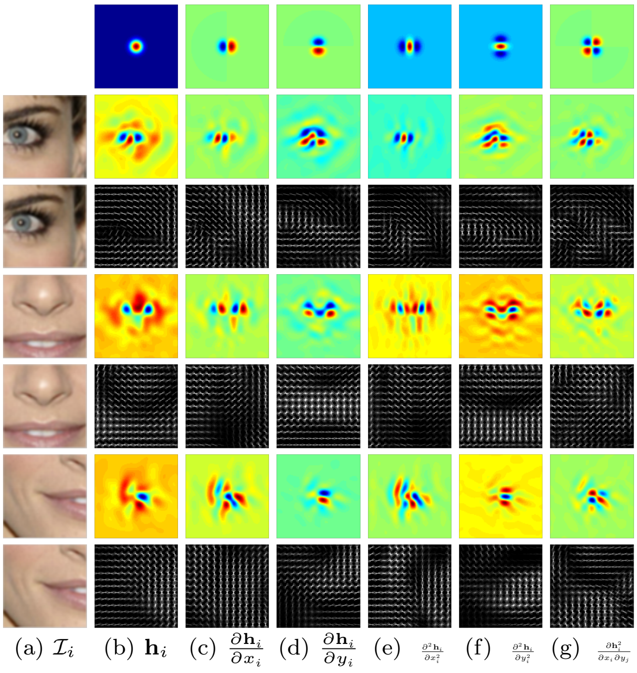
The GSM leaves out the lower dimensional representation of the shape (the PDM) and uses as latent variables the shape itself (which can be seen as a PDM without any PCA reduction). The GSM seeks to minimize the cost function, $f \in \mathbb{R}^{2v+4} \to \mathbb{R}$, given by

$$f(\mathbf{s}, \boldsymbol{\theta}) = -\sum_{i=1}^{v} D_i(\mathbf{I}(\mathbf{s}_i), \boldsymbol{\theta}) + \lambda_1 R(\mathbf{s}, \boldsymbol{\theta}) \qquad (14)$$

where D_i is a similar data term (in the sense that inverse warping with parameters $\boldsymbol{\theta}$ is still required) and $R(\mathbf{s}, \boldsymbol{\theta})$ is a new regularization term given by

$$R(\mathbf{s}, \boldsymbol{\theta}) = \left(\mathcal{S}(\mathbf{s}, \boldsymbol{\theta}) - \mathbf{s}_0 \right)^T \Sigma_{\mathbf{s}}^{-1} \left(\mathcal{S}(\mathbf{s}, \boldsymbol{\theta}) - \mathbf{s}_0 \right). \qquad (15)$$

Now $\mathbf{s}$ represents the shape expressed at the image frame, which requires mapping it with $\mathcal{S}(\mathbf{s}, \boldsymbol{\theta})$ to properly apply a regularization. The $\Sigma_{\mathbf{s}}$ is the full $2v \times 2v$ covariance of all shapes (precomputed) and λ_1 is a scalar regularization weight.



(a) $\mathcal{I}_i$ (b) $\mathbf{h}_i$ (c) $\frac{\partial \mathbf{h}_i}{\partial x_i}$ (d) $\frac{\partial \mathbf{h}_i}{\partial y_i}$ (e) $\frac{\partial^2 \mathbf{h}_i}{\partial x_i^2}$ (f) $\frac{\partial^2 \mathbf{h}_i}{\partial y_i^2}$ (g) $\frac{\partial \mathbf{h}_i^2}{\partial x_i \partial y_j}$

Fig. 2 Local detector gradients for the eye, nose and mouth corner regions, respectively (displaying both grey level and HoG based representations). The (b) column shows the typical detectors used in CLM (when $\mathbf{g}$ is a standard Gaussian). The regression targets $\mathbf{g}$, $\frac{\partial \mathbf{g}}{\partial x_i}$, $\frac{\partial \mathbf{g}}{\partial y_i}$, $\frac{\partial^2 \mathbf{g}}{\partial x_i^2}$, $\frac{\partial^2 \mathbf{g}}{\partial y_i^2}$ and $\frac{\partial^2 \mathbf{g}}{\partial x_i y_i}$ are represented in the first row of images.

The goal is to optimize $\mathbf{s}$ and $\boldsymbol{\theta}$ using a true Newton approach. By linearity, the overall gradient will be the summation of two parts (data and regularization)

$$\nabla_f(\mathbf{s}, \boldsymbol{\theta}) = \nabla_{\text{D}}(\mathbf{s}) + \lambda_1 \nabla_{\text{R}}(\mathbf{s}, \boldsymbol{\theta}) \qquad (16)$$

similarly, the Hessian will be

$$\mathbf{H}_f(\mathbf{s}, \boldsymbol{\theta}) = \mathbf{H}_{\text{D}}(\mathbf{s}) + \lambda_1 \mathbf{H}_{\text{R}}(\mathbf{s}, \boldsymbol{\theta}). \qquad (17)$$

The following sections will cover each of the previous terms, independently.

3.2 Gradient of the Data Term

For the sake of notation, let's define a vectorized version of the sampled warped patch, from eq. 8, as $\mathcal{I}_i \equiv \mathcal{S}^{-1}(\mathbf{I}(\mathbf{s}_i), \boldsymbol{\theta})$. The differentiation property of the convolution allows us to write $\frac{\partial}{\partial \mathbf{s}_i}(\mathcal{I}_i * \mathbf{h}_i) = \frac{\partial \mathcal{I}_i}{\partial \mathbf{s}_i} * \mathbf{h}_i = \mathcal{I}_i * \frac{\partial \mathbf{h}_i}{\partial \mathbf{s}_i}$. However, in this approach, we are only concerned when both convolution signals are aligned (as Newton methods only care about the evaluation of gradients at the operating point) or $\frac{\partial}{\partial \mathbf{s}_i}(\mathcal{I}_i * \mathbf{h}_i)|_{\mathbf{s}_i} = \mathcal{I}_i^T \frac{\partial \mathbf{h}_i}{\partial \mathbf{s}_i}$, meaning that the gradient of the data term can be easily evaluated by a simple dot product between the sampled (pose-normalized) patch and the gradient of the local detector (both vectorized).

Recalling eq. 5, we see that the filter $\mathbf{h}_i$ is a linear function of the desired correlation $\mathbf{g}$, as the Fourier transform (and its inverse) is linear (this remark is easier to see in eq. 6). Formally, defining $\boldsymbol{\Omega}$ as the Discrete Fourier Transform (DFT) matrix and considering that $\mathcal{F}\{\mathbf{f}\} = \boldsymbol{\Omega}\,\mathbf{f}$ and $\mathcal{F}^{-1}\{\mathbf{f}\} = \boldsymbol{\Omega}^{-1}\mathbf{f} = \boldsymbol{\Omega}^T\mathbf{f}$ we can rewrite any of the eqs 5 or 6 as

$$\mathbf{h}_i = \boldsymbol{\Omega}^T\mathbf{K}_1\,\boldsymbol{\Omega}\,\mathbf{g} = \mathbf{K}_2\,\mathbf{g} \qquad (18)$$

where $\mathbf{K}_1$ and $\mathbf{K}_2$ are constant matrices that do not depend on the spatial dimensions.

According, evaluating the gradient of the local detector only requires to build the filter with the gradient of $\mathbf{g}$ (by modifying the regression targets/labels). Additionally the minus term of the data term (in eq. 14) can be even absorbed

$$\frac{\partial(-\mathbf{g})}{\partial x_i} = \frac{x_i}{\sigma_h^2}\exp\left(\frac{-\left(x_i^2+y_i^2\right)}{2\sigma_h^2}\right), \qquad (19)$$

similarly for the vertical component

$$\frac{\partial(-\mathbf{g})}{\partial y_i} = \frac{y_i}{\sigma_h^2}\exp\left(\frac{-\left(x_i^2+y_i^2\right)}{2\sigma_h^2}\right). \qquad (20)$$

Following this approach the overall gradient of the data term (vector $2v+4$) can be written as

$$\nabla_{\mathrm{D}}(\mathbf{s}) = \left[\mathcal{I}_1^T\frac{\partial\mathbf{h}_1}{\partial x_1}\ \cdots\ \mathcal{I}_v^T\frac{\partial\mathbf{h}_v}{\partial x_v}\ \mathcal{I}_1^T\frac{\partial\mathbf{h}_1}{\partial y_1}\ \cdots\ \mathcal{I}_v^T\frac{\partial\mathbf{h}_v}{\partial y_v}\ \mathbf{0}_4\right]^T. \qquad (21)$$

Figure 2 (a-d) shows examples for sampled patches ($\mathcal{I}_i$), the standard MCCF filter ($\mathbf{h}_i$), and their horizontal ($\frac{\partial h_i}{\partial x_i}$) and vertical gradients ($\frac{\partial h_i}{\partial y_i}$), respectively. Both grey level and HoG (Dalal and Triggs, 2005) features representations are shown. The first row of images highlight the regression targets.

3.3 Gradient of the Regularization Term

Naturally, the regularization term (in eq. 15) has a gradient given by

$$\nabla_{\mathrm{R}}(\mathbf{s},\boldsymbol{\theta}) = \left[\frac{\partial R}{\partial\mathbf{s}}\ \frac{\partial R}{\partial a}\ \frac{\partial R}{\partial b}\ \frac{\partial R}{\partial t_x}\ \frac{\partial R}{\partial t_y}\right]^T. \qquad (22)$$

A closer look to eq. 15 shows that $R(\mathbf{s},\boldsymbol{\theta})$ is in fact a composition of two functions. The first maps the shape into the base mesh and the second is simply a multidimensional Gaussian distribution. Defining the shape expressed at the base mesh as

$$\mathbf{s}_{\mathrm{BM}} = \mathcal{S}(\mathbf{s},\boldsymbol{\theta}) \qquad (23)$$

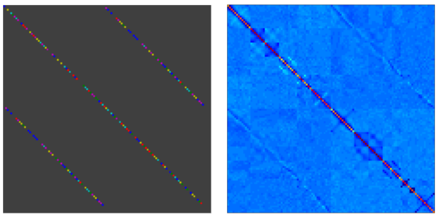

(a) $\mathbf{H}_{\mathrm{D}}(\mathbf{s})$ (b) $\mathbf{H}_{\mathrm{R}}(\mathbf{s},\boldsymbol{\theta})$

Fig. 3 Visual representation of the Hessian matrix (eq. 17).

and noting that $\Sigma_{\mathbf{s}}^{-1}$ is a symmetric matrix, it follows that

$$\frac{\partial R}{\partial\mathbf{s}} = 2\Sigma_{\mathbf{s}}^{-1}\left(\mathbf{s}_{\mathrm{BM}} - \mathbf{s}_0\right). \qquad (24)$$

The remaining partial differentials for the pose parameters ($\boldsymbol{\theta}$) are all scalar values, and can be found by applying the chain rule, as

$$\frac{\partial R}{\partial\boldsymbol{\theta}_j} = \left(\frac{\partial R}{\partial\mathbf{s}_{\mathrm{BM}}}\right)^T\frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial\boldsymbol{\theta}_j}, \quad j = 1,\ldots,4 \quad \text{with} \qquad (25)$$

$$\frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial a} = \mathbf{s}-\mathbf{s}_m, \quad \frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial b} = \begin{bmatrix}\mathbf{s}_m^y - \mathbf{s}^y\\ \mathbf{s}^x - \mathbf{s}_m^x\end{bmatrix}, \qquad (26)$$

$$\frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial t_x} = \begin{bmatrix}\mathbf{1}_v\\ \mathbf{0}_v\end{bmatrix}, \quad \frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial t_y} = \begin{bmatrix}\mathbf{0}_v\\ \mathbf{1}_v\end{bmatrix}, \qquad (27)$$

that can be easily taken from eq. 2. The symbols $\mathbf{0}_v$ and $\mathbf{1}_v$ are v sized vectors with zeros and ones, respectively. The $\mathbf{s}_m$ was previously defined in eq. 2 and stands for the mean shape center of mass. However, from now on, we expand this representation into a $2v$ vector as $\mathbf{s}_m = \begin{bmatrix}\mathbf{s}_m^x\\ \mathbf{s}_m^y\end{bmatrix}$, where $\mathbf{s}_m^x = \mathbf{1}_v\left(\frac{1}{v}\sum_i^v\mathbf{s}_{0_i}^x\right)$ and $\mathbf{s}_m^y = \mathbf{1}_v\left(\frac{1}{v}\sum_i^v\mathbf{s}_{0_i}^y\right)$ are vectorized versions of the 2D coordinates of center of mass.

In short, evaluating $\nabla_{\mathrm{R}}(\mathbf{s},\boldsymbol{\theta})$ only involves a few computations using eq. 24 that afterwards becomes constant to use in eq. 25. Moreover, $\Sigma_{\mathbf{s}}^{-1}$, $\mathbf{s}_m$, $\frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial t_x}$ and $\frac{\partial\mathbf{s}_{\mathrm{BM}}}{\partial t_y}$ can be precomputed.

3.4 Hessian of the Data Term

Evaluating the Hessian of the data term follows a similar process as before in section 3.2, only with a few more partial differentials. For instance, computing $\frac{\partial^2\mathbf{h}_i}{\partial x_i^2}$ requires using eq. 5 with $\mathbf{g}$ being the second order differential of a Gaussian w.r.t. x_i

$$\frac{\partial^2(-\mathbf{g})}{\partial x_i^2} = \left(\frac{1}{\sigma_h^2}-\frac{x_i^2}{\sigma_h^4}\right)\exp\left(\frac{-\left(x_i^2+y_i^2\right)}{2\sigma_h^2}\right), \qquad (28)$$

similarly for $\frac{\partial^2 \mathbf{h}_i}{\partial y_i^2}$

$$\frac{\partial^2(-\mathbf{g})}{\partial y_i^2} = \left(\frac{1}{\sigma_h^2} - \frac{y_i^2}{\sigma_h^4}\right) \exp\left(\frac{-\left(x_i^2 + y_i^2\right)}{2\sigma_h^2}\right), \tag{29}$$

and the mixed partials (equal by the Schwarz' theorem)

$$\frac{\partial^2(-\mathbf{g})}{\partial x_i \partial y_i} = \frac{\partial^2(-\mathbf{g})}{\partial y_i \partial x_i} = \frac{x_i y_i}{\sigma_h^4} \exp\left(\frac{-\left(x_i^2 + y_i^2\right)}{2\sigma_h^2}\right). \tag{30}$$

In general, the (i, j) element of the Hessian matrix (data part) is given by

$$\mathbf{H}_{\mathrm{D}_{ij}}(\mathbf{s}) = \mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_j}{\partial \mathbf{s}_i \partial \mathbf{s}_j}. \tag{31}$$

These 2^{nd} order partial differentials are only non zero at the main diagonal and at two other smaller diagonals (when $i = j + v$ or $j = i + v$, which are identical due to equality of mixed partials), see figure 3(a). Formally, this Hessian term can be written as

$$\mathbf{H}_{\mathrm{D}}(\mathbf{s}) = \begin{bmatrix} \operatorname{diag}\left(\mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial x_i^2}\right) & \operatorname{diag}\left(\mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial x_i \partial y_i}\right) & \mathbf{0}_{v \times 4} \\ \operatorname{diag}\left(\mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial y_i \partial x_i}\right) & \operatorname{diag}\left(\mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial y_i^2}\right) & \mathbf{0}_{v \times 4} \\ \mathbf{0}_{4 \times v} & \mathbf{0}_{4 \times v} & \mathbf{0}_{4 \times 4} \end{bmatrix}. \tag{32}$$

Note that only $(2v + v)$ dot products between image patches and (precomputed) filters $\frac{\partial^2 \mathbf{h}_j}{\partial \mathbf{s}_i \partial \mathbf{s}_j}$ are required to fully estimate $\mathbf{H}_{\mathrm{D}}(\mathbf{s})$.

3.5 Hessian of the Regularization Term

Following the previous section 3.3, finding $\mathbf{H}_{\mathrm{R}}(\mathbf{s}, \boldsymbol{\theta})$ only requires to use the chain-rule for higher dimensions (also known as the Faà di Bruno's formula (Jacobs, 2014)) which gives us (rearranging the terms to avoid summations)

$$\frac{\partial^2 R}{\partial \alpha \partial \beta} = \left(\frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial \alpha}\right)^T \frac{\partial^2 R}{\partial \mathbf{s}_{\mathrm{BM}}^2} \frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial \beta} + \left(\frac{\partial^2 \mathbf{s}_{\mathrm{BM}}}{\partial \alpha \partial \beta}\right)^T \frac{\partial R}{\partial \mathbf{s}_{\mathrm{BM}}} \tag{33}$$

where α, β represent any pair of the parameters of the set $\{\{x_i\}_1^v, \{y_i\}_1^v, a, b, t_x, t_y\}$.

The 2^{nd} partial differential w.r.t. the base mesh (eq. 23) results in

$$\frac{\partial^2 R}{\partial \mathbf{s}_{\mathrm{BM}}^2} = 2\Sigma_{\mathbf{s}}^{-1}. \tag{34}$$

```
Precompute: Shape model ($\mathbf{s}_0$, $\Sigma_s$) and detectors partial gradients $\left\{\frac{\partial \mathbf{h}_i}{\partial x_i}, \frac{\partial \mathbf{h}_i}{\partial y_i}, \frac{\partial^2 \mathbf{h}_i}{\partial x_i^2}, \frac{\partial^2 \mathbf{h}_i}{\partial y_i^2}, \frac{\partial^2 \mathbf{h}_i}{\partial x_i y_i}\right\}_1^v$
Define the combined parameters vector $\mathbf{p} = \left[\mathbf{s}^T | \boldsymbol{\theta}^T\right]^T$
Initial estimate for $\boldsymbol{\theta}^{(0)}$ and $\mathbf{s}^{(0)} = \mathbf{s}_0$ (mean shape).
repeat
    Warp image into the base mesh $\mathbf{I}(.) \rightarrow \mathcal{S}^{-1}(\mathbf{I}(.), \boldsymbol{\theta})$
    for  Landmark $i = 1$ to $v$ do
        Sample local region (at $\mathbf{s}_i$) $\mathcal{I}_i \equiv \mathcal{S}^{-1}(\mathbf{I}(\mathbf{s}_i), \boldsymbol{\theta})$
        Compute data gradient (at base mesh):
        $\nabla_{\mathrm{D}}(\mathbf{s}_i) = \left[\mathcal{I}_i^T \frac{\partial \mathbf{h}_i}{\partial x_i} \; \mathcal{I}_i^T \frac{\partial \mathbf{h}_i}{\partial y_i}\right]$
        Compute data Hessian:
        $\mathbf{H}_{\mathrm{D}}(\mathbf{s}_i) = \begin{bmatrix} \mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial x_i^2} & \mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial x_i y_i} \\ \mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial y_i x_i} & \mathcal{I}_i^T \frac{\partial^2 \mathbf{h}_i}{\partial y_i^2} \end{bmatrix}$
    end
    Compute gradient of the regularization term $\nabla_{\mathrm{R}}$
    Compute Hessian of the regularization term $\mathbf{H}_{\mathrm{R}}$
    $\nabla_f(\mathbf{s}, \boldsymbol{\theta}) = \nabla_{\mathrm{D}}(\mathbf{s}) + \lambda_1 \nabla_{\mathrm{R}}(\mathbf{s}, \boldsymbol{\theta})$ (overall gradient)
    $\mathbf{H}_f(\mathbf{s}, \boldsymbol{\theta}) = \mathbf{H}_{\mathrm{D}}(\mathbf{s}) + \lambda_1 \mathbf{H}_{\mathrm{R}}(\mathbf{s}, \boldsymbol{\theta})$ (overall Hessian)
    Newton step: $\mathbf{p}^{(\tau+1)} \leftarrow \mathbf{p}^{(\tau)} - \gamma \mathbf{H}_f^{-1} \nabla_f(\mathbf{s}, \boldsymbol{\theta})$
until $||\boldsymbol{p}^{(\tau)} - \boldsymbol{p}^{(\tau-1)}|| \leq \varepsilon$ or max. iterations reached ;
```

Algorithm 1: Gradient Shape Model (GSM) 2D fitting algorithm.

Defining $\boldsymbol{\delta}_i$ to be a v-dimensional vector filled with zeros, except with a scalar of 1 over the i^{th} element location, the remaining partial differentials required to fully compute eq. 33 are given by

$$\frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial x_i} = \left[\frac{a\,\boldsymbol{\delta}_i}{b\,\boldsymbol{\delta}_i}\right], \quad \frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial y_i} = \left[\frac{-b\,\boldsymbol{\delta}_i}{a\,\boldsymbol{\delta}_i}\right], \tag{35}$$

$$\frac{\partial \mathbf{s}_{\mathrm{BM}}^2}{\partial x_i \partial a} = \left[\frac{\boldsymbol{\delta}_i}{\mathbf{0}_v}\right], \quad \frac{\partial \mathbf{s}_{\mathrm{BM}}^2}{\partial y_i \partial a} = \left[\frac{\mathbf{0}_v}{\boldsymbol{\delta}_i}\right], \tag{36}$$

$$\frac{\partial \mathbf{s}_{\mathrm{BM}}^2}{\partial x_i \partial b} = \left[\frac{\mathbf{0}_v}{\boldsymbol{\delta}_i}\right], \quad \frac{\partial \mathbf{s}_{\mathrm{BM}}^2}{\partial y_i \partial b} = \left[\frac{-\boldsymbol{\delta}_i}{\mathbf{0}_v}\right], \tag{37}$$

$$\frac{\partial \mathbf{s}_{\mathrm{BM}}^2}{\partial x_i \partial t_x} = \mathbf{0}_{2v}, \qquad \frac{\partial \mathbf{s}_{\mathrm{BM}}^2}{\partial y_i \partial t_y} = \mathbf{0}_{2v}. \tag{38}$$

Like before, the figure 3(b) provides a visual representation for $\mathbf{H}_{\mathrm{R}}(\mathbf{s}, \boldsymbol{\theta})$.

In closure, we point out to the algorithm box 1 that describe step-by-step details of the GSM fitting method. In short, the training stage (very similar to a CLM) only requires to evaluate the mean shape, the full shape covariance (after a Procrutes Analysis) and the 1^{st} and 2^{nd} order partial gradients for each local detector (which needs warping of every training image). Fitting a GSM only requires image warping and a few dot products to estimate both the gradient and the Hessian. All gradient terms are defined in the main paper, except for some details in the Hessian of the regularization term $\mathbf{H}_{\mathrm{R}}$ which appears below in the appendix section.

4 Combined 2D+3D GSM

Face alignment using 3D shape data is always preferable, in the sense that it has the potential to deliver more accurate solutions. The main reason for this lies in the fact that a projection of a 3D model can produce a more feasible realizable shape when compared to a pure 2D model (Xiao et al, 2004a). Moreover, 3D data can be easily retrieved using Non-Rigid Structure from Motion (NRSfM) methods (Xiao et al, 2004b) (Akhter et al, 2008).

Remarkably, fitting with a 3D model just extends the formulation introduced in section 3 by adding two additional constrains

$$f^{3\mathrm{D}} = f(\mathbf{s},\boldsymbol{\theta}) + \lambda_2(\bar{\mathbf{s}}-\bar{\mathbf{s}}_0)^T\Sigma_{\bar{\mathbf{s}}}^{-1}(\bar{\mathbf{s}}-\bar{\mathbf{s}}_0) + \lambda_3\|\mathbf{r}\|^2. \quad (39)$$

A 3D shape is now represented by the $3v$ vector $\bar{\mathbf{s}} = (x_1,\dots,x_v,y_1\dots,y_v,z_1,\dots,z_v)^T$. The first constraint, coupled with the λ_2 weight, is intended to penalize large 3D shape deformations. It acts like before in the 2D case. The projection between the 3D model and the 2D is included in the last constraint, by the $2v$ 'residual' vector $\mathbf{r}$, where λ_3 is set to be a hard constant, linking the two models together

$$\mathbf{r} = \mathbf{s} - \sigma\underbrace{\begin{pmatrix} i_x & i_y & i_z \\ j_x & j_y & j_z \end{pmatrix}}_{\mathbf{R}_o}\otimes\mathbf{I}_v\ \bar{\mathbf{s}} - \begin{pmatrix} o_x \\ o_y \end{pmatrix}\otimes\mathbf{1}_v. \quad (40)$$

In eq. 40, $\mathbf{P} = \sigma\mathbf{R}_o$ is the scaled orthographic projection matrix, (o_x, o_y) are the camera offsets, the symbol $\otimes$ is the Kronecker product and $\mathbf{I}_v$ is an v dimensional identity matrix. The camera pose is updated (using a first order linearization) according to

$$\mathbf{R}_o \leftarrow \mathbf{R}_o\begin{bmatrix} 1 & -\Delta\theta_z & \Delta\theta_y \\ \Delta\theta_z & 1 & -\Delta\theta_x \\ -\Delta\theta_y & \Delta\theta_x & 1 \end{bmatrix} \quad (41)$$

where enforcing orthonormality on $\mathbf{R}_o$ is required.

The extended optimization now operates over the full set of parameters $(\mathbf{s},\boldsymbol{\theta},\bar{\mathbf{s}},\sigma,\Delta\theta_x,\Delta\theta_y,\Delta\theta_z,o_x,o_y)^T$, where the extended Jacobian becomes

$$\mathbf{J}_{f^{3\mathrm{D}}} = \nabla_f(\mathbf{s},\boldsymbol{\theta}) + \lambda_2\nabla_{\mathrm{R3D}}(\bar{\mathbf{s}}) + 2\lambda_3\mathbf{r}^T\nabla\mathbf{r} \quad (42)$$

and the extended Hessian follows the same structure

$$\mathbf{H}_{f^{3\mathrm{D}}} = \mathbf{H}_f(\mathbf{s},\boldsymbol{\theta}) + \lambda_2\mathbf{H}_{\mathrm{R3D}}(\bar{\mathbf{s}}) + 2\lambda_3\nabla\mathbf{r}^T\nabla\mathbf{r}. \quad (43)$$

The gradient of the 3D regularization term is

$$\nabla_{\mathrm{R3D}}(\bar{\mathbf{s}}) = \left[\,\mathbf{0}_{2v+4}\ 2\Sigma_{\bar{\mathbf{s}}}^{-1}(\bar{\mathbf{s}}-\bar{\mathbf{s}}_0)\ \mathbf{0}_6\,\right]^T \quad (44)$$

and the gradient of the 3D to 2D projection

$$\nabla\mathbf{r} = \left[\frac{\partial\mathbf{r}}{\partial\mathbf{s}}\ \frac{\partial\mathbf{r}}{\partial\boldsymbol{\theta}}\ \frac{\partial\mathbf{r}}{\partial\bar{\mathbf{s}}}\ \frac{\partial\mathbf{r}}{\partial\sigma}\ \frac{\partial\mathbf{r}}{\partial\Delta\theta_x}\ \frac{\partial\mathbf{r}}{\partial\Delta\theta_y}\ \frac{\partial\mathbf{r}}{\partial\Delta\theta_z}\ \frac{\partial\mathbf{r}}{\partial o_x}\ \frac{\partial\mathbf{r}}{\partial o_y}\right]^T. \quad (45)$$

```
Precompute: Shape model's ($\mathbf{s}_0$, $\Sigma_s$, $\bar{\mathbf{s}}_0$, $\Sigma_{\bar{\mathbf{s}}}$) and detectors gradients $\left\{\frac{\partial\mathbf{h}_i}{\partial x_i}, \frac{\partial\mathbf{h}_i}{\partial y_i}, \frac{\partial^2\mathbf{h}_i}{\partial x_i^2}, \frac{\partial^2\mathbf{h}_i}{\partial y_i^2}, \frac{\partial^2\mathbf{h}_i}{\partial x_i y_i}\right\}_1^v$
Define the extended parameters vector $\bar{\mathbf{p}} = \left[\,\mathbf{s}^T|\boldsymbol{\theta}^T|\bar{\mathbf{s}}^T|\sigma|\Delta\theta_x|\Delta\theta_y|\Delta\theta_z|o_x|o_y\,\right]^T$
Initial estimate for $\boldsymbol{\theta}^{(0)}$, $\sigma^{(0)}$, $\mathbf{R}_o^{(0)}$, $o_x^{(0)}$ and $o_y^{(0)}$ ($\mathbf{s}^{(0)} = \mathbf{s}_0,\ \bar{\mathbf{s}}^{(0)} = \bar{\mathbf{s}}_0,\ \Delta\theta_x = 0,\ \Delta\theta_y = 0,\ \Delta\theta_z = 0$)
repeat
    Warp image into the base mesh $\mathbf{I}(.) \to \mathcal{S}^{-1}(\mathbf{I}(.),\boldsymbol{\theta})$
    Evaluate 3D to 2D projection error
        $\mathbf{r} = \mathbf{s} - (\sigma\mathbf{R}_o)\otimes\mathbf{I}_v\ \bar{\mathbf{s}} - \begin{pmatrix} o_x \\ o_y \end{pmatrix}\otimes\mathbf{1}_v$
    for Landmark $i = 1$ to $v$ do
        Sample local region (at $\mathbf{s}_i$) $\mathcal{I}_i \equiv \mathcal{S}^{-1}(\mathbf{I}(\mathbf{s}_i),\boldsymbol{\theta})$
        Compute Data gradient (at base mesh):
            $\nabla_{\mathrm{D}}(\mathbf{s}_i) = \left[\mathcal{I}_i^T\frac{\partial\mathbf{h}_i}{\partial x_i}\ \ \mathcal{I}_i^T\frac{\partial\mathbf{h}_i}{\partial y_i}\right]$
        Compute Data Hessian:
            $\mathbf{H}_{\mathrm{D}}(\mathbf{s}_i) = \begin{bmatrix} \mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial x_i^2} & \mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial x_i y_i} \\ \mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial y_i x_i} & \mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial y_i^2} \end{bmatrix}$
    end
    Compute the overall Jacobian: $\mathbf{J}_{f^{3\mathrm{D}}} = \nabla_{\mathrm{D}}(\mathbf{s}) + \lambda_1\nabla_{\mathrm{R}}(\mathbf{s},\boldsymbol{\theta}) + \lambda_2\nabla_{\mathrm{R3D}}(\bar{\mathbf{s}}) + 2\lambda_3\mathbf{r}^T\nabla\mathbf{r}$
    Compute the overall Hessian: $\mathbf{H}_{f^{3\mathrm{D}}} = \mathbf{H}_{\mathrm{D}}(\mathbf{s}) + \lambda_1\mathbf{H}_{\mathrm{R}}(\mathbf{s},\boldsymbol{\theta}) + \lambda_2\mathbf{H}_{\mathrm{R3D}}(\bar{\mathbf{s}}) + 2\lambda_3\nabla\mathbf{r}^T\nabla\mathbf{r}$
    Newton step: $\bar{\mathbf{p}}^{(\tau+1)} \leftarrow \bar{\mathbf{p}}^{(\tau)} - \gamma\mathbf{H}_{f^{3\mathrm{D}}}^{-1}\mathbf{J}_{f^{3\mathrm{D}}}$
    Update camera's rotation matrix
        $\mathbf{R}_o \leftarrow \mathbf{R}_o\begin{bmatrix} 1 & -\Delta\theta_z & \Delta\theta_y \\ \Delta\theta_z & 1 & -\Delta\theta_x \\ -\Delta\theta_y & \Delta\theta_x & 1 \end{bmatrix}$
    Enforce orthonormality constraints
        $\mathbf{R}_o = \mathbf{U}\mathbf{V}^T \leftarrow [\mathbf{U},\mathbf{S},\mathbf{V}] = \mathrm{svd}(\mathbf{R}_o)$
until $||\bar{\mathbf{p}}^{(\tau)} - \bar{\mathbf{p}}^{(\tau-1)}|| \leq \varepsilon$ or max. iterations reached ;
```

Algorithm 2: Gradient Shape Model (GSM) 2D+3D algorithm.

Please refer to the appendix section where the missing partial differentials ($\nabla\mathbf{r}$ and $\mathbf{H}_{\mathrm{R3D}}$) are defined.

The algorithm box 2 provides step-by-step algorithmic details of the GSM 2D+3D approach. In summary, fitting/aligning a GSM requires image warping, sampling each local region once, getting an estimate for the gradient and the Hessian (equivalent to one dot product per landmark), and applying a standard Newton update. The combined 2D+3D GSM additionally requires a measure of the 3D to 2D projection error, their gradients (again simple dot products), and enforcing orthonormality in the camera rotation vectors.

5 Cascade Regression GSM

In general, the cascade regression framework aims to learn a succession of regression matrices $\{\mathbf{R}_k\}_1^K$ that, when applied from a initial parameters estimate $\mathbf{p}_0$, it converges into a optimal state (the training set ground truth), according to

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \mathbf{R}_{k-1}\mathbf{f}(\mathbf{p}_{k-1}) + \boldsymbol{\beta}_{k-1}, \quad k = 1, \ldots, K \tag{46}$$

where, for the sake of notation, the optimization parameters are now redefined to $\mathbf{p} = \left[\mathbf{s}^T \middle| \boldsymbol{\theta}^T\right]^T$ (the previous GSM combined set). Note that this re-formulation breaks down the previous assumption of a linear shape model (the control parameters now follow a nonparametric representation). Still regarding eq. 46, the $\mathbf{f}(\mathbf{p})$ denotes feature extraction ($L \times L \times v$ local regions all concatenated into a single vector) at the landmarks location generated by the shape parameters, $\boldsymbol{\beta}_k$ represents a bias term, the index k represents a cascade level and K is the total number of levels. Note that each regression matrix $\mathbf{R}_k$ relates the extracted features to additive updates (corrections) to be made to the parameters, in a simple sequential chain.

Following the SDM (Xiong and De la Torre, 2014) framework for cascade optimization, learning each regression step can be obtained by minimizing the expected loss between the predicted and the optimal parameters displacement under many possible initializations

$$\arg\min_{\mathbf{R}_k} \sum_{I_l}^{N} \int p(\mathbf{p}_k^l)\left(\Delta\mathbf{p}_k^l + \mathbf{R}_k\mathbf{f}(\mathbf{p}_k^l)\right)^2 \partial\mathbf{p}_k^l \tag{47}$$

where $\Delta\mathbf{p}_k^l = \mathbf{p}_{\mathrm{gt}} - \mathbf{p}_k^l$ is the difference between the disturbed parameter and the ground truth (the regression labels), once more $\mathbf{f}(\mathbf{p}_k^l)$ represents image features extracted at $\mathbf{p}_k$ and N is the number of training images. Note that the bias term ($\boldsymbol{\beta}_k$) was omitted because it can be absolved into an additional column of the regression matrix.

Assuming that $\mathbf{p}_k^l$ is drawn from a Normal distribution (capturing the variance of the initial estimate provided by the face detector), the previous optimization 47 can be approximated by the discrete form

$$\arg\min_{\mathbf{R}_k} \sum_{I_l}^{N} \sum_{j=1}^{M} \left(\Delta\mathbf{p}_k + \mathbf{R}_k\mathbf{f}(\mathbf{p}_k)\right)^2 + \lambda_4\|\mathbf{R}_k\|^2 \tag{48}$$

where M is the number of perturbations / simulations. The previous, in fact, extends the initially proposed L2 regression into ridge regression where λ_4 is the regularization parameter.

The solution of optimization 48 takes the form of

$$\mathbf{R}_k = \left(\mathbf{F}^T\mathbf{F} + \lambda_4\mathbf{I}\right)^{-1}\mathbf{F}^T\Delta\mathbf{p}_k \tag{49}$$

where $\mathbf{F}$ is a data matrix holding all accumulated extracted features (by rows) and $\mathbf{I}$ is an identity matrix with appropriate dimensions.

```
Precompute: Shape model ($\mathbf{s}_0$), detector partial
    gradients $\left\{\frac{\partial\mathbf{h}_i}{\partial x_i}, \frac{\partial\mathbf{h}_i}{\partial y_i}, \frac{\partial^2\mathbf{h}_i}{\partial x_i^2}, \frac{\partial^2\mathbf{h}_i}{\partial y_i^2}, \frac{\partial^2\mathbf{h}_i}{\partial x_i y_i}\right\}_1^v$ and the
    sequence of regression matrices $\{\mathbf{R}_k\}_1^K$
Define the combined parameters vector $\mathbf{p} = \left[\mathbf{s}^T \middle| \boldsymbol{\theta}^T\right]^T$
Initial estimate for $\boldsymbol{\theta}^{(0)}$ and $\mathbf{s}^{(0)} = \mathbf{s}_0$ (mean shape).
for cascade $k = 1$ to $K$ do
    Warp image into the base mesh $\mathbf{I}(.) \rightarrow \mathcal{S}^{-1}(\mathbf{I}(.), \boldsymbol{\theta})$
    for Landmark $i = 1$ to $v$ do
        Sample local region (at $\mathbf{s}_i$) $\mathcal{I}_i \equiv \mathcal{S}^{-1}(\mathbf{I}(\mathbf{s}_i), \boldsymbol{\theta})$
        Compute correlation (centred) $c_i(\mathbf{s}_i) = \mathcal{I}_i^T\mathbf{h}_i$
        Compute data gradient $\nabla_{\mathrm{D}_i}(\mathbf{s}_i)$
        Compute data Hessian $\tilde{\mathbf{H}}_{\mathrm{D}_i}(\mathbf{s}_i)$
    end
    $\mathbf{f} = \left[c(\mathbf{s})\ \nabla_{\mathrm{D}}(\mathbf{s})^T\ \tilde{\mathbf{H}}_{\mathrm{D}}(\mathbf{s})^T\right]^T$
    Cascade update step $\mathbf{p}_{k+1} \leftarrow \mathbf{p}_k + \mathbf{R}_k\mathbf{f}$
end
```

Algorithm 3: Gradient Shape Model - Cascade Regression (GSM-CR) fitting algorithm.

The data matrix is an extremely large data structure as it holds feature vectors taken from every image and every perturbation trial, making it (very) difficult to train with regular desktop machines on large datasets (say with more than a few hundreds of images). Fortunately, we can take advantage of the structure of the GSM local detectors and make use of its efficient way to extract features. Instead of locally computing the features like HoG (Dalal and Triggs, 2005) or SIFT (Lowe, 2004) as in the original SDM formulation, we propose to use

$$\mathbf{f} = \left[\mathcal{I}_1^T\mathbf{h}_1 \ldots \mathcal{I}_v^T\mathbf{h}_v\ \nabla_{\mathrm{D}}(\mathbf{s})^T\ \tilde{\mathbf{H}}_{\mathrm{D}}(\mathbf{s})^T\right]^T \tag{50}$$

which is only a $6v$ dimensional vector (instead of a large concatenation of data given by the vectorized features of the local support regions of all landmarks). This representation holds a measure of alignment score (equivalent to the CLM's data term) combined with its directional gradients and Hessian terms. The initial v elements represent the (centred) correlation at each landmark location $\mathbf{s}_i$, the $\nabla_{\mathrm{D}}(\mathbf{s}_i) = \left(\mathcal{I}_i^T\frac{\partial\mathbf{h}_i}{\partial x_i}, \mathcal{I}_i^T\frac{\partial\mathbf{h}_i}{\partial y_i}\right)$ is the gradient of the data term defined in section 3.2 and, similarly, the $\tilde{\mathbf{H}}_{\mathrm{D}}(\mathbf{s}_i) = \left(\mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial x_i^2}, \mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial y_i^2}, \mathcal{I}_i^T\frac{\partial^2\mathbf{h}_i}{\partial x_i y_i}\right)$ are the Hessian data terms (section 3.4). In the last, we are just considering the vectorized versions of the 'independent' second order gradients.

This modification has a number of advantages, as it allows to drastically increase the density of virtual samples (M) improving on the estimation of each regression matrix $\mathbf{R}_k$ (i.e. eqs 47 and 48 can become more close). Additionally, it does not require a low dimensional reduction step (which usually is the bottleneck in the cascade learning process), hence it becomes faster to train.

```
Initial estimate for all virtual samples $\mathbf{p}_0^{ij}$
    ($i = 1, \dots, N$ images; $j = 1, \dots, M$ virtual samples)
for cascade $k = 1$ to $K$ do
    Estimate noise $r = \mathrm{std}(\mathbf{p}_k^{ij} - \mathbf{p}_k^{gt})$
    for image $i = 1$ to $N$ do
        for perturbation $j = 1$ to $M$ do
            Add noise to virtual sample
            $\mathbf{p}_k^{ij} = \mathbf{p}_k^{ij} + \nu, \quad \nu \sim \mathcal{N}(\mathbf{0}, r)$
            Shape deviation from ground truth
            $\Delta\mathbf{p}_k = \mathbf{p}_k^{gt} - \mathbf{p}_k^{ij}$
            Warp image into the base mesh
            $\mathbf{I}(.) \rightarrow \mathcal{S}^{-1}(\mathbf{I}(.), \mathbf{p}_k^{ij})$
            for Landmark $l = 1$ to $v$ do
                Sample local region $\mathcal{I}_l \equiv \mathcal{S}^{-1}(\mathbf{I}_i, \mathbf{p}_k^{ij})$
                Compute correlation $c_l(\mathbf{p}_k^{ij}) = \mathcal{I}_l^T \mathbf{h}_i$
                Compute data gradient $\nabla_{\mathrm{D}_l}(\mathbf{p}_k^{ij})$
                Compute data Hessian $\tilde{\mathbf{H}}_{\mathrm{D}_l}(\mathbf{p}_k^{ij})$
            end
            $\mathbf{f} = \left[\, c(\mathbf{p}_k^{ij}) \;\; \nabla_{\mathrm{D}}(\mathbf{p}_k^{ij})^T \;\; \tilde{\mathbf{H}}_{\mathrm{D}}(\mathbf{p}_k^{ij})^T \,\right]^T$
            Hold feature data $\mathbf{F}^{ij} = \mathbf{f}$
        end
        Regression $\mathbf{R}_k = \left(\mathbf{F}^T\mathbf{F} + \lambda_4 \mathbf{I}\right)^{-1} \mathbf{F}^T \Delta\mathbf{p}_k$
    end
    Cascade parameters update $\mathbf{p}_{k+1} \leftarrow \mathbf{p}_k + \mathbf{R}_k \mathbf{F}$
end
```

Algorithm 4: Learning a Gradient Shape Model - Cascade Regression (GSM-CR).

In summary, the algorithm 3 highlights the step-by-step fitting procedure of the GSM-CR approach. Fitting such a model simply consists of reusing eq. 46, by evaluating the GSM gradient features at a given location and obtaining its updates by regression.

Regarding the learning of a GSM-CR instance, it follows a standard procedure of training a sequential cascade with just some modifications at the feature extraction. The entire procedure is detailed in the algorithm box 4.

In the training stage the perturbation variables (described as 'virtual samples') are represented as $\mathbf{p}_k^{ij}$ where the k subscript refers to the cascade level and the superscripts refer to the j^{th} perturbation w.r.t. the i^{th} image. Note that there are M virtual samples for each one of the N training images. Initially a face detector is used to roughly estimate the 2D pose parameters (assigning starting values to every virtual sample). Afterwards the main cascade level loops between: computing the parameters deviation from the ground truth; disturb all virtual samples; evaluate the modified feature vector (eq. 50); find the current regression matrix $\mathbf{R}_k$ (eq. 49) and finally, update the virtual samples to the next level.

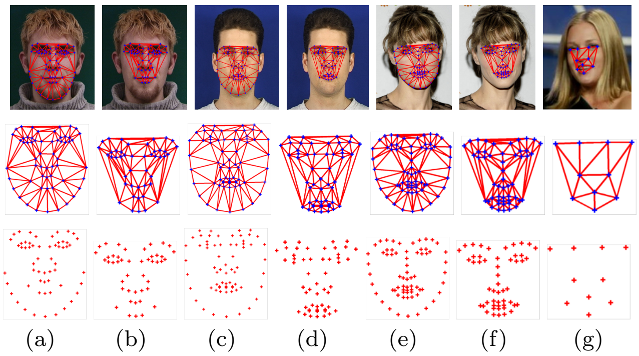


Fig. 4 Facial landmark's configuration in each dataset. Each column displays a set of landmarks: (a) IMM with the original set of $v = 58$ landmarks, (b) IMM with $v = 45$ (without face out boundary), (c) XM2VTS $v = 68$, (d) XM2VTS $v = 53$, (e) LFPW $v = 68$, (f) LFPW $v = 51$ and (g) LFW $v = 10$. Both HELEN and 300W follow the same format than LFPW.

6 Evaluation Results

6.1 Datasets

The performance evaluation experiments was conducted on several standard databases:

1. The IMM (Nordstrom et al, 2004) database consists of 240 annotated images of 40 different human faces presenting different head poses, illumination, and facial expression (58 landmarks).
2. The XM2VTS (Messer et al, 1999) database has 2360 images of frontal faces taken from 295 subjects (68 landmarks). The XM2VTS mainly focuses on variations in identity, nevertheless, it exhibits large diversity in appearance due to facial hair, glasses, ethnicity and other subtle changes.
3. The Labeled Faces in the Wild (LFW) (Huang et al, 2007) database has more than 13000 images (10 landmarks) taken 'in the wild'. Meaning that the images were taken in unconstrained scenarios under pose, lighting, focus, facial expression and occlusions changes.
4. The LFPW (Belhumeur et al, 2011) database, also an 'in the wild set', contains 811 (train) and 224 (test) images collected over web searches (google, flickr and yahoo). The original release has 29 landmarks, but we are using the 68 landmarks annotation provided by (Sagonas et al, 2013a).
5. The HELEN (Le et al, 2012) database holds 2000 (train) plus 330 (test) images taken from the flickr web site. Like before, the initial release provides 194 landmarks but for consistency we are using the same 68 landmarks format than LFPW.
6. The 300W (Sagonas et al, 2016) (Sagonas et al, 2013a) (Sagonas et al, 2013b) supplies two subsets,

each holding 300 images taken in indoor and outdoor scenarios, respectively. In our experiments we combined both sets and established an overall test set with 600 images. The training set of this database is defined as standard: a combination of the AFW (Zhu and Ramanan, 2012) database (337 images), the HELEN, the iBug (Sagonas et al, 2013a) (135 images), the LFPW and the XM2VTS set, making a total of 6197 images (68 landmarks).

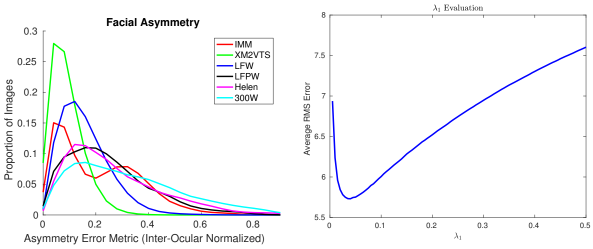
An overview between the different landmark's annotation formats can be seen in figure 4. In this section, each dataset was additionally (and independently) evaluated with a slight less number of landmarks, by removing the boundary points over the face outer contour. This was made because of the increasing difficulty in localizing, with accuracy, those landmark points. Still regarding landmark formats, it is important to mention that the LFPW, the HELEN and the 300W datasets all share the same configuration. Credits to (Sagonas et al, 2016) (Sagonas et al, 2013a).

Before presenting the evaluation results, it is worth comparing the relative complexity in aligning each database Following (Belhumeur et al, 2011), a facial asymmetry derived metric was used to measure the degree of difficulty of each image. Such asymmetry metric consists of reflecting natural symmetric features (such as the eyes out corners and mouth corners) about a vertical line passing the nose centre and then measure the (normalized) average distances between them. According, this metric holds a lower value (close to zero) in near frontal faces and higher values otherwise.

The figure 5(a) shows the described facial asymmetry metric gathered over the evaluated sets. It can be seen that XM2VTS holds more symmetric images (more frontal), and by other hand, the remaining 'in the wild' sets (LFPW, HELEN and 300W) have indeed more challenging images with a lot more 3D pose variability, therefore more difficult to align (more asymmetric images scattered across the full set).

6.2 Evaluation Procedure

Due to the intrinsic nature of the proposed models, the main experiments was split in two parts: (1) evaluate the GSM against CLM like approaches and (2) evaluate the GSM-CR against cascaded regression based methods. Later on, for the sake of completeness, our cascaded regression approach is compared against Convolutional Neural Networks (CNNs) based methods.



(a) Face asymmetry (b) λ_1 evaluation

Fig. 5 (a) Face asymmetry in the datasets. (b) Evaluation of the regularization parameter λ_1.

6.2.1 GSM vs CLMs

This section puts to the test the proposed GSM and GSM 2D+3D variants against CLM alignment solutions, in particular, ASM (Cootes et al, 1995), CQF (Wang et al, 2008), SCMS (Saragih et al, 2010) and BCLM (Martins et al, 2012). It is important to stress that we made a careful evaluation ensuring that all evaluated techniques use the same kind of local detector, all methods are regularized by the same shape model and they always start from the same initial estimate. Note that, to be fair, the GSM should be pitted against comparable CLM baselines. According, the cascade regression methods should not be directly compared in this section, as they operate using a multi-stage framework and capture the variance of the face detector.

Most of the datasets described in the previous section, already define the train and the test data (LFPW, HELEN and 300W). Regarding the remaining sets: the XM2VTS and LFW were split into 70/30 train/test portions and the IMM (the smallest set) was trained with 511 images collected from 40 individuals at our institution (following the same landmark annotation format with 58 points).

In all cases, the nonrigid parameters started from the mean shape ($\mathbf{s} = \mathbf{s}_0$ in GSM and $\mathbf{b}_s = \mathbf{0}_n$ in CLMs), the pose parameters $\boldsymbol{\theta}$ were initialized by regression from a face detector (Viola and Jones, 2002) output and all models were fitted until convergence up to a maximum of 30 iterations. The shape initialization was also included in the evaluation charts.

The local detectors, both in GSM and CLMs, were built from grey level features with the $L \times L$ support region size of (46×46). The same standard deviation $\sigma_h = 5$ and regularization $\epsilon = 10^{-4}$ were used. The GSM results with MCCF detectors built from HoG features have $L = 46$, cell size $= 3$ and $\sigma_h = 1.5$. Additionally, the CLM methods that rely in Kernel Density Estimation (KDE) approximations of the data term use a kernel bandwidth schedule of $\sigma_h^2 = (15, 10, 5, 2)$ (which applies to SCMS, BCLM).

In GSM, the overall regularization weights λ_1 was subject to evaluation to find the best value. The figure 5(b) shows the average Root Mean Squared (RMS) error in a randomly selected test set (1000/250 of train/test images taken from both LFPW and HELEN databases) for several values of λ_1. The minimal value of $\lambda_1 = 0.04$ was found and used afterwards. In the GSM 2D+3D method, λ_2 is set to be equal to λ_1 and $\lambda_3 = 10$ (established to be a hard constraint). The 3D data, used in GSM 2D+3D, was retrieved by applying NRSfM (Akhter et al, 2008) in the training shape annotations of each corresponding dataset.

The standard evaluation procedure quantifies the alignment error by the mean error per landmark as fraction of the inter-ocular distance (measured between the outer corners of the eyes), d_{eyes}, as

$$e_m(\mathbf{s}) = \frac{1}{v\ d_{\text{eyes}}} \sum_{i=1}^{v} \|\mathbf{s}_i - \mathbf{s}_i^{\text{gt}}\| \tag{51}$$

where $\mathbf{s}_i^{\text{gt}}$ is the location of i^{th} landmark in the ground truth annotation.

Figure 6 shows the fitting performance curves, using the normalized inter-ocular error metric for the IMM, XM2VTS, LFPW, LFW, HELEN and 300W datasets, respectively. These curves are cumulative distribution functions that show the percentage of faces that achieved convergence with a given error amount. The table bellow in the same figure provides a qualitative measure of the results, which is defined as the ratio, in percentage, between the area under fitting curve and the total area of a ground truth curve (a step curve). Likewise, the figure 7 shows another set of fitting curves taken by re-training all models[2], now with a reduced set of landmarks (discarding the outer contour points of the face, recall the figure 4).

The results show, firstly, that the relative performance between CLMs methods behaves as expected. Ranking from lower to higher accuracy we get: ASM, CQF, SCMS and BCLM. In some cases the ASM performs better than CQF (in the LFW, HELEN and 300W), the reason being the excellent overall performance of the MCCF filter. The CQF has a tendency to oversmooth the response maps, SCMS outperforms the previous mostly because of the high accuracy provided by the mean-shift algorithm and finally, BCLM improves on the results of SCMS due to the enhanced parameter update. Regarding GSM, the experiments show that the technique has a comparable performance to BCLM and, in some cases, even performs better (LFW & XM2VTS sets). This happens because GSM optimize the landmarks locations directly using a more capable shape model (a full Gaussian with a proper regularization instead of a low dimensional representation).

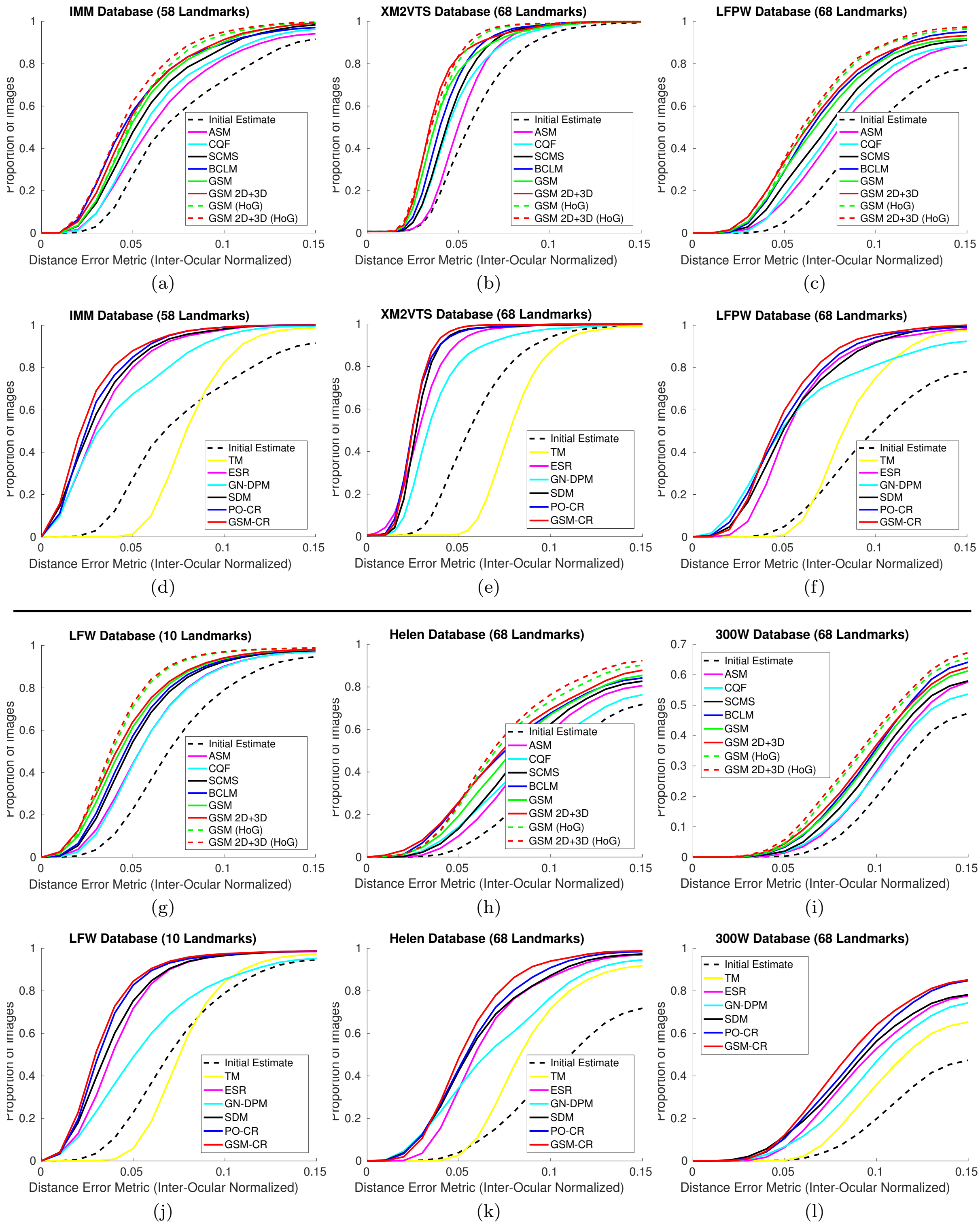
We remark that CLMs rely in a exhaustive local search while GSM use a gradient based search. In theory, exhaustive search would be better but much more computational expensive. In practice, however, the results do not show a large disparity. In fact, in some cases, gradient search achieved better results (although with an improved regularization). Finally, and as expected, the enhanced GSM 2D+3D method improves over the 2D variant, and the HoG based detectors further improve on the overall performances.

6.2.2 GSM-CR vs Cascade Regression

This section evaluates the GSM-CR against other popular cascade regression based methodologies. In practice, we make comparisons against the Supervised Descent Method (SDM) (Xiong and De la Torre, 2013), the Project-Out Cascaded Regression (PO-CR) (Tzimiropoulos, 2015) and the Explicit Shape Regression (ESR) (Cao et al, 2012). Additionally we also include two other approaches (not cascade regression based), a simplified version of the Gauss-Newton Deformable Parts Model (GN-DPM) (Tzimiropoulos and Pantic, 2014) that is optimized using the Project-Out Inverse Compositional (POIC) (Baker and Matthews, 2001) strategy and the Tree-Model (TM) (Zhu and Ramanan, 2012). Like before, the all mentioned methods use our own implementations (except for the ESR (Guo, 2014) and TM (Zhu and Ramanan, 2012) cases).

The same local patch settings were used as in the previous section (HoG features, $L = 46$, cell size = 3, $\sigma_h = 1.5$). The number of cascade levels was established to be equal to $K = 5$. The regression procedure has some computational memory concerns regarding the number of perturbations per image (M). Since the data matrix $\mathbf{F}$ (in eq. 49) needs to collect all feature samples, it can be intractable to manage for a large number of perturbations, specially if the dataset is to large (or have to many landmarks). In our experiments we were able to learn SDM and PO-CR models for the LFPW, XM2VTS and IMM databases using $M = 20$. In the case of the LFW $M = 10$, and for HELEN and 300W $M = 5$. These selected values are related to the maximum available memory in our test machine (note that the required memory grows linearly by a factor MN, where N is the number of training images). In contrast, our proposed GSM-CR allowed to estimate

[2] The LFW dataset was excluded due the lack of landmark annotations in the face outer region.



(a) (b) (c)

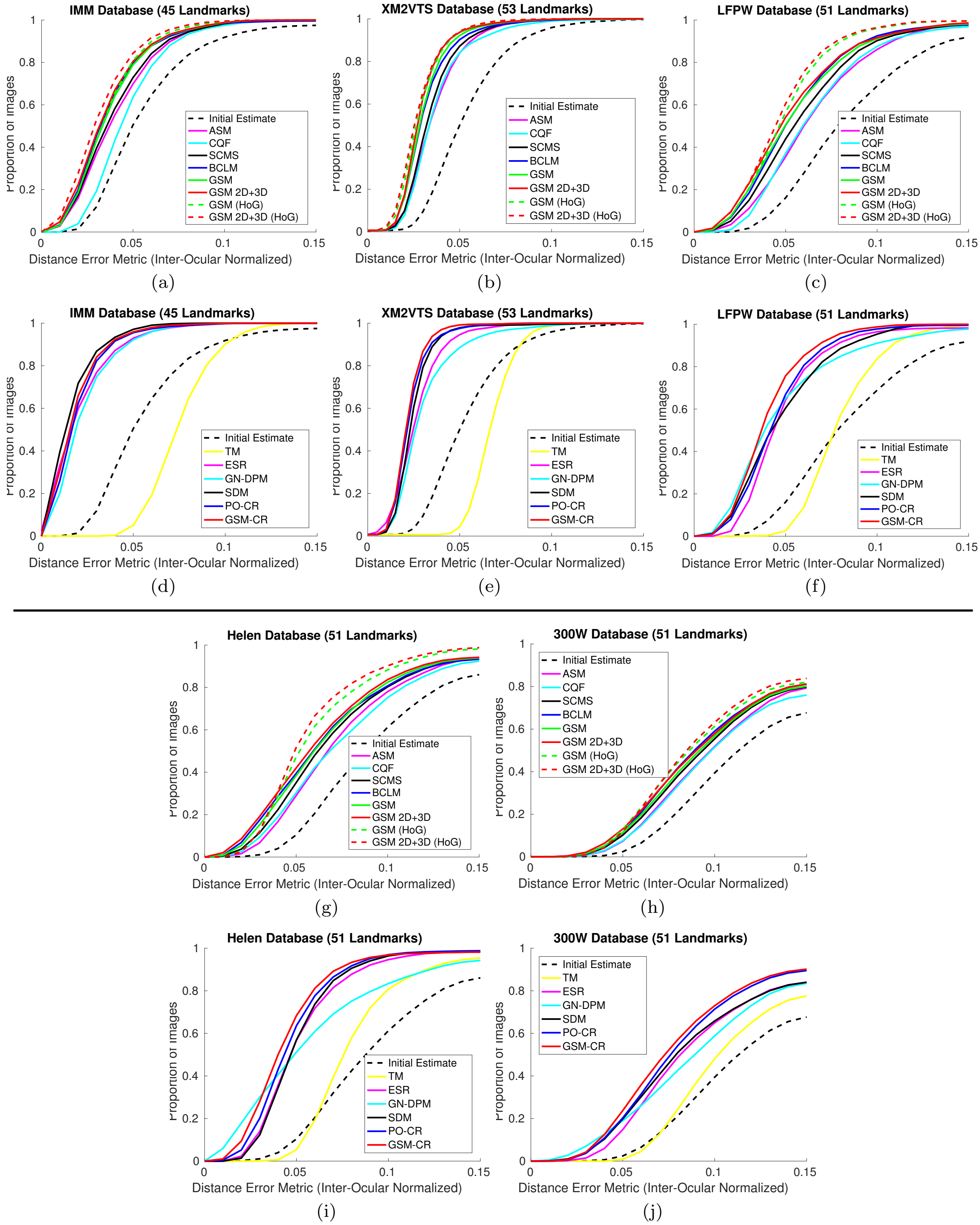
(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(%) Area under cdf curve / total area	IMM	XM2VTS	LFPW	LFW	HELEN	300W
Initial Estimate (Viola and Jones, 2002)	48.0	60.2	31.8	49.7	26.9	14.5
ASM (Cootes et al, 1995)	54.8	65.2	42.4	60.4	36.2	19.3
CQF (Wang et al, 2008)	57.0	67.4	44.5	60.0	35.0	18.7
SCMS (Saragih et al, 2010)	60.5	69.1	47.5	64.2	39.1	21.0
BCLM (Martins et al, 2016)	64.2	71.3	52.3	65.4	44.7	23.8
GSM (our method) (Grey / HoG features)	62.7/64.6	71.9/73.6	51.0/55.1	67.3/71.2	42.7/47.6	22.9/26.2
GSM 2D+3D (our method) (Grey / HoG features)	64.2/67.5	74.1/74.9	53.6/56.5	68.5/72.0	46.2/49.2	24.1/27.2
TM (Zhu and Ramanan, 2012) (p146)	45.1	46.5	42.2	47.1	40.5	22.8
ESR (Cao et al, 2012)	77.3	79.9	61.4	71.5	56.1	32.8
GN-DPM (Tzimiropoulos and Pantic, 2014) (POIC)	72.5	74.4	59.5	60.7	52.5	29.8
SDM (Xiong and De la Torre, 2013)	79.2	81.0	63.0	73.3	59.0	35.2
PO-CR (Tzimiropoulos, 2015)	80.4	82.2	65.4	75.7	60.7	37.5
GSM-CR (our method - HoG features)	82.1	82.6	66.6	76.8	62.9	39.5

Fig. 6 Fitting performance curves in IMM, XM2VTS, LFPW, LFW, HELEN and 300W sets. The evaluation of 'GSM vs CLMs' and 'GSM-CR vs cascade regression methods' appears separately (two charts for each database - one on top of another). The table bellow gives a measure of the ratio between the area under the curve and the total area (in percentage).



(a) (b) (c)

(d) (e) (f)

(g) (h)

(i) (j)

(%) Area under cdf curve / total area	IMM	XM2VTS	LFPW	HELEN	300W
Initial Estimate (Viola and Jones, 2002)	62.1	64.0	43.8	38.5	25.2
ASM (Cootes et al, 1995)	72.8	75.8	56.8	50.8	33.2
CQF (Wang et al, 2008)	68.7	75.1	56.6	50.0	32.5
SCMS (Saragih et al, 2010)	73.6	76.8	60.2	53.8	35.4
BCLM (Martins et al, 2016)	75.5	78.7	63.2	55.8	37.1
GSM (our method) (Grey / HoG features)	75.3/76.0	79.4/80.8	63.0/67.0	55.7/60.2	36.2/38.6
GSM 2D+3D (our method) (Grey / HoG features)	76.3/78.4	80.3/81.4	64.4/68.4	57.6/62.1	37.3/39.3
TM (Zhu and Ramanan, 2012) (p146)	49.7	55.0	46.6	45.8	29.1
ESR (Cao et al, 2012)	86.2	82.2	67.6	65.0	40.7
GN-DPM (Tzimiropoulos and Pantic, 2014) (POIC)	84.1	79.5	67.7	61.5	39.8
SDM (Xiong and De la Torre, 2013)	85.9	83.5	68.2	65.8	42.3
PO-CR (Tzimiropoulos, 2015)	86.8	84.6	70.0	68.2	45.2
GSM-CR (our method - HoG features)	88.1	85.4	73.1	70.2	46.9

Fig. 7 Fitting performance curves in IMM, XM2VTS, LFPW, HELEN and 300W sets having a reduced set of landmarks (removing the face contour boundary - see figure 4). Like before, separate charts are presented for GSM / GSM-CR evaluations, and the table bellow gives a measure of the area under the curve ratio.

each regression step using $M = 40$ in all datasets (since it uses a considerable lower dimensional feature vector).

The fitting performance curves for these experiments are shown in figures 6 and 7, where the later refers to the sets with reduced landmarks. Looking at the results, and once more ranking from lower to higher performance we get: TM, GN-DPM, ESR, SDM, PO-CR and GSM-CR. As pointed out, the first two, are not really cascade regressing methods. The TM was mainly proposed as a detector and its lower accuracy results from the simple regularization used (made for fast inference). The GN-DPM adds stronger regularization (a full shape model), and in some sense, can be seen as the PO-CR without the cascade framework. Regarding the overall fitting performance on cascade based methods, the SDM performed slightly better than ESR, the PO-CR better that SDM (as expected due to underlying shape structure included in the regression), and finally GSM-CR was able to outperform all the previous (due to the denser regression perturbations).

In closure, figure 9 shows some qualitative examples of GSM-CR fitting taken from the LFPW, HELEN and 300W databases.

6.2.3 GSM-CR vs CNNs

For reference, this section presents a comparison between our best ranking model (naturally, the cascaded regression version) and some Convolutional Neural Networks (CNNs) based techniques. The proposed method GSM-CR is here evaluated against the Tasks-Constrained Deep Convolutional Network (TCDCN) (Zhang et al, 2014b), the 2D version of the Face Alignment Network (FAN) (Bulat and Tzimiropoulos, 2017b), the binarized Hourglass-like convolutional network (referred here as BFAN) (Bulat and Tzimiropoulos, 2017a), the Convolutional Experts Constrained Local Model (CE-CLM) (Zadeh et al, 2017) and finally, the commercial solution Face++ (Face++, 2018) (Huang et al, 2015).

Following the previous sections, the fitting performance curves of these experiments in the IMM, XM2VTS, LFPW, LFW, HELEN and 300W datasets are presented in figure 8. Like before, all these curves measure the inter-ocular normalized error as expressed in eq. 51.

Briefly, the results show that the approaches Face++ and FAN take the lead in terms of fitting accuracy, followed by CE-CLM and BFAN. Our technique (GSM-CR) comes next, performing slightly better than TCDCN. In general, and as expected, our method can only be comparable with CNNs based approaches in smaller datasets (p.e. in the LFPW and the IMM). On larger sets, with more 'in the wild' images (recall figure 5(a)), our model has more difficulty to keep up (note that the results in the LFW set might be a bit misleading due to the landmarks configuration that match strong image features - check figure 4(g)). However, please keep in mind that, in fairness, our approach should not be directly compared against CNNs. The CNNs are intrinsically different, they optimize millions of parameters, include non-linear mappings between layers, require large amounts of training data and take a huge amount of time to train (usually assisted by GPUs). Our approach shares many of the advantages of CLMs, its notably simpler than CNNs, requires far less memory, it trains in minutes and performs model fitting in milliseconds (with regular CPU).

Table 1 Computational efficiency comparison (GSM vs CLMs).

	SCMS	BCLM	GSM	GSM 2D+3D
Generate shape	$O(v\,n)$	$O(v\,n)$	$O(v)$	$O(v)$
Warp image	$O(m)$	$O(m)$	$O(m)$	$O(m)$
Compute data term	$O(vL^2R^2)$	$O(vL^2R^2)$	$O(vL^2)$	$O(vL^2)$
Local optimization	$O(vR^4T)$	$O(vR^4T)$	—	—
Evaluate Jacobian	$O(v\,n)$	—	$O(vL^2)$	$O(vL^2+v)$
Compute Hessian	$O(n^2v)$	—	$O(v^2L^2)$	$O(v^2L^2+v)$
Invert the Hessian	$O(n^3)$	—	$O(v^3)$	$O(v^3)$
Update parameters	$O(n)$	$O(n^3+v^2)$	$O(v)$	$O(v)$
Running time (s)	10.454	10.943	**0.282**	**0.303**

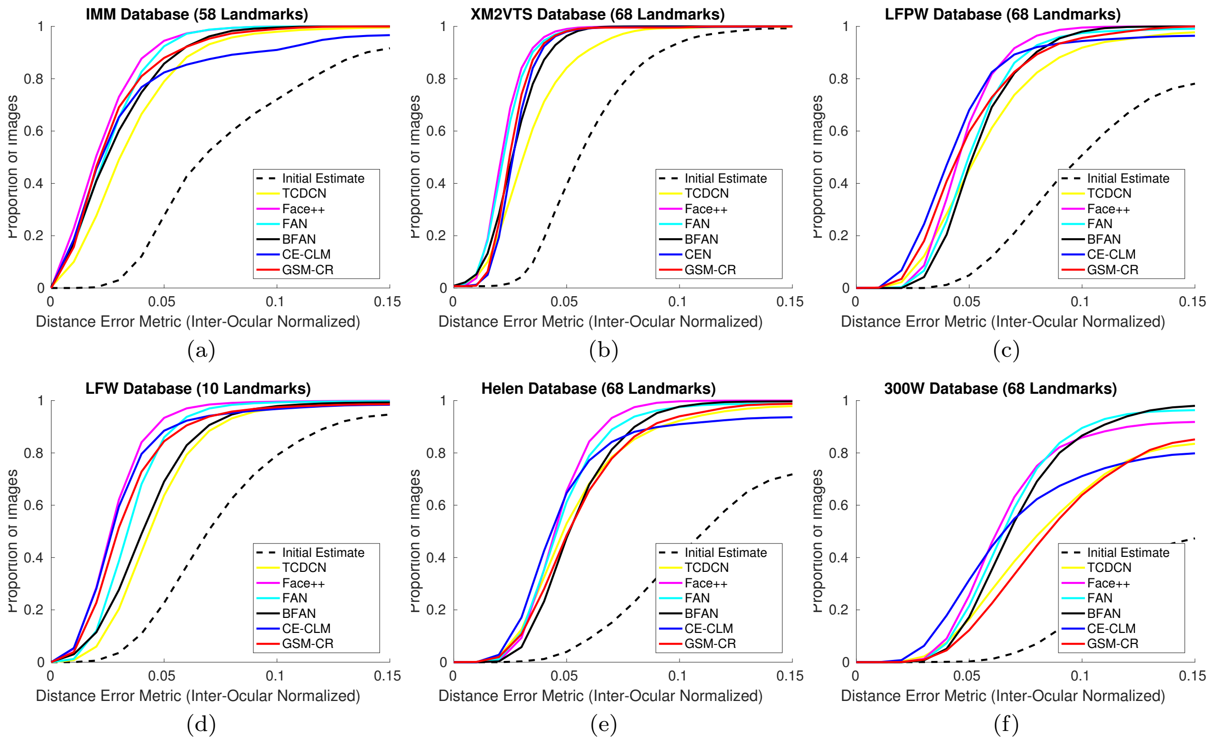
6.3 Computational Performance

This section presents a computational performance analysis between GSM vs CLMs and also GSM-CR vs cascade regression based methods. As pointed out, the GSM and CLMs have very similar learning steps, hence similar learning times. The only difference comes down to the 'amount' of local detectors needed for each landmark, where the GSM requires five detector gradients in total. Regarding the GSM-CR and the mentioned cascade regression methods, they have similar fitting costs (in general, by simply implementing the eq. 46).

The central discussion here, reduces to the computational analysis of the GSM fitting stage and the GSM-CR learning stage. Afterwards, it is shown an execution time comparison (i.e. online fitting) between all the mentioned methods in the evaluation section.

6.3.1 Fitting - GSM vs CLMs

Table 1 shows, in detail, a comparative view of the computational cost of one iteration between the proposed GSM and some state-of-the-art CLM strategies. The mentioned parameters are v landmarks, n CLM parameters, m pixels in the base mesh, $L \times L$ local detectors size, $R \times R$ CLM scan regions (convolution area) and T is maximum number of mean-shift iterations (which is related to the Kernel Density Estimator in the SCMS and BCLM methods).



(%) Area under cdf curve / total area	IMM	XM2VTS	LFPW	LFW	HELEN	300W
Initial Estimate (Viola and Jones, 2002)	48.0	60.2	31.8	49.7	26.9	14.5
TCDCN (Zhang et al, 2014b) (Zhang et al, 2016)	76.7	77.4	61.1	68.7	63.2	41.1
Face++ (Face++, 2018) (Huang et al, 2015)	84.9	85.2	68.2	81.1	69.0	53.0
FAN (Bulat and Tzimiropoulos, 2017b)	82.9	84.6	64.8	76.1	67.2	53.6
BFAN (Bulat and Tzimiropoulos, 2017a)	81.0	82.1	63.7	71.2	63.7	51.7
CE-CLM (Zadeh et al, 2017)	77.3	81.8	68.6	78.5	65.0	46.9
GSM-CR (our method w/ HoG features)	82.1	82.6	66.6	76.8	62.9	39.5

Fig. 8 Evaluation against CNNs based methods. Following the previous structure, the presented graphics show the fitting performance curves of IMM, XM2VTS, LFPW, LFW, HELEN and 300W datasets, respectively. The table below provides a measure of the area under the curve (in percentage).

The reported running times use a non-parallel MatLab implementation with grey level features, $v = 68$, $n = 24$, $m = 44440$, $L = 41$, $R = 15$ and $T = 75$. Notice the huge performance advantage of the GSM, being up to 38.8× faster than BCLM. As pointed out before, the key to GSM's efficiency is the very fast evaluation of the data term, avoiding the need of response maps.

6.3.2 Learning - GSM-CR vs Cascaded Regression

As described, the GSM-CR learning/training times is an issue that deserves some attention. When comparing the GSM-CR against other cascade regression approaches, a large gap in computational costs and memory requirements exists.

The table 2 shows a comparative view of the computational cost of one cascade step between the proposed GSM-CR and two well established state-of-the-art approaches (SDM and PO-CR). The undefined parameters c and D represent the numbers of appearance parameters (PO-CR) and the effective dimensions after applying the low dimensional reduction (SDM), respectively. The amount of regression perturbations is once more represented as M.

From the algorithmic point of view (taking SDM as an example), each cascade level requires to generate shape perturbations, extracting and gathering features in a data matrix, a dimensional reduction step and finally solving a regression problem. This means dealing with data matrices ($\mathbf{F}$ in eq. 49) of size $L^2 v \times NM$, making it very difficult to use regular PCs while learning with thousands of images. The same can be said for PO-CR that follows a similar procedure (stacking all features in a very large matrix, then estimating a Jacobian by regression, removing the appearance effects with a Project-Out (Baker and Matthews, 2001) step and estimate an 'overall' regression matrix). On the other hand, training a GSM-CR just involves to manage data matrices with size $6v \times NM$, requiring considerable less memory.

Table 2 Computational efficiency comparison in learning a single cascade level (GSM-CR vs cascade based methods).

	SDM	PO-CR	GSM-CR
Compute std error	$O(Nv)$	$O(Nn)$	$O(Nv)$
Feature extraction	$O(MNvL^2)$	$O(MNvL^2)$	$O(MNvL^2)$
Dim. reduction	$O((MvL^2)^2N+(MvL^2)^3)$	-	-
Project-out app.	-	$O(L^2vn+cL^2vn)$	-
Regression	$O(D^3)$	$O((MNvL^2)^3)$	$O((MNv)^3)$
Update step	$O(NM)$	$O(NM)$	$O(NM)$
Running time (s)	125.8	61.4	**48.6**

The reported times refer to a test run on a MatLab implementation with just $N = 100$ images (taken from the LFPW set), $v = 68$ landmarks, $M = 20$ perturbations, $L = 46$ patch size, $n = 27$ shape parameters (PO-CR), $c = 417$ appearance parameters (PO-CR) and $D = 2000$ reduced dimensions (SDM). According, and as described earlier, the GSM-CR is faster to train (2.58× faster than SDM) and requires considerable less computational resources.

6.3.3 Execution Times

The table 3 shows a comparative view of the execution times of the techniques mentioned in the evaluation section. Each table entry shows the average running time of fitting one image, in a representative test set, for a given algorithm (which in turn, was implemented in a particular code environment). The table is organized by similar classes of methods, i.e. CLMs, cascaded regression and CNNs. Once more we highlight that both GSM and GSM 2D+3D should be compared against CLMs, and similarity, GSM-CR should be compared against other cascade regression techniques. The CNNs timing results are just shown for reference.

In this experiment, each fitting algorithm uses the same structural settings described in section 6.2. The test set was chosen to be a subset of 100 images taken from the LFPW test database. The hardware involved was a PC holding a Intel Core i7-3930K (3.20GHz, 6 cores) CPU, 32GB RAM, 2 Nvidia graphics cards (GeForce GTX Titan X and Tesla K40c) and running Linux Fedora 25 OS. Most of the evaluated fitting algorithms are based in implementations made by us (as indicated in the table). Please note that, these implementations are MatLab based, unoptimized, non-parallel and use only CPU hardware. The remaining techniques, except ESR (Guo, 2014), consist of the corresponding author's supplied code.

The results show that both GSM and GSM-CR, as expected, perform the fastest in their respective categories, i.e. the GSM is several times faster than any CLM and the GSM-CR is marginally faster than the cascaded regression techniques. Regarding the CNNs timing results, note that the comparative performance might be a bit misleading because some methods rely heavy in both the GPUs that we have installed. Nevertheless, the GSM-CR can run faster than TCDCN (implemented in similar code environment) and the FAN.

Table 3 Comparative view of the execution times of all the evaluated techniques: CLMs, cascaded regression, our proposed GSMs and CNNs. Each table entry shows the average execution time (in a particular code environment) of fitting an image in a subset of the LFPW database.

	Code Environment	Running Time (s)	Notes
ASM	MatLab*	9.86	Grey
CQF	MatLab*	11.32	Grey
SCMS	MatLab*	77.822	Grey
BCLM	MatLab*	79.226	Grey
GSM	MatLab*	1.981/2.919	Grey/HoG
GSM 2D+3D	MatLab*	2.227/3.145	Grey/HoG
TM	MatLab	6.414	HoG
ESR	MatLab	17.052	Fern
GN-DPN	MatLab*	1.801	HoG
SDM	MatLab*	0.411	HoG
PO-CR	MatLab*	0.348	HoG
GSM-CR	MatLab*	0.295	HoG
TCDCN	MatLab	3.675	
Face++	Megvii Servers	0.195	
FAN	Python	9.530/3.489	CPU/GPU
BFAN	Lua	0.0537	GPU
CE-CLM	C++	0.159	

(*) Our implementation.

7 Conclusions

This paper introduces the Gradient Shape Model (GSM) that aims to replace exhaustive local searches (convolutions) with a fast gradient estimate in the CLM formulation. The proposed approach considers two other significant extensions: a 2D+3D combined model and a cascade regression strategy.

In its basic form, the GSM is able to align/fit a shape model by sampling each local region only once, estimating a gradient direction (and Hessian terms) using a true analytical Newton update. The combined 2D + 3D GSM enhances the previous, by including a additional 3D shape model to 2D projection constraint. The cascade regression GSM approach, benefits of the efficient feature data extraction, enforcing the sampling density in the estimation of each each regression matrix.

All proposed techniques are evaluated in detail on several standard datasets (IMM, XM2VTS, LFPW, LFW, HELEN and 300W) and compared against state-of-the-art CLM and CR methods. Several results are presented: (1) the proposed models match the performance of leading CLM methods, while using only a fraction of the computation; (2) the usage of a 3D constrained search improves on the previous model; (3) the multidimensional detectors further improve the accuracy and (4) the cascade regression variant allow faster training times (when compared with other cascade regression methods) and it exhibits the best overall performance.

Acknowledgements This work was supported by the Portuguese Science Foundation (FCT) through the grant SFRH/BPD/90200/2012 (P. Martins).

References

Akhter I, Sheikh Y, Khan S, Kanade T (2008) Nonrigid structure from motion in trajectory space. In: Neural Information Processing Systems

Alabort-i-Medina J, Zafeiriou S (2017) A unified framework for compositional fitting of active appearance models. International Journal of Computer Vision 121(1):26–64

Asthana A, Zafeiriou S, Cheng S, Pantic M (2013) Robust discriminative response map fitting with constrained local models. In: IEEE Conference on Computer Vision and Pattern Recognition

Baker S, Matthews I (2001) Equivalence and efficiency of image alignment algoritms. In: IEEE Conference on Computer Vision and Pattern Recognition

Baltrušaitis T, Robinson P, Morency L (2013) Constrained local neural fields for robust facial landmark detection in the wild. In: IEEE International Conference on Computer Vision Workshop, 300 Faces In-the-Wild Challenge (300-W)

Belhumeur PN, Jacobs DW, Kriegman DJ, Kumar N (2011) Localizing parts of faces using a consensus of exemplars. In: IEEE Conference on Computer Vision and Pattern Recognition

Boddeti VN, Kanade T, Kumar BVKV (2013) Correlation filters for object alignment. In: IEEE Conference on Computer Vision and Pattern Recognition

Bolme DS, Beveridge JR, Draper BA, Lui YM (2010) Visual object tracking using adaptive correlation filters. In: IEEE Conference on Computer Vision and Pattern Recognition

Bulat A, Tzimiropoulos G (2017a) Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources. In: IEEE International Conference on Computer Vision

Bulat A, Tzimiropoulos G (2017b) How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks). In: IEEE International Conference on Computer Vision

Burgos-Artizzu XP, Perona P, Dollár P (2013) Robust face landmark estimation under occlusion. In: IEEE International Conference on Computer Vision

Cao X, Wei Y, Wen F, Sun J (2012) Face alignment by explicit shape regression. In: IEEE Conference on Computer Vision and Pattern Recognition

Comaniciu D, Meer P (2002) Mean Shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5):603–619

Cootes TF, Taylor CJ (2004) Statistical models of appearance for computer vision. Tech. rep., Imaging Science and Biomedical Engineering, University of Manchester

Cootes TF, Taylor CJ, Cooper DH, Graham J (1995) Active shape models-their training and application. Computer Vision and Image Understanding 61(1):38–59

Cootes TF, Edwards GJ, Taylor CJ (2001) Active appearance models. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(6):681–685

Cootes TF, Ionita M, Lindner C, Sauer P (2012) Robust and accurate shape model fitting using random forest regression voting. In: European Conference on Computer Vision

Cristinacce D, Cootes TF (2006) Feature detection and tracking with constrained local models. In: British Machine Vision Conference

Cristinacce D, Cootes TF (2007) Boosted regression active shape models. In: British Machine Vision Conference

Cristinacce D, Cootes TF (2008) Automatic feature localisation with constrained local models. Pattern Recognition 41(10):3054–3067

Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition

Dantone M, Gall J, Fanelli G, Gool LV (2012) Real-time facial feature detection using conditional regression forests. In: IEEE Conference on Computer Vision and Pattern Recognition

Face++ (2018) Megvii face api. URL http://www.faceplusplus.com

Fan H, Zhou E (2016) Approaching human level facial landmark localization by deep learning. Image and Vision Computing 47:27–35

Fanelli G, Dantone M, Gool LV (2013) Real time 3d face alignment with random forests-based active appearance models. In: IEEE International Conference on Automatic Face and Gesture Recognition

Galoogahi HK, Sim T, Lucey S (2013) Multi-channel correlation filters. In: IEEE International Conference on Computer Vision

Gu L, Kanade T (2008) A generative shape regularization model for robust face alignment. In: European Conference on Computer Vision

Guo P (2014) URL https://github.com/phg1024/CSCE625/tree/master/finalproject

Henriques JF, Carreira J, Caseiro R, Batista J (2013) Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In: IEEE

International Conference on Computer Vision

Huang GB, Ramesh M, Berg T, Learned-Miller E (2007) Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep. 07-49, University of Massachusetts, Amherst

Huang Z, Zhou E, Cao Z (2015) Coarse-to-fine face alignment with multi-scale local patch regression, arXiv:1511.04901

Jacobs HO (2014) How to stare at the higher-order n-dimensional chain rule without losing your marbles. Tech. Rep. arXiv:1410.3493v3

Jourabloo A, Liu X (2015) Pose-invariant 3d face alignment. In: IEEE International Conference on Computer Vision

Kazemi V, Sullivan J (2014) One millisecond face alignment with an ensemble of regression trees. In: IEEE Conference on Computer Vision and Pattern Recognition

Le V, Brandt J, Lin Z, Boudev L, Huang TS (2012) Interactive facial feature localization. In: European Conference on Computer Vision

Lee D, Park H, Yoo CD (2015) Face alignment using cascade gaussian process regression trees. In: IEEE Conference on Computer Vision and Pattern Recognition

Lowe DG (2004) Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2):91–110

Martins P, Caseiro R, Henriques JF, Batista J (2012) Discriminative bayesian active shape models. In: European Conference on Computer Vision

Martins P, Caseiro R, Batista J (2014) Non-parametric bayesian constrained local models. In: IEEE Conference on Computer Vision and Pattern Recognition

Martins P, Henriques JF, Caseiro R, Batista J (2016) Bayesian constrained local models revisited. IEEE Transactions on Pattern Analysis and Machine Intelligence 38(4):704–716

Matthews I, Baker S (2004) Active appearance models revisited. International Journal of Computer Vision 60(1):135–164

Messer K, Matas J, Kittler J, Luettin J, Maitre G (1999) XM2VTSDB: The extended M2VTS database. In: International Conference on Audio and Video-based Biometric Person Authentication

Newell A, Yang K, Deng J (2016) Stacked hourglass networks for human pose estimation. In: European Conference on Computer Vision

Nordstrom M, Larsen M, Sierakowski J, Stegmann M (2004) The IMM face database - an annotated dataset of 240 face images. Tech. rep., Technical University of Denmark, DTU

Paquet U (2009) Convexity and bayesian constrained local models. In: IEEE Conference on Computer Vision and Pattern Recognition

Ren S, Cao X, Wei Y, Sun J (2014) Face alignment at 3000 fps via regressing local binary features. In: IEEE Conference on Computer Vision and Pattern Recognition

Sagonas C, Tzimiropoulos G, Zafeiriou S, Pantic M (2013a) 300 faces in-the-wild challenge: The first facial landmark localization challenge. In: IEEE International Conference on Computer Vision Workshops

Sagonas C, Tzimiropoulos G, Zafeiriou S, Pantic M (2013b) A semi-automatic methodology for facial landmark annotation. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops

Sagonas C, Antonakos E, Tzimiropoulos G, Pantic M (2016) 300 faces in-the-wild challenge: database and results. Image and Vision Computing, Special Issue on Facial Landmark Localisation 'In-The-Wild' 47:3–18

Sánchez-Lozano E, Tzimiropoulos G, Martinez B, De la Torre F, Valstar M (2018) A functional regression approach to facial landmark tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence DOI 10.1109/TPAMI.2017.2745568

Saragih J, Lucey S, Cohn J (2009) Face alignment through subspace constrained mean-shifts. In: IEEE International Conference on Computer Vision

Saragih J, Lucey S, Cohn J (2010) Deformable model fitting by regularized landmark mean-shifts. International Journal of Computer Vision 91(2):200–215

Silverman BW (1986) Density Estimation for Statistics and Data Analysis. Chapman and Hall, London

Songsri-in K, Trigeorgis G, Zafeiriou S (2018) Deep & deformable: Convolutional mixtures of deformable part-based models. In: IEEE Conference on Automatic Face and Gesture Recognition

Sun Y, Wang X, Tang X (2013) Deep convolutional network cascade for facial point detection. In: IEEE Conference on Computer Vision and Pattern Recognition

Trigeorgis G, Snape P, Nicolaou M, Antonakos E, Zafeiriou S (2016) Mnemonic descent method: A recurrent process applied for end-to-end face alignment. In: IEEE Conference on Computer Vision and Pattern Recognition

Tzimiropoulos G (2015) Project-out cascaded regression with an application to face alignment. In: IEEE Conference on Computer Vision and Pattern Recognition

Tzimiropoulos G, Pantic M (2014) Gauss-newton deformable part models for face alignment in-the-wild. In: IEEE Conference on Computer Vision and Pattern Recognition

Tzimiropoulos G, Pantic M (2017) Fast algorithms for fitting active appearance models to unconstrained images. International Journal of Computer Vision 122(1):17–33

Tzimiropoulos G, i Medina JA, Zafeiriou S, Pantic M (2012) Generic active appearance models revisited. In: Asian Conference on Computer Vision

Valstar MF, Martinez B, Binefa X, Pantic M (2010) Facial point detection using boosted regression and graph models. In: IEEE Conference on Computer Vision and Pattern Recognition

Viola P, Jones M (2002) Robust real-time object detection. International Journal of Computer Vision 57(2):137–154

Wang Y, Lucey S, Cohn J (2008) Enforcing convexity for improved alignment with constrained local models. In: IEEE Conference on Computer Vision and Pattern Recognition

Xiao J, Baker S, Matthews I, Kanade T (2004a) Real-time combined 2d+3d active appearance models. In: IEEE Conference on Computer Vision and Pattern Recognition

Xiao J, Chai J, Kanade T (2004b) A closed-form solution to non-rigid shape and motion recovery. In: European Conference on Computer Vision

Xiong X, De la Torre F (2013) Supervised descent method and its application to face alignment. In: IEEE Conference on Computer Vision and Pattern Recognition

Xiong X, De la Torre F (2014) Supervised descent method for solving nonlinear least squares problems in computer vision. Tech. Rep. arXiv:1405.0601

Xiong X, De la Torre F (2015) Global supervised descent method. In: IEEE Conference on Computer Vision and Pattern Recognition

Zadeh A, Baltrušaitis T, Morency LP (2017) Convolutional experts constrained local model for facial landmark detection. In: IEEE Computer Vision and Pattern Recognition Workshop (CVPRW), 2nd Facial Landmark Localisation Competition

Zhang J, Shan S, Kan M, Chen X (2014a) Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In: European Conference on Computer Vision

Zhang Z, Luo P, Loy CC, Tang X (2014b) Facial landmark detection by deep multi-task learning. In: European Conference on Computer Vision

Zhang Z, Luo P, Loy CC, Tang X (2016) Learning deep representation for face alignment with auxiliary attributes. IEEE Transactions on Pattern Analysis and Machine Intelligence 38:918–930

Zhou E, Fan H, Cao Z, Jiang Y, Yin Q (2013a) Extensive facial landmark localization with coarse-to-fine convolutional network cascade. In: IEEE International Conference on Computer Vision Workshop, 300 Faces In-the-Wild Challenge (300-W)

Zhou F, Brandt J, Lin Z (2013b) Exemplar-based graph matching for robust facial landmark localization. In: IEEE International Conference on Computer Vision

Zhu S, Li C, Loy C, Tang X (2015) Face alignment by coarse-to-fine shape searching. In: IEEE Conference on Computer Vision and Pattern Recognition

Zhu X, Ramanan D (2012) Face detection, pose estimation, and landmark localization in the wild. In: IEEE Conference on Computer Vision and Pattern Recognition

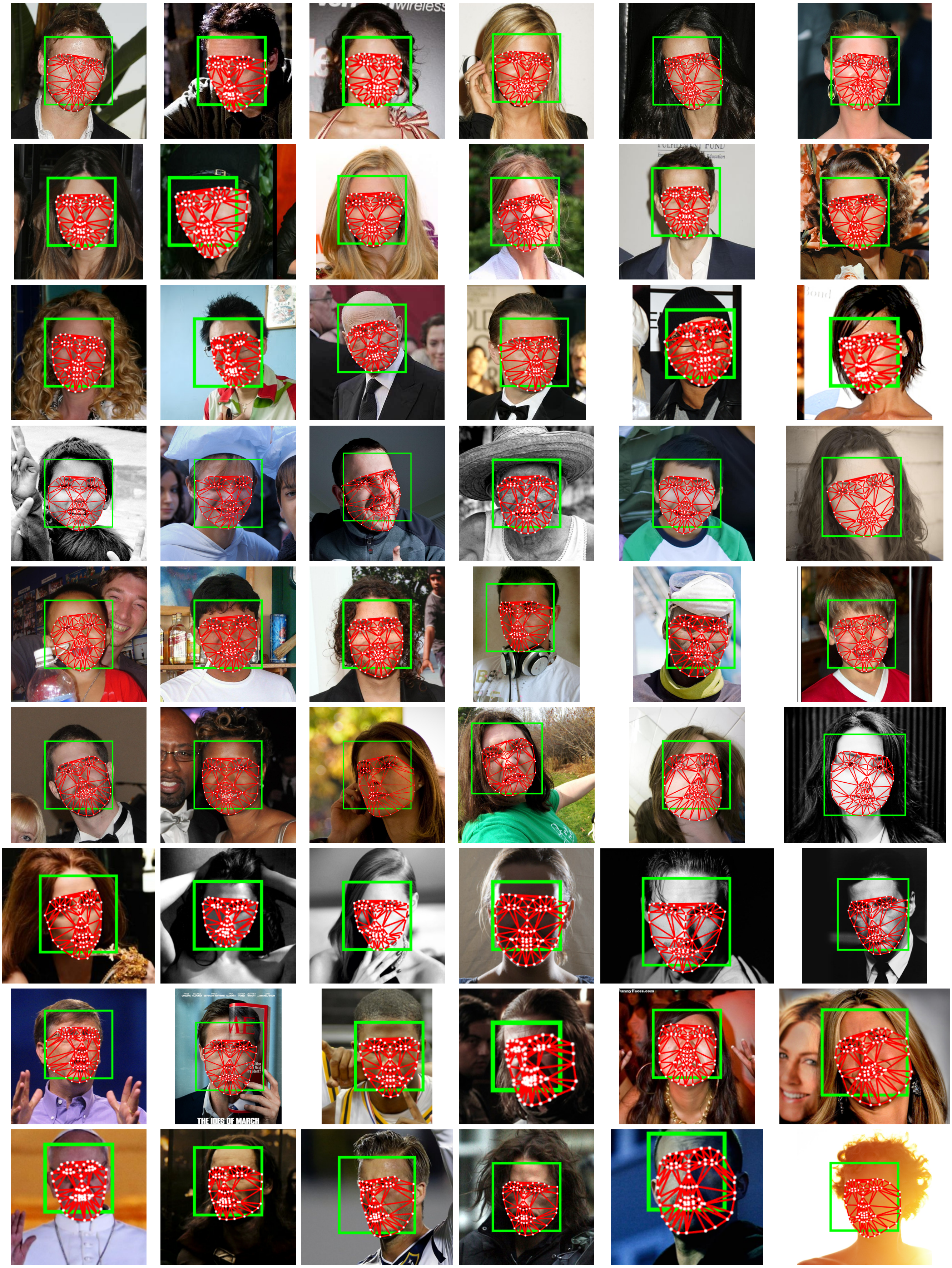


Fig. 9 Fitting examples in the LFPW, HELEN and 300W databases (top three rows, middle rows and last three rows, respectively) taken using the proposed GSM-CR technique.

A Gradient Definitions

A.1 Hessian of the 2D Regularization Term ($\mathbf{H}_{\mathrm{R}}$)

The Hessian of the 2D regularization term is a $(2v+4)$ square matrix of the form:

$$\mathbf{H}_{\mathrm{R}} = \begin{bmatrix} \ddots & \cdots & & & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial^2 R}{\partial x_i^2} & & & & \frac{\partial^2 R}{\partial x_i \partial a} & \frac{\partial^2 R}{\partial x_i \partial b} & \frac{\partial^2 R}{\partial x_i \partial t_x} & \frac{\partial^2 R}{\partial x_i \partial t_y} \\ \vdots & \ddots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & \frac{\partial^2 R}{\partial y_i^2} & & \frac{\partial^2 R}{\partial y_i \partial a} & \frac{\partial^2 R}{\partial y_i \partial b} & \frac{\partial^2 R}{\partial y_i \partial t_x} & \frac{\partial^2 R}{\partial y_i \partial t_y} \\ & \cdots & & \ddots & \vdots & \vdots & \vdots & \vdots \\ \cdots \frac{\partial^2 R}{\partial a \partial x_i} & \cdots & \frac{\partial^2 R}{\partial a \partial y_i} & \cdots & \frac{\partial^2 R}{\partial a^2} & \frac{\partial^2 R}{\partial a \partial b} & \frac{\partial^2 R}{\partial a \partial t_x} & \frac{\partial^2 R}{\partial a \partial t_y} \\ \cdots \frac{\partial^2 R}{\partial b \partial x_i} & \cdots & \frac{\partial^2 R}{\partial b \partial y_i} & \cdots & \frac{\partial^2 R}{\partial b \partial a} & \frac{\partial^2 R}{\partial b^2} & \frac{\partial^2 R}{\partial b \partial t_x} & \frac{\partial^2 R}{\partial b \partial t_y} \\ \cdots \frac{\partial^2 R}{\partial t_x \partial x_i} & \cdots & \frac{\partial^2 R}{\partial t_x \partial y_i} & \cdots & \frac{\partial^2 R}{\partial t_x \partial a} & \frac{\partial^2 R}{\partial t_x \partial b} & \frac{\partial^2 R}{\partial t_x^2} & \frac{\partial^2 R}{\partial t_x \partial t_y} \\ \cdots \frac{\partial^2 R}{\partial t_y \partial x_i} & \cdots & \frac{\partial^2 R}{\partial t_y \partial y_i} & \cdots & \frac{\partial^2 R}{\partial t_y \partial a} & \frac{\partial^2 R}{\partial t_y \partial b} & \frac{\partial^2 R}{\partial t_y \partial t_x} & \frac{\partial^2 R}{\partial t_y^2} \end{bmatrix} \tag{52}$$

where the main $(2v \times 2v)$ sub-matrix (constant, therefore can be precomputed) is

$$\frac{\partial^2 R}{\partial \mathbf{s}^2} = 2\Sigma_{\mathbf{s}}^{-1}. \tag{53}$$

The 2D pose diagonal terms are given by

$$\begin{aligned} \frac{\partial^2 R}{\partial a^2} &= \left(\frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial a}\right)^T \frac{\partial^2 R}{\partial \mathbf{s}_{\mathrm{BM}}^2} \frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial a} \\ &= 2\left(\mathbf{s} - \mathbf{s}_m\right)^T \Sigma_{\mathbf{s}}^{-1} \left(\mathbf{s} - \mathbf{s}_m\right) \end{aligned} \tag{54}$$

$$\begin{aligned} \frac{\partial^2 R}{\partial b^2} &= \left(\frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial b}\right)^T \frac{\partial^2 R}{\partial \mathbf{s}_{\mathrm{BM}}^2} \frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial b} \\ &= 2\begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix}^T \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix} \end{aligned} \tag{55}$$

$$\frac{\partial^2 R}{\partial t_x^2} = \left(\frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial t_x}\right)^T \frac{\partial^2 R}{\partial \mathbf{s}_{\mathrm{BM}}^2} \frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial t_x} = 2\begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix}^T \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix} \tag{56}$$

$$\frac{\partial^2 R}{\partial t_y^2} = \left(\frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial t_y}\right)^T \frac{\partial^2 R}{\partial \mathbf{s}_{\mathrm{BM}}^2} \frac{\partial \mathbf{s}_{\mathrm{BM}}}{\partial t_y} = 2\begin{pmatrix} \mathbf{0}_v \\ \mathbf{1}_v \end{pmatrix}^T \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{0}_v \\ \mathbf{1}_v \end{pmatrix} \tag{57}$$

where $\mathbf{0}_v$ and $\mathbf{1}_v$ are v sized vectors filled with zeros and ones, respectively. In the previous, $\mathbf{s}^x$ and $\mathbf{s}^y$ represent the x and y components (v sized vectors) of the shape $\mathbf{s}$. Additionally, note that $\mathbf{s}_m$ ($2v$ expanded vector that defines the base mesh centre of mass) is constant.

The 2D pose mixed terms are given by

$$\frac{\partial^2 R}{\partial a \partial b} = \frac{\partial^2 R}{\partial b \partial a} = 2\left(\mathbf{s} - \mathbf{s}_m\right) \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix} \tag{58}$$

$$\frac{\partial^2 R}{\partial a \partial t_x} = \frac{\partial^2 R}{\partial t_x \partial a} = 2\left(\mathbf{s} - \mathbf{s}_m\right) \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix} \tag{59}$$

$$\frac{\partial^2 R}{\partial a \partial t_y} = \frac{\partial^2 R}{\partial t_y \partial a} = 2\left(\mathbf{s} - \mathbf{s}_m\right) \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{0}_v \\ \mathbf{1}_v \end{pmatrix} \tag{60}$$

$$\frac{\partial^2 R}{\partial b \partial t_x} = \frac{\partial^2 R}{\partial t_x \partial b} = 2\begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix} \tag{61}$$

$$\frac{\partial^2 R}{\partial b \partial t_y} = \frac{\partial^2 R}{\partial t_y \partial b} = 2\begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{0}_v \\ \mathbf{1}_v \end{pmatrix} \tag{62}$$

$$\frac{\partial^2 R}{\partial t_x \partial t_y} = \frac{\partial^2 R}{\partial t_y \partial t_x} = 2\begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix}^T \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{0}_v \\ \mathbf{1}_v \end{pmatrix}. \tag{63}$$

Finally, the remaining mixed terms, are

$$\begin{aligned} \frac{\partial^2 R}{\partial x_i \partial a} &= \frac{\partial^2 R}{\partial a \partial x_i} = 2\begin{pmatrix} a\boldsymbol{\delta}_i \\ b\boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \left(\mathbf{s} - \mathbf{s}_m\right) \\ &\quad + 2\begin{pmatrix} \boldsymbol{\delta}_i \\ \mathbf{0}_v \end{pmatrix} \Sigma_{\mathbf{s}}^{-1}(\mathbf{s}_{\mathrm{BM}} - \mathbf{s}_0) \end{aligned} \tag{64}$$

$$\begin{aligned} \frac{\partial^2 R}{\partial y_i \partial a} &= \frac{\partial^2 R}{\partial a \partial y_i} = 2\begin{pmatrix} -b\boldsymbol{\delta}_i \\ a\boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \left(\mathbf{s} - \mathbf{s}_m\right) \\ &\quad + 2\begin{pmatrix} \mathbf{0}_v \\ \boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1}(\mathbf{s}_{\mathrm{BM}} - \mathbf{s}_0) \end{aligned} \tag{65}$$

$$\begin{aligned} \frac{\partial^2 R}{\partial x_i \partial b} &= \frac{\partial^2 R}{\partial b \partial x_i} = 2\begin{pmatrix} a\boldsymbol{\delta}_i \\ b\boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix} \\ &\quad + \begin{pmatrix} \mathbf{0}_v \\ \boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1}(\mathbf{s}_{\mathrm{BM}} - \mathbf{s}_0) \end{aligned} \tag{66}$$

$$\begin{aligned} \frac{\partial^2 R}{\partial y_i \partial b} &= \frac{\partial^2 R}{\partial b \partial y_i} = 2\begin{pmatrix} -b\boldsymbol{\delta}_i \\ a\boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{s}_m^y - \mathbf{s}^y \\ \mathbf{s}^x - \mathbf{s}_m^x \end{pmatrix} \\ &\quad + \begin{pmatrix} -\boldsymbol{\delta}_i \\ \mathbf{0}_v \end{pmatrix} \Sigma_{\mathbf{s}}^{-1}(\mathbf{s}_{\mathrm{BM}} - \mathbf{s}_0) \end{aligned} \tag{67}$$

$$\frac{\partial^2 R}{\partial x_i \partial t_x} = \frac{\partial^2 R}{\partial t_x \partial x_i} = 2\begin{pmatrix} a\boldsymbol{\delta}_i \\ b\boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix} \tag{68}$$

$$\frac{\partial^2 R}{\partial y_i \partial t_x} = \frac{\partial^2 R}{\partial t_x \partial y_i} = 2\begin{pmatrix} -b\boldsymbol{\delta}_i \\ a\boldsymbol{\delta}_i \end{pmatrix} \Sigma_{\mathbf{s}}^{-1} \begin{pmatrix} \mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix}. \tag{69}$$

where $\boldsymbol{\delta}_i$ is a v-dimensional vector filled with zeros, except on a scalar of 1 at the i^{th} element location.

A.2 Gradients of the 2D+3D Model

The gradient of the 3D regularization term, in eq. 42, is

$$\nabla_{\mathrm{R3D}}(\bar{\mathbf{s}}) = \left[\, \mathbf{0}_{2v+4} \; 2\Sigma_{\bar{\mathbf{s}}}^{-1}(\bar{\mathbf{s}} - \bar{\mathbf{s}}_0) \; \mathbf{0}_6 \,\right]. \tag{70}$$

Recalling gradient of the 3D to 2D projection error, defined by a $(2v) \times (2v+4+3v+6)$ matrix as

$$\nabla \mathbf{r} = \left[\, \frac{\partial \mathbf{r}}{\partial \mathbf{s}} \; \frac{\partial \mathbf{r}}{\partial \boldsymbol{\theta}} \; \frac{\partial \mathbf{r}}{\partial \bar{\mathbf{s}}} \; \frac{\partial \mathbf{r}}{\partial \sigma} \; \frac{\partial \mathbf{r}}{\partial \Delta\theta_x} \; \frac{\partial \mathbf{r}}{\partial \Delta\theta_y} \; \frac{\partial \mathbf{r}}{\partial \Delta\theta_z} \; \frac{\partial \mathbf{r}}{\partial o_x} \; \frac{\partial \mathbf{r}}{\partial o_y} \,\right]$$

with

$$\frac{\partial \mathbf{r}}{\partial \mathbf{s}} = \mathbf{I}_{2v}, \quad \frac{\partial \mathbf{r}}{\partial \boldsymbol{\theta}} = \mathbf{0}_{2v}, \quad \frac{\partial \mathbf{r}}{\partial \bar{\mathbf{s}}} = -\mathbf{P} \otimes \mathbf{I}_v, \quad \frac{\partial \mathbf{r}}{\partial \sigma} = -\mathbf{I}_v \otimes \mathbf{R}_o \, \bar{\mathbf{s}},$$

$$\frac{\partial \mathbf{r}}{\partial \Delta\theta_x} = \mathbf{I}_v \otimes \left(\mathbf{P} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \right) \bar{\mathbf{s}}, \quad \frac{\partial \mathbf{r}}{\partial \Delta\theta_y} = \mathbf{I}_v \otimes \left(\mathbf{P} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \right) \bar{\mathbf{s}},$$

$$\frac{\partial \mathbf{r}}{\partial \Delta\theta_z} = \mathbf{I}_v \otimes \left(\mathbf{P} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right) \bar{\mathbf{s}}, \quad \frac{\partial \mathbf{r}}{\partial o_x} = \begin{pmatrix} -\mathbf{1}_v \\ \mathbf{0}_v \end{pmatrix}, \quad \frac{\partial \mathbf{r}}{\partial o_y} = \begin{pmatrix} \mathbf{0}_v \\ -\mathbf{1}_v \end{pmatrix}$$

where $\mathbf{I}_n$ represents a n dimensional identity matrix and the $\otimes$ symbol is the Kronecker product.

Finally, the Hessian of the 3D shape regularization term (in eq. 43) is a $(2v+4+3v+6)$ square matrix given by

$$\mathbf{H}_{\mathrm{R3D}}(\bar{\mathbf{s}}) = \begin{bmatrix} \mathbf{0}_{2v} & \mathbf{0}_4 & \mathbf{0}_{3v} & \mathbf{0}_6 \\ \mathbf{0}_{2v} & \mathbf{0}_4 & \mathbf{0}_{3v} & \mathbf{0}_6 \\ \mathbf{0}_{2v} & \mathbf{0}_4 & 2\Sigma_{\bar{\mathbf{s}}}^{-1} & \mathbf{0}_6 \\ \mathbf{0}_{2v} & \mathbf{0}_4 & \mathbf{0}_{3v} & \mathbf{0}_6 \end{bmatrix} \tag{71}$$

note that this matrix is constant, therefore, it can be precomputed.