

Automatically Learning the Objective Function for Model Fitting

Matthias WIMMER[†] and Bernd RADIG[†]

[†] Fakultät für Informatik, Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany

Abstract Model-based image interpretation has proven to be appropriate to extract high-level information from images. A priori knowledge about the object of interest represents the basis of this task. Fitting algorithms search for the model that best matches a given image by searching for the global optimum of an objective function. The objective function is usually designed manually, based on implicit and domain-dependent knowledge. In this paper, we present an approach that learns objective functions from annotated training images. It automates many critical decisions and the remaining manual steps hardly require domain-dependent knowledge. This approach yields highly accurate objective functions. Our evaluation fits a face model to a publicly available image database and compares the obtained results to a recent state-of-the-art approach.

Key words Model fitting, facial feature extraction, machine learning, model-based image interpretation

1. Introduction

Model-based image interpretation systems exploit a priori knowledge about objects, such as shape or texture. Model fitting is the computational challenge of finding the model configuration that describes the content of the image best [1]. Model-based image interpretation consists of three main components: the model, the fitting algorithm, and the objective function.

The *model* contains a parameter vector \mathbf{p} that represents its configuration, such as position, rotation, scaling, and deformation. Those parameters are usually mapped to the surface of an image, via a contour or a textured region. The *objective function* $f(I, \mathbf{p})$ yields a comparable value that determines how accurately a parameterized model \mathbf{p} fits to an image I . In this paper, smaller values denote a better model fit. Depending on context, they are also known as the likelihood, similarity, energy, cost, goodness or quality functions. The *fitting algorithm* searches for the model parameters \mathbf{p} that optimize the objective function. Since the described methods are independent of the used fitting algorithm, this paper shall not elaborate on them but we refer to [1] for a recent overview and categorization.

Problem Statement. Fitting algorithms have been the subject of intensive research and evaluation [1]. This comprises the challenge of initial pose estimation for model tracking as well. In contrast, the involved objective function is usually determined ad hoc and heuristically, using the designer's intuitions about a good measure of fitness. He manually selects salient image features and manually composes

the calculation rules. The appropriateness of the objective function is subjectively determined by inspecting example images and example model parameterizations. If the result is not satisfactory the objective function is tuned or redesigned from scratch, see Figure 1 (upper). The consequences are that this design approach requires much implicit and domain-dependent knowledge and therefore does not objectively determine the best model fit. Its iterative nature also makes it a time-consuming process of unpredictable duration.

Solution Idea. The proposed approach focuses on improving the objective function based on objective measures. We explicitly formulate the properties of *ideal* objective functions and give a concrete example of such a function. Since it is based on manual image annotations, it cannot be applied to previously unseen images. Therefore, we propose to learn the objective function from comprehensive training data specified partly by hand and partly with the help of the ideal objective function. Therefore, this methodology achieves high accuracy, because it approximates the properties of the ideal objective function. Most steps are automated and the remaining manual steps require little domain-dependent knowledge, see Figure 1 (lower). Furthermore, the *design-inspect* loop is eliminated, which makes the time requirements predictable.

Outline. Section 2. describes the traditional approach and points out its shortcomings. Section 3. elaborates on ideal objective functions. Section 4. explains the proposed learn approach in detail. Section 5. experimentally evaluates the obtained results. Section 6. summarizes our contributions and suggests further work.

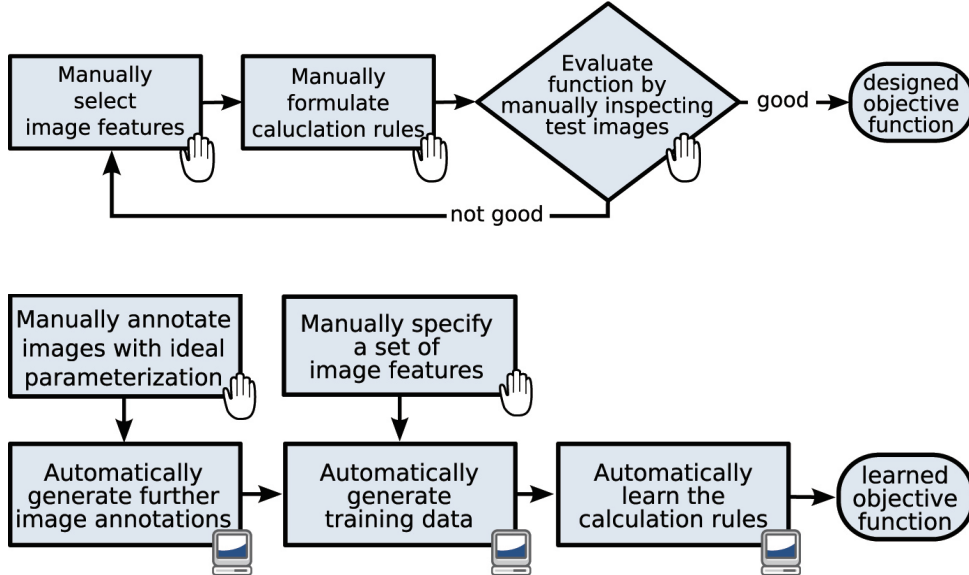


Fig. 1 The traditional procedure for designing objective functions (upper), and the proposed method for learning objective functions from annotated training images (lower).

2. Design Approach

In order to explain the proposed technique, this paper utilizes a two-dimensional, deformable, contour model of a human face according to the Active Shape Model approach [2]. The model parameters $\mathbf{p}=(t_x, t_y, s, \theta, \mathbf{b})^T$ describes the translation t_x and t_y , the scaling s , the rotation θ and the deformation \mathbf{b} . The projection function $\mathbf{c}_n(\mathbf{p})$ computes the location of the n^{th} contour point with $1 \leq n \leq N$.

Model-based techniques require to determine the minimum of the objective function $f(I, \mathbf{p})$. According to common approaches [2], we split the objective function into N local parts $f_n(I, \mathbf{x})$, one for each contour point, see Equation 1. These local objective functions evaluate the content of the image in the vicinity of the contour point and give evidence about its fitness. The result of the global objective function is the sum of the local function values. From now on, we will concentrate on the local objective functions f_n and refer to them as objective functions. Note, that $f_n(I, \mathbf{x})$ must be optimized in pixel space $\mathbf{x} \in \mathbb{R}^2$, whereas $f(I, \mathbf{p})$ must be optimized in parameter space $\mathbf{p} \in \mathbb{R}^P$ with $P = \dim(\mathbf{p})$ and $2 \ll P$.

$$f(I, \mathbf{p}) = \sum_{n=1}^N f_n(I, \mathbf{c}_n(\mathbf{p})) \quad (1)$$

Objective functions are usually designed by manually selecting salient features from the image and mathematically composing them, as illustrated in Figure 1 (upper). The feature selection and the mathematical composition are both based on the designer’s intuition and implicit knowledge of the domain. In [2] for instance, the objective function is computed from edge values of the image. Each contour point is

considered to be located well if it overlaps a strong edge of the image. A similar objective function is shown in Equation 2, where $0 \leq E(I, \mathbf{x}) \leq 1$ denotes the edge magnitude.

$$f_n^e(I, \mathbf{x}) = 1 - E(I, \mathbf{x}) \quad (2)$$

Unfortunately, the traditional approach of designing objective functions has comprehensive shortcomings and unexpected side-effects as illustrated with the example in Figure 2. 2a) visualizes one of the contour points of the face model as well as its perpendicular towards the contour. 2b) and 2c) depict the content of the image along this perpendicular as well as the corresponding edge magnitudes $E(I, \mathbf{x})$. 2d) shows the value of the designed objective function f_n^e along the perpendicular. Obviously, this function has many local minima within this one-dimensional space. Furthermore, the global minimum does not correspond to the ideal location that is denoted by the vertical line. Because of this amount of local minima, fitting algorithms have difficulty in finding the global minimum. Even if an algorithm found the global minimum, it would be wrong, because it does not correspond to the ideal location.

3. Ideal Objective Functions

This section makes the observations from Figure 2 explicit by formulating two properties P1 and P2. We call an objective function *ideal* once it has both of them. The mathematical formalization of P1 uses the *ideal* model parameters \mathbf{p}_I^* , which are the parameters with the best fitness to a specific image I . Similarly, $\mathbf{c}_n(\mathbf{p}_I^*)$ denote the ideal contour points. Note that \mathbf{p}_I^* must be determined manually.

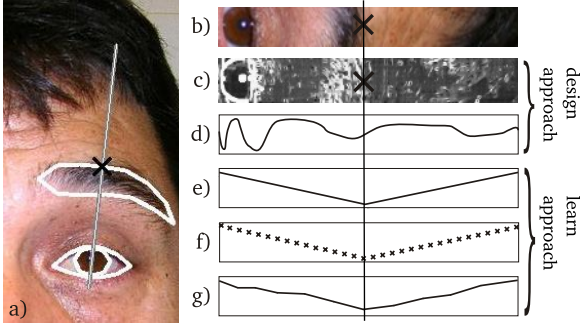


Fig. 2 a) Contour point with perpendicular, b) Image data, c) Edge magnitudes, d) Designed objective function f_n^e , e) Ideal objective function f_n^* , f) Training data, g) Learned objective function f_n^l ; Note, b) – g) are taken along that perpendicular visible in a). The vertical line represents the location of the ideal contour point $\mathbf{c}_n(\mathbf{p}_I^*)$

P1: Correctness property: The global minimum corresponds to the best model fit.

$$\forall \mathbf{x}(\mathbf{c}_n(\mathbf{p}_I^*) \neq \mathbf{x}) \Rightarrow f_n(I, \mathbf{c}_n(\mathbf{p}_I^*)) < f_n(I, \mathbf{x})$$

P2: Uni-modality property: The objective function has no local extrema.

$$\exists \mathbf{m} \forall \mathbf{x} (\mathbf{m} \neq \mathbf{x}) \Rightarrow f_n(I, \mathbf{m}) < f_n(I, \mathbf{x})$$

$$\wedge \nabla f_n(I, \mathbf{x}) \neq \mathbf{0}$$

Note that P2 guarantees that any determined minimum represents the global minimum. This facilitates search, because fitting algorithms can not get stuck in local minima. The global minimum \mathbf{m} does not need to correspond to the best fit. This is only required by the independent property P1.

Equation 3 introduces a concrete instance of an ideal objective function $f_n^*(I, \mathbf{x})$. It computes the distance between the ideal contour point $\mathbf{c}_n(\mathbf{p}_I^*)$ and a pixel \mathbf{x} located on the image surface. A significant feature of f_n^* is that it uses the ideal parameters \mathbf{p}_I^* to compute its value. This implies that f_n^* cannot be applied to previously unseen images, because the value of \mathbf{p}_I^* is not known for these images.

$$f_n^*(I, \mathbf{x}) = |\mathbf{x} - \mathbf{c}_n(\mathbf{p}_I^*)| \quad (3)$$

4. Learn Approach

This section explains the five steps of our approach that learns the objective function $f_n^l(I, \mathbf{x})$ from annotated training images, see Figure 1 (lower). The key idea behind the approach is that f_n^* has the properties P1 and P2 and will be the basis for learning f_n^l . Therefore, this learned function will also approximately have these properties.



Fig. 3 Four images that are manually annotated with the ideal parameters of our face model.

4.1 Annotating Images with Ideal Model Parameters

We manually annotate a lot of images I_k with $1 \leq k \leq K$ with the ideal model parameters $\mathbf{p}_{I_k}^*$. For real-world images, however, the ideal model parameters depend on the user's judgment. Note that for synthetic images, \mathbf{p}_I^* is known and can be used in such cases. These parameters enable computing the ideal objective function f_n^* in Equation 3. This annotation is the only laborious step in the entire procedure of the proposed approach. Figure 3 shows four images that are annotated with the ideal parameters of our face model.

4.2 Generating Further Image Annotations

According to P1, the ideal objective function returns the minimum $f_n^*(I, \mathbf{x})=0$ for all image annotations, because $\mathbf{x}=\mathbf{c}_n(\mathbf{p}_I^*)$. This data is not sufficient to learn the characteristics of f_n^l . Therefore, we will acquire image annotations $\mathbf{x} \neq \mathbf{c}_n(\mathbf{p}_I^*)$, for which $f_n^*(I, \mathbf{x}) \neq 0$. In general, any position within the image may represent one of these annotations. However, it is more practicable to restrict this motion in terms of distance and direction.

Therefore, we generate a number of displacements $\mathbf{x}_{k,n,d}$ with $-D \leq d \leq D$ that are located on the perpendicular to the contour line with a maximum distance Δ to the contour point. This procedure is illustrated in Figure 4. The center row depicts the manually annotated images, for which $f_n^*(I, \mathbf{x}_{k,n,0}) = f_n^*(I, \mathbf{c}_n(\mathbf{p}_{I_k}^*)) = 0$. The other rows depict the displacements $\mathbf{x}_{k,n,d \neq 0}$ with $f_n^*(I, \mathbf{x}_{k,n,d \neq 0}) > 0$ as defined by P1.

Due to different image sizes, the size of the visible face varies substantially. Distance measures, such as the return value of the ideal objective function, error measures and Δ , should not be biased by this variation. Therefore, all distances in pixels are converted to the interocular measure, by dividing them by the pixel distance between the pupils.

4.3 Specifying Image Features

Our approach learns a mapping from I_k and $\mathbf{x}_{k,n,d}$ to $f_n^l(I_k, \mathbf{x}_{k,n,d})$, which is called $f_n^l(I, \mathbf{x})$. Since f_n^l has no access to \mathbf{p}_I^* , it must compute its value from the content of the image. Instead of learning a direct mapping from \mathbf{x} and I to f_n^* , we use a feature-extraction method [1]. The idea is to provide a multitude of image features, and let the

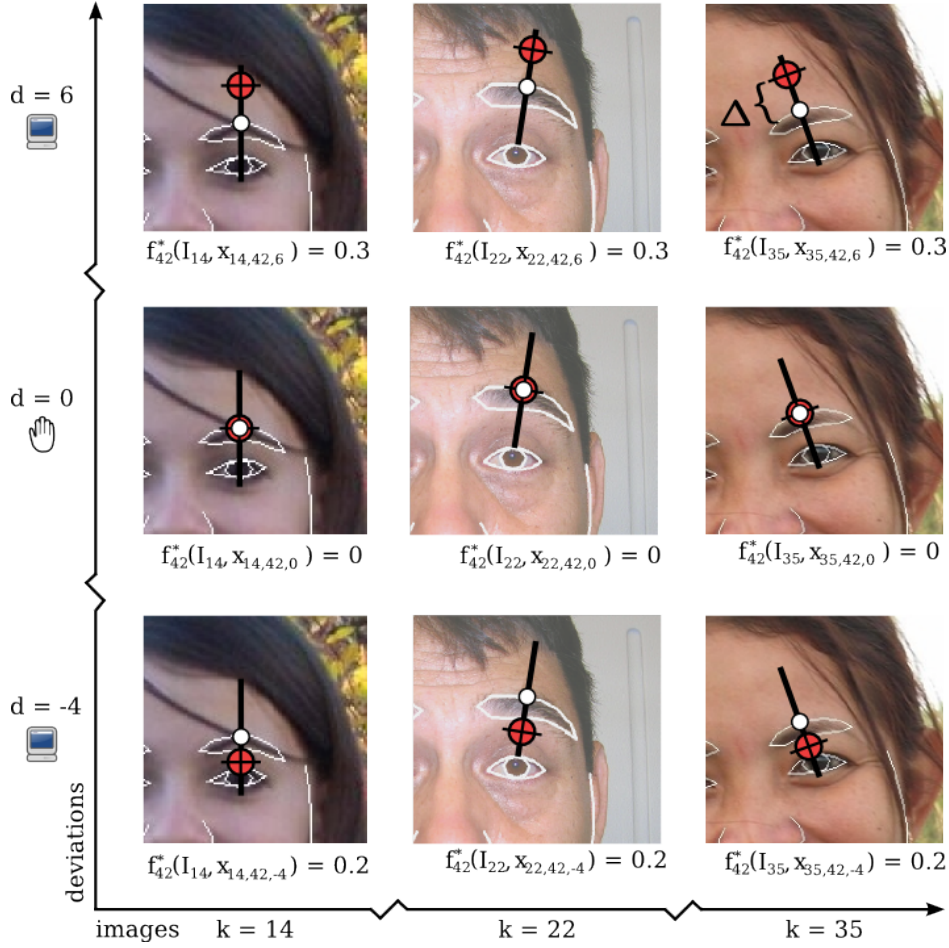


Fig. 4 For the considered contour point, each of the K images is annotated with $2D+1$ displacements. Manual work is only necessary for annotating $d=0$ in the middle row. The other displacements are computed automatically. Note Δ in the upper right image that indicates the learning radius. The unit of the ideal objective function values and Δ is the interocular measure.

learning algorithm choose which of them are relevant to the computation rules of the objective function.

In our approach, we use Haar-like image features of different styles and sizes [3], see Figure 5. These features greatly cope with noisy images. They are not only computed at the location of the contour point itself, but also at locations within its vicinity specified by a grid. This number of A image features enables the objective function to exploit the texture of the image at the model's contour point and in its surrounding area. We denote image features by $\mathbf{h}_a(I, \mathbf{x})$,

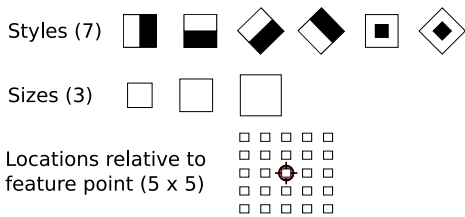


Fig. 5 The set of $A=7 \cdot 3 \cdot 5 \cdot 5=525$ features utilized for learning the objective functions.

with $1 \leq a \leq A$. Each of these features returns a scalar value.

4.4 Generating Training Data

The result of the manual annotation step (Section 4.1) and the automated annotation step (Section 4.2) is a list of $K(2D+1)$ image locations for each of the N contour points. Adding the corresponding target value f_n^* yields the list in Equation 4.

$$[I_k, \mathbf{x}_{k,n,d}, f_n^*(I_k, \mathbf{x}_{k,n,d})] \quad (4)$$

$$[\mathbf{h}_1(I_k, \mathbf{x}_{k,n,d}), \dots, \mathbf{h}_A(I_k, \mathbf{x}_{k,n,d}), f_n^*(I_k, \mathbf{x}_{k,n,d})] \quad (5)$$

$$\text{with } 1 \leq k \leq K, 1 \leq n \leq N, -D \leq d \leq D$$

Applying each feature to Equation 4 yields the training data in Equation 5. This step simplifies matters greatly. We have reduced the problem of mapping images and pixel locations to the target value $f_n^*(I, \mathbf{x})$, to mapping a list of feature values to the target value.

4.5 Learning the Calculation Rules

The local objective function f_n^ℓ maps the feature values to

the value of f_n^* . Machine learning infers this mapping from the training data in Equation 5. Our proof-of-concept uses model trees [4] for this task, which are a generalization of decision trees. Whereas decision trees have nominal values at their leaf nodes, model trees have line segments, allowing them to map features to a continuous value, such as the value returned by f_n^* . These trees are learned by recursively partitioning the feature space. A linear function is then fitted to the training data in each partition using linear regression. One of the advantages of model trees is that they tend to use only features that are relevant to predict the target values. Currently, we are providing $A=525$ image features, as illustrated in Figure 5. The model tree selects around 20 of them for learning the calculation rules.

After these five steps, a local objective function is learned for each contour point. It can now be evaluated at an arbitrary pixel \boldsymbol{x} of an arbitrary image I .

5. Experimental Evaluation

This section evaluates learned objective functions in the context of face model fitting. Thereby, we gather 500 images of frontal faces from the Internet.

5.1 Visualization of Global Objective Functions

Figure 6 visualizes how the value of the global objective function depends on varying pairs of elements of the parameter vector \boldsymbol{p} . The deformation parameter b_1 determines the angle at which the face model is viewed, and b_2 opens and closes the mouth of the model. As proposed by Cootes et al. [2] the deformation parameters vary from -2σ to 2σ of the deviation within the examples used for training the deformable model. It is clearly visible that the learned global objective function is closer to be ideal than the edge-based function. The plateaus with many local minima arise because they are outside of the area on which the objective function was trained. In these areas, the objective function cannot be expected to be ideal.

5.2 Comparison to State-of-the-art

In a further experiment, we compare our approach to a state-of-the-art model fitting approach using the BioID database [5]. Figure 7 shows the result of our fitting algorithm using a learned objective function (solid line). We determine the point-to-point distance between the results of the fitted models and the annotated models. Figure 7 visualizes the result of our experiment. The x -axis indicates the point-to-point distance measure between the manually specified models and the results of the fitting step and the y -axis indicates their cumulative percentage. Model fitting using our learned objective function (solid curve) improves global face localization (dashed line). 95% of all faces are fitted within a distance measure of 0.12 by applying the learning

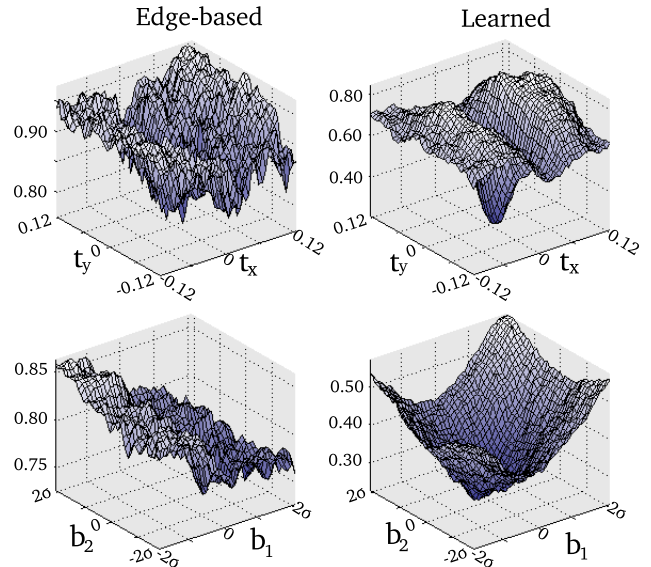


Fig. 6 Comparing the behavior of the edge-based to the learned global objective function, by varying pairs of parameters

approach. Applying only global face localization the distance measure for locating 95% of the faces is 0.16. That corresponds to an up to 30% higher deviation from the annotated model parameters. The set-up of this experiment equals to the one of [6] in terms of the utilized image database and the format of the results. Their approach conducts template matching in order to determine facial feature points. The quality of our results is comparable to those of [6].

6. Summary and Outlook

In this paper, we formalize the properties of ideal objective functions and give a concrete example of such functions. We show that designed objective functions are far from ideal. Therefore, we have developed a novel method that learns objective functions from annotated example images. The resulting objective functions are more accurate, because automated learning algorithms select relevant features from the many features provided and customize each local objective functions to local image conditions. Since many images are used for training, the learned objective function generalizes well. Using a publicly available image database, we verify that learned objective functions enable fitting algorithms to accurately determine the best fit. This approach automates many critical decisions and the remaining manual steps require less domain-dependent knowledge. Non-computer vision experts are able to use this methodology and therefore, it has the potential for commercialization.

Ongoing research applies our methods to model tracking. Since tracking algorithms exploit available knowledge from the current image of a sequence to bias search in the next image, they perform faster and more accurate than algorithms

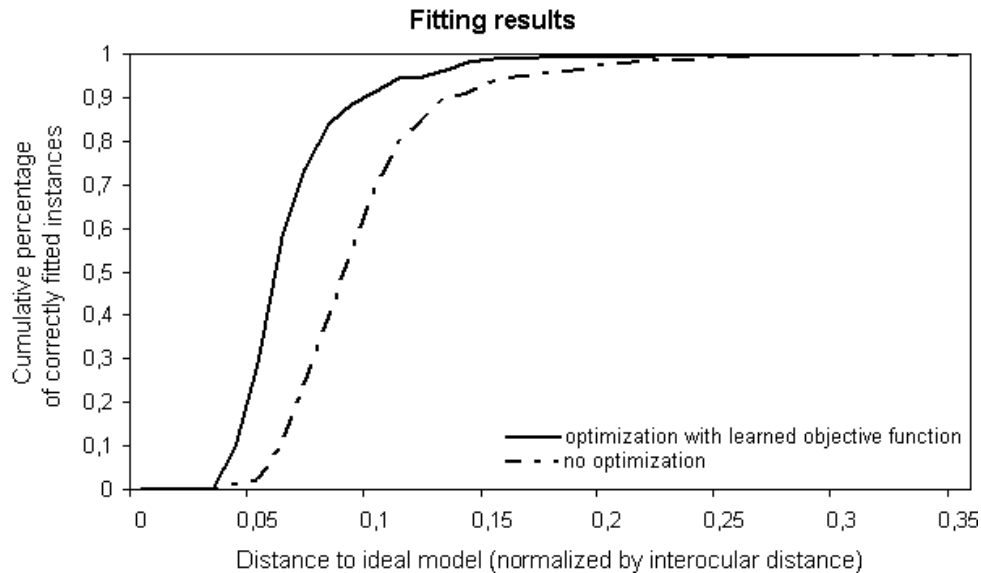


Fig. 7 The dashed line indicates the initial position of the face model as it is obtained from a rough localization. The solid line illustrates the same results fitting the model via the learned objective function.

for fitting a model to single images. Furthermore we explore the benefit of our approach to three-dimensional models.

References

- [1] Hanek, R.: Fitting Parametric Curve Models to Images Using Local Self-adapting Separation Criteria. PhD thesis, Department of Informatics, Technische Universität München (2004)
- [2] Cootes, T.F., Taylor, C.J.: Statistical models of appearance for computer vision. Technical report, University of Manchester, Wolfson Image Analysis, Imaging Science and Biomedical Engineering, Manchester M13 9PT, UK (2004)
- [3] Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In CVPR. (2001)
- [4] Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
- [5] Jesorsky, O., Kirchberg, K.J., Frischholz, R.: Robust face detection using the hausdorff distance. In: 3rd Int. Conf. on Audio- and Video-Based Biometric Person Authentication, Halmstad, Sweden, Springer-Verlag (2001) 90–95
- [6] Cristinacce, D., Cootes, T.F.: Facial feature detection and tracking with automatic template selection. In: 7th IEEE International Conference on Automatic Face and Gesture Recognition, Southhampton, England. (2006) 429–434