Learning Robust Objective Functions with Application to Face Model Fitting

Matthias Wimmer[1], Sylvia Pietzsch[2], Freek Stulp[3], and Bernd Radig[2]

[1] Faculty of Science and Engineering, Waseda University, Tokyo, Japan
[2] Institut für Informatik, Technische Universität München, Germany
[3] Kognitive Neuroinformatik, Universität Bremen, Germany

Abstract. Model-based image interpretation extracts high-level information from images using a priori knowledge about the object of interest. The computational challenge is to determine the model parameters that best match a given image by searching for the global optimum of the involved objective function. Unfortunately, this function is usually designed manually, based on implicit and domain-dependent knowledge, which prevents the fitting task from yielding accurate results.
In this paper, we demonstrate how to improve model fitting by learning objective functions from annotated training images. Our approach automates many critical decisions and the remaining manual steps hardly require domain-dependent knowledge. This yields more robust objective functions that are able to achieve the accurate model fit. Our evaluation uses a publicly available image database and compares the obtained results to a recent state-of-the-art approach.

1 Introduction

Model-based image interpretation systems exploit a priori knowledge about objects, such as shape or texture. The model contains a parameter vector $\boldsymbol{p}$ that represents its configuration, including position, rotation, scaling, and deformation. These parameters are usually mapped to the surface of an image, via a set of feature points, a contour, or a textured region.

Model fitting is the computational challenge of finding the model parameters that describe the content of the image best [1]. This task consists of two components: the fitting algorithm and the objective function. The *objective function* yields a comparable value that determines how accurately a parameterized model fits to an image. 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 $\boldsymbol{p}$ 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. In contrast, the objective function is usually determined ad hoc and heuristically, using the designer's intuitions about a good measure of

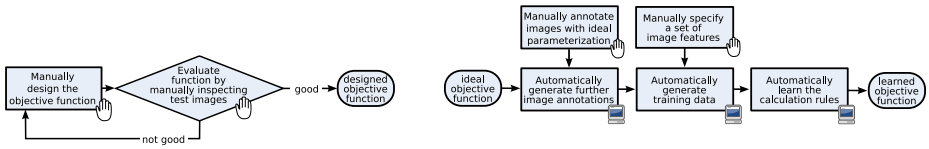

Fig. 1. The traditional procedure for designing objective functions (left), and the proposed method for learning objective functions from annotated training images (right).

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

Solution Idea. In contrast, this paper explicitly formulates the properties of *ideal* objective functions and gives a concrete example of such a function based on manual image annotations. Unfortunately, it is impossible to obtain ideal objective functions for real-world scenarios. Therefore, we propose to learn the objective function from comprehensive training data specified by the ideal objective function. This methodology approximates the ideal objective function and therefore achieves high accuracy. It automates most steps and the remaining manual steps require little domain-dependent knowledge, see Figure 1 (right). Furthermore, the *design-inspect* loop is eliminated, which makes the time requirements predictable.

Section 2 describes the design approach and points out its shortcomings. Section 3 specifies properties of ideal objective functions. Section 4 explains the proposed approach in detail. Section 5 experimentally evaluates the obtained results. Section 6 refers to related work and Section 7 summarizes our contributions and suggests further work.

2 Manually Designing Objective Functions

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 $\boldsymbol{p}=(t_x, t_y, s, \theta, \boldsymbol{b})^T$ describes the translation t_x and t_y, the scaling s, the rotation θ and the deformation $\boldsymbol{b}$. The function $\boldsymbol{c}_n(\boldsymbol{p})$ computes the location of the n^{th} contour point with $1{\leq}n{\leq}N$.

Model-based image interpretation requires determining the model that fits best to the image. For this reason, the objective function $f(I, \boldsymbol{p})$ computes the fitness between the model parameters $\boldsymbol{p}$ and the image I. According to common approaches [2], we split the objective function into N local parts $f_n(I, \boldsymbol{x})$, one for each contour point $\boldsymbol{c}_n(\boldsymbol{p})$. These local functions evaluate the image variations around the corresponding contour point and give evidence about its fitness.

Note, that the search on local objective functions $f_n(I, \boldsymbol{x})$ is conducted in pixel space $\boldsymbol{x} \in \mathbb{R}^2$, whereas the search on global objective function $f(I, \boldsymbol{p})$ is conducted in parameter space $\boldsymbol{p} \in \mathbb{R}^P$ with $P = dim(\boldsymbol{p})$. The result of the global objective function is the sum of the local function values, as in Equation 1. From now on, we will concentrate on local objective functions f_n, and simply refer to them as objective functions.

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

Objective functions are usually designed by manually selecting salient features from the image and mathematically composing them. The feature selection and the mathematical composition are both based on the designer's intuition and implicit knowledge of the domain. In [3] 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 $0 \leq E(I, \boldsymbol{x}) \leq 1$ denotes the edge magnitude.

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

As illustrated with the example in Figure 2, the design approach has comprehensive shortcomings and unexpected side-effects. 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 $E(I, \boldsymbol{x})$. 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 The Properties of 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 $\boldsymbol{p}_I^\star$, which are defined to be the model parameters with the best fitness to a specific image I. Similarly, $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$ denote the ideal contour points.

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

$$\forall \boldsymbol{x} (\boldsymbol{c}_n(\boldsymbol{p}_I^\star) \neq \boldsymbol{x}) \quad \Rightarrow \quad f_n(I, \boldsymbol{c}_n(\boldsymbol{p}_I^\star)) < f_n(I, \boldsymbol{x})$$

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

$$\exists \boldsymbol{m} \forall \boldsymbol{x} \ (\boldsymbol{m} \neq \boldsymbol{x}) \quad \Rightarrow \quad f_n(I, \boldsymbol{m}) < f_n(I, \boldsymbol{x}) \quad \wedge \quad \nabla f_n(I, \boldsymbol{x}) \neq \boldsymbol{0}$$

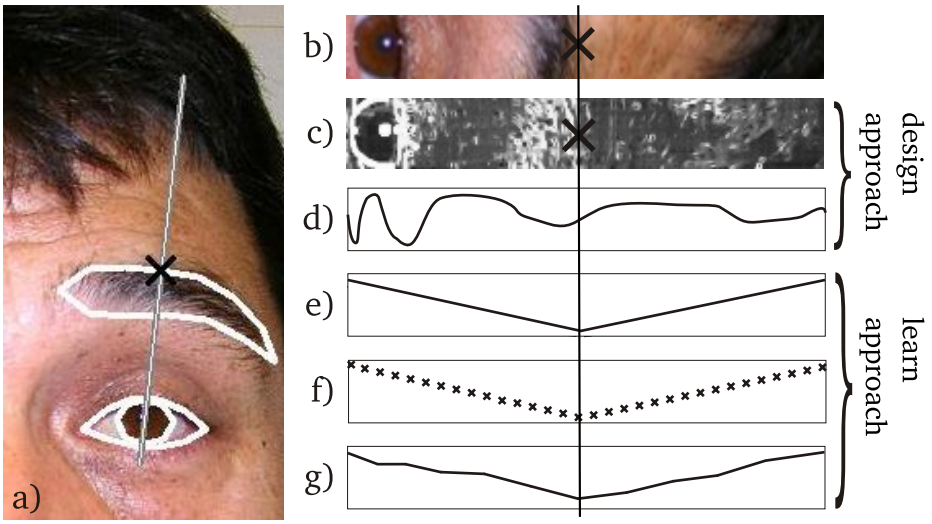

Fig. 2. a) Contour point with perpendicular, b) Image data, c) Edge magnitudes, d) Designed objective function f_n^e, e) Ideal objective function $f_n^\star$, f) Training data, g) Learned objective function f_n^ℓ; Note, b) – g) are taken along that perpendicular visible in a). The vertical line represents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

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. Thereby, the global minimum $\boldsymbol{m}$ does not need to correspond to the best fit. This is only required by the independent property P1.

$$f_n^\star(I, \boldsymbol{x}) = |\boldsymbol{x} - \boldsymbol{c}_n(\boldsymbol{p}_I^\star)| \tag{3}$$

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

4 Learning Robust Objective Functions

This section explains the five steps of our approach that learns objective functions from annotated training images, see Figure 1 (right). The key idea behind the approach is that $f_n^\star$ has the properties P1 and P2, and it generates the training data for learning an objective function $f_n^\ell(I, \boldsymbol{x})$. Therefore, this learned function will also approximately have these properties. Since it is "approximately ideal" we will refer to it as a *robust* objective function.

4.1 Annotating Images with Ideal Model Parameters

We manually annotate a set of images I_k with $1 \leq k \leq K$ with the ideal model parameters $\boldsymbol{p}^\star_{I_k}$. These parameters help to compute the ideal objective function $f^\star_n$ in Equation 3. This annotation is the only laborious step in the entire procedure of the proposed approach, whereat the time need remains predictable. An experienced human needs about one minute to determine the ideal parameters of our face model for one image. Figure 3 shows four images that are annotated with the ideal parameters of our face model. Note that for synthetic images, $\boldsymbol{p}^\star_I$ is known, and can be used in such cases. For real-world images, however, the ideal model parameters depend on the user's judgment.

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

4.2 Generating Further Image Annotations

According to P1, the ideal objective function returns the minimum $f^\star_n(I, \boldsymbol{x}) = 0$ for all image annotations. This data is not sufficient to learn f^ℓ_n, because training data must also contain image annotations, for which $f^\star_n(I, \boldsymbol{x}) \neq 0$. To acquire these annotations, $\boldsymbol{x}$ must be varied. General variations move $\boldsymbol{x}$ to any position within the image, however, it is more practicable to restrict this motion in terms of distance and direction.

Therefore, we generate a number of displacements $\boldsymbol{x}_{k,n,d}$ 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. Taking only these relocations facilitates the later learning step and improves the accuracy of the resulting calculation rules. This procedure is depicted in Figure 4. The center row depicts the manually annotated images, for which $f^\star_n(I, \boldsymbol{x}_{k,n,0}) = f^\star_n(I, \boldsymbol{c}_n(\boldsymbol{p}^\star_{I_k})) = 0$. The other columns depict the displacements $\boldsymbol{x}_{k,n,d\neq 0}$ with $f^\star_n(I, \boldsymbol{x}_{k,n,d\neq 0}) > 0$ as defined by P1. At these displacements values of $f^\star_n$ are obtained by applying Equation 3

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.

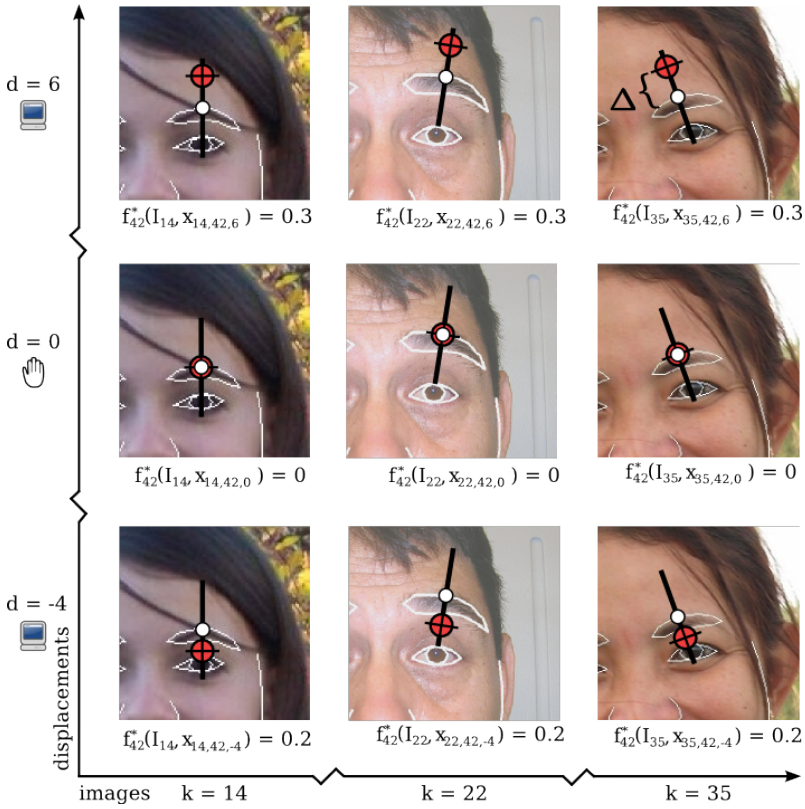

Fig. 4. In each of the K images each of the N contour points is annotated with $2D{+}1$ displacements. Manual work is only necessary for annotating $d{=}0$, which is depicted in the middle row. The other displacements are computed automatically. Note Δ in the upper right image that indicates the learning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

4.3 Specifying Image Features

Our approach learns a mapping from I_k and $\boldsymbol{x}_{k,n,d}$ to $f_n^\star(I_k,\ \boldsymbol{x}_{k,n,d})$, which is called $f_n^\ell(I, \boldsymbol{x})$. Since f_n^ℓ has no access to $\boldsymbol{p}_I^\star$, it must compute its value from the content of the image. Instead of learning a direct mapping from $\boldsymbol{x}$ and I to $f_n^\star$, we use a feature-extraction method [1]. The idea is to provide a multitude of image features, and let the 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 [4], see Figure 5, which greatly cope with noisy images. These features are not only computed at the location of the contour point itself, but also at locations within its vicinity specified by a grid, see Figure 5. This variety of image features enables the objective function to exploit the texture of the image at the model's contour point and in its surrounding area.

Fig. 5. The set of A=6·3·5·5=450 features utilized for learning the objective functions.

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 $f_n^\star$ yields the list in Equation 4.

$$[\quad I_k, \quad \boldsymbol{x}_{k,n,d} \quad , f_n^\star(I_k,\ \boldsymbol{x}_{k,n,d}) \ ] \tag{4}$$

$$[\ \boldsymbol{h}_1(I_k,\ \boldsymbol{x}_{k,n,d}), \ldots, \boldsymbol{h}_A(I_k,\ \boldsymbol{x}_{k,n,d})\ , f_n^\star(I_k,\ \boldsymbol{x}_{k,n,d}) \ ] \tag{5}$$

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

We denote image features by $\boldsymbol{h}_a(I, \boldsymbol{x})$, with $1{\leq}a{\leq}A$. Each of these features returns a scalar value. 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 $f_n^\star(I, \boldsymbol{x})$, 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 values of the features to the value of $f_n^\star$. This mapping is learned from the training data by learning a model tree [5]. Model trees are a generalization of decision trees. Whereas decision trees have nominal values at their leaf nodes, model trees have line segments, allowing them to also map features to a continuous value, such as the value returned by $f_n^\star$. They 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=450 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 called with 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 parameters from 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. [3] 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 a State-of-the-art Approach

In a further experiment, we compare our approach to a state-of-the-art model fitting approach using the BioID database [6]. 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

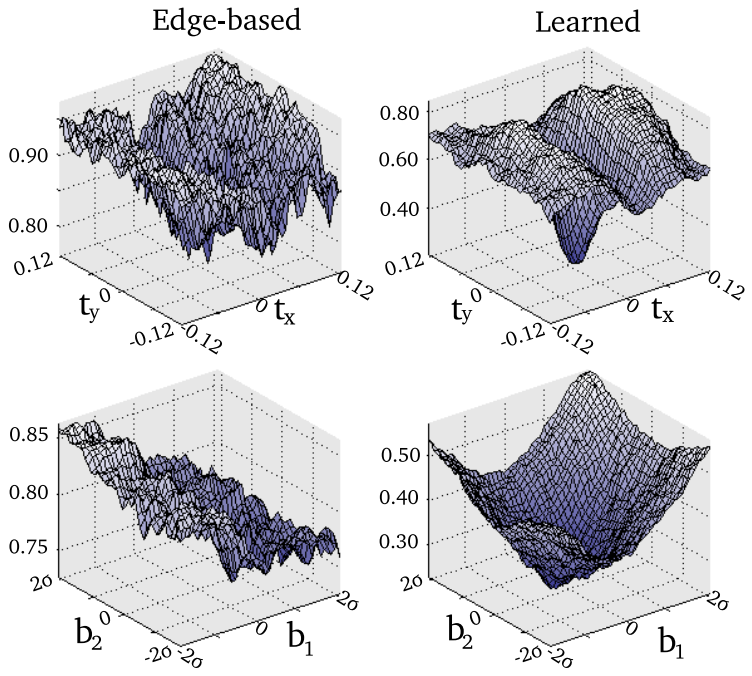

Fig. 6. Comparing the behavior of the edge-based (left column) to the learned (right column) global objective function, by varying pairs of parameters.

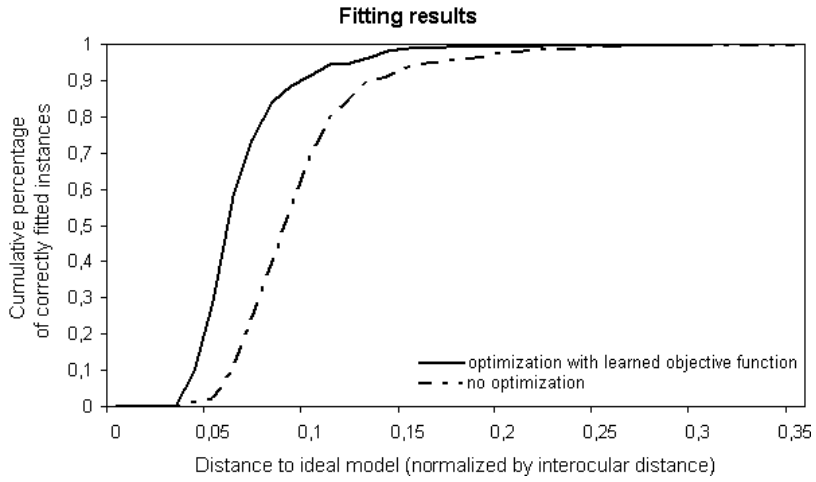

Fig. 7. The initial position of the face model (dashed line) is highly improved by fitting it with a learned objective function (solid line).

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 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 is directly comparable to the one of [7] in terms of the utilized image database and the format of the obtained results. Their approach conducts template matching in order to determine facial feature points. The quality of our results is comparable to those of [7], who achieved the fitting of 90% of the faces within a distance measure of 0.075 and 96% within a distance measure of 0.15. In our experiment 90% of all faces are fitted within a distance measure of 0.09 and the distance measure for fitting 96% is 0.13.

6 Related Work

The insights and the approach of Ginneken et al. [8] are most comparable to our work. They consider objective functions to be ideal if they fulfill properties similar to P1 and P2. Annotated training images serve for learning local objective functions. Their approach also determines relevant image features from a set of image features. However, they do not learn the objective function from an ideal objective function but manually specify calculation rules. Therefore, their approach aims at obtaining Property P1 but does not consider Property P2. Furthermore, their approach turns out to be slow, which is a direct result from applying the k-Nearest-Neighbor classifier.

Currently, model fitting is often conducted using Active Appearance Models [2], which do not only contain the contour of the object but also the texture of the surface as it appears in the training images. The objective function is usually taken to be the sum of the square pixel errors between the synthesized texture of the model and the underlying image. Model fitting aims at minimizing this error by conducting a gradient decent approach. Obviously, this approach matches P1 very well. However, this approach does not consider P2 at all. Therefore, model fitting only achieves reasonable results within a small convergence area around the ideal model parameters.

7 Summary and Outlook

In this paper, we formalize the properties of ideal objective functions and give a concrete example of such functions. Since designed objective functions are far from ideal. Therefore, we have developed a novel method that learns objective functions from annotated example images. This approach automates many critical decisions and the remaining manual steps require less domain-dependent knowledge. The resulting objective functions are more accurate, because automated learning algorithms select relevant features from the many features provided and customize each local objective function 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 robustly determine the best fit.

Ongoing research applies our method to tracking three-dimensional models through image sequences. They exploit knowledge from the current image to bias search in the next image, which makes them perform fast and accurately.

References

1. Hanek, R.: Fitting Parametric Curve Models to Images Using Local Self-adapting Seperation Criteria. PhD thesis, Technische Universität München (2004)
2. Cootes, T.F., Taylor, C.J.: Statistical models of appearance for computer vision. Technical report, University of Manchester, Manchester M13 9PT, UK (2004)
3. Cootes, T.F., Taylor, C.J.: Active shape models – smart snakes. In: Proc. of the 3rd British Machine Vision Conference 1992, Springer Verlag (1992) 266 – 275
4. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition (CVPR). (2001)
5. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2 edn. Morgan Kaufmann, San Francisco (2005)
6. Jesorsky, O., Kirchberg, K.J., Frischholz, R.: Robust face detection using the hausdorff distance. In: Proc. of the 3rd Int. Conference on Audio- and Video-Based Biometric Person Authentication, Halmstad, Sweden, Springer-Verlag (2001) 90–95
7. 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, Southampton, England. (2006) 429–434
8. Ginneken, B., Frangi, A., Staal, J., Haar, B., Viergever, R.: Active shape model segmentation with optimal features. Trans. on Medical Imaging **21** (2002) 924–933