Enabling Users to Guide the Design of Robust Model Fitting Algorithms

Matthias Wimmer*
Perceptual Computing Lab
Waseda University
Tokyo, Japan

Freek Stulp
Cognitive Neuroinformatics Group
University of Bremen
Bremen, Germany

Bernd Radig
Image Understanding Group
Technische Universität München
Garching, Germany

Abstract

Model-based image interpretation extracts high-level information from images using a priori knowledge about the object of interest. The computational challenge in model fitting is to determine the model parameters that best match a given image, which corresponds to finding the global optimum of the objective function.

When it comes to the robustness and accuracy of fitting models to specific images, humans still outperform state-of-the-art model fitting systems. Therefore, we propose a method in which non-experts can guide the process of designing model fitting algorithms. In particular, this paper demonstrates how to obtain robust objective functions for face model fitting applications, by learning their calculation rules from example images annotated by humans. We evaluate the obtained function using a publicly available image database and compare it to a recent state-of-the-art approach in terms of accuracy.

1. Introduction

Model-based image interpretation has proven to be appropriate to infer high-level scene descriptors from the content of images [4, 12, 9, 6]. These 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 [7]. This task consists of two components: (1) The *objective function* returns a 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. (2) The *fitting algorithm* searches for the model parameters $\boldsymbol{p}$ that optimize the objective function, i.e. they try to find the global minimum or maximum, depending on the definition of 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 [7] 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. The designer manually selects a small set of image features that he considers to be appropriate and mathematically composes the calculation rules from them. Afterwards, he visually inspects the appropriateness of the objective function by evaluating it on a few example images. If the result is not satisfactory the function is tuned or redesigned from scratch. 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. The calculation rules cannot be specified appropriately enough and therefore, the best model fit is not objectively determined.

Solution Idea. In contrast, it is very easy for humans to determine the model parameters that best match a given image. Therefore, we propose to learn the objective function from these preferred model parameters, specified by (non-expert) users. Furthermore, this paper investigates the properties of objective functions and explicitly formulates

*This research is partly funded by a JSPS Postdoctoral Fellowship for North American and European Researchers (FY2007).

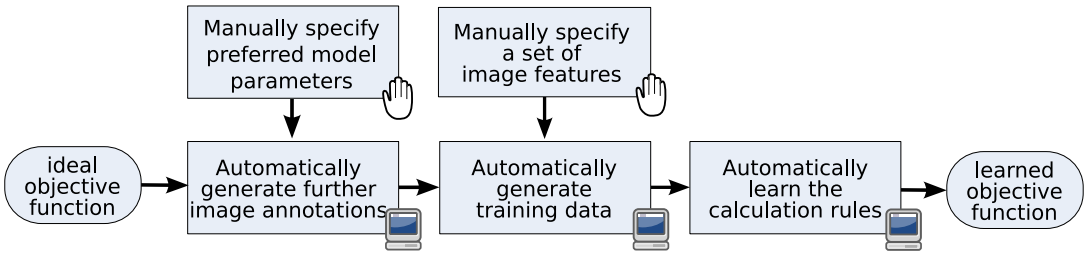

Figure 2. The proposed methodology requires users to annotate example images with the preferred model parameters. Further steps are automated. This proposed procedure yields highly accurate objective functions.

the properties of *ideal* objective functions. We also give a concrete example of such a function, which it is based on image annotations. The proposed methodology approximates the ideal objective function and therefore, the learned objective functions are approximately ideal. It automates most steps and the remaining manual step of annotating images requires little domain-dependent knowledge, see Figure 2. Furthermore, the *design-inspect* loop is eliminated, which makes the time requirements predictable. Also, by annotating only a particular class of images (e.g. bearded faces), the user determines on which class of images the learning algorithm will be specialized. This approach also yields more accurate objective functions, because the selection of image features is based on objective relevance measures.

The remainder of this paper is organized as follows. 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. 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 $1{\leq}n{\leq}N$.

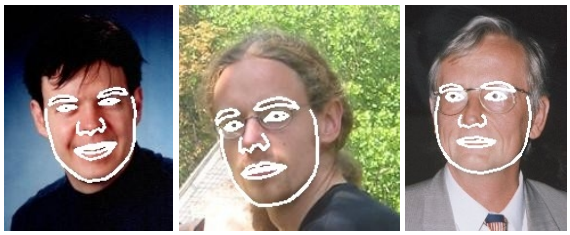

Figure 1. Specifying the preferred parameters of the face model for a set of images represents the most important part of our approach. The creation of the calculation rules is guided by this information.

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, 5, 10, 11], 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 minimization of local objective functions $f_n(I, \boldsymbol{x})$ is conducted in pixel space $\boldsymbol{x}{\in}\mathbb{R}^2$, whereas it is conducted in parameter space $\boldsymbol{p}{\in}\mathbb{R}^P$ for global objective function $f(I, \boldsymbol{p})$ 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 [1] 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 3, the design approach has comprehensive shortcomings and unexpected side-effects. 3a) visualizes one of the contour points of the face model as well as its perpendicular towards the contour. 3b) and 3c) depict the content of the image along this perpendicular as well as the corresponding edge magnitudes $E(I, \boldsymbol{x})$. 3d) 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

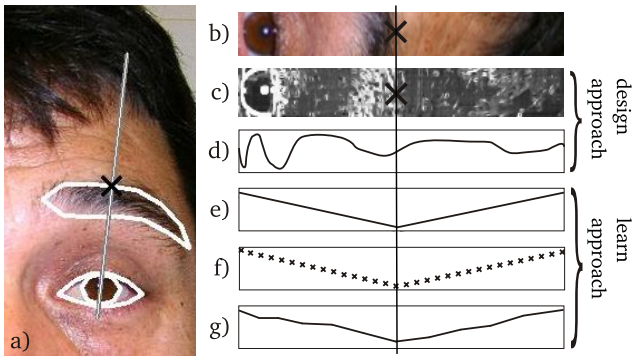

Figure 3. a) Contour point with perpendicular, b) Image data, c) Edge magnitudes, d) Designed objective function f_n^e, e) Ideal objective function derived from human guidance $f_n^\star$, f) Training data, g) Learned objective function f_n^ℓ; Note, b) – g) are taken along that perpendicular visible in a). The vertical line represents the location of the preferred contour point $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$.

not correspond to the preferred 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 preferred location.

3. The Properties of Ideal Objective Functions

This section makes the observations from Figure 3 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 *preferred* 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 preferred contour points. The aim of our approach is to determine $\boldsymbol{p}_I^\star$ manually in order to obtain highly appropriate calculation rules. The image in Figure 3a) is annotated with $\boldsymbol{p}_I^\star$.

P1: 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: There is no local minimum or maximum.

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

Property P1 relates to the *correctness* of the local objective function. Fitting algorithms search for its global minimum and P1 ensures that the search result corresponds to the best fit of the contour point. Although it might seem obvious that this is a desirable property for objective functions to have, designing them does not always guarantee that this is the case, such as in the example in Section 2.

Property P2 guarantees that any determined minimum represents the global minimum. This facilitates search, because algorithms can not get stuck in a local minimum. Simple local minimization strategies suffice to find the global minimum. The mathematical formalization states that all locations $\boldsymbol{x}$ that are not the global minimum $\boldsymbol{m}$ are not allowed to have a zero gradient, and are therefore not minima. Note that the global minimum $\boldsymbol{m}$ does not need to correspond with the best fit; this is only required by the independent property P1.

3.1. An Ideal Objective Function

We now introduce a concrete instance of an ideal objective function, which is denoted with $f_n^\star(I, \boldsymbol{x})$, see Equation 3. It computes the distance between the preferred contour point $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$ and a pixel $\boldsymbol{x}$ located on the image.

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

A significant feature of $f_n^\star$ is that it relies on the preferred model parameters $\boldsymbol{p}_I^\star$, which are specified by user knowledge, to compute its value. This implies that ideal objective functions cannot be obtained without human interaction. Furthermore, it implies that the ideal objective function $f_n^\star$ cannot be evaluated for previously unseen images using automated model fitting algorithms, because the preferred model parameters $\boldsymbol{p}_I^\star$ are not known for these images.

4. Learning Robust Objective Functions from Human Knowledge

This section explains the five steps of the proposed approach that learns objective functions from training images, see Figure 2. The key idea behind the approach is that $f_n^\star$, which is obtained by user knowledge, has the properties P1 and P2, and it generates the training data for learning another objective function $f_n^\ell(I, \boldsymbol{x})$. Therefore, this learned function will also approximately have these properties.

4.1. Manually Annotating Images

First, humans manually annotate a set of images I_k with the preferred model parameters $\boldsymbol{p}_{I_k}^\star$ with $1 \leq k \leq K$. This step cannot be accomplished automatically, because it requires an amount of human knowledge that has not yet been achieved with computer algorithms. If it had already been achieved, there would be no need for further sophisticated model fitting algorithms. All further steps rely on these annotations, because they allow to compute the ideal objective function $f_n^\star$, see Equation 3. This step is the only laborious part of the entire procedure of the proposed approach. An experienced human needs about one minute to determine

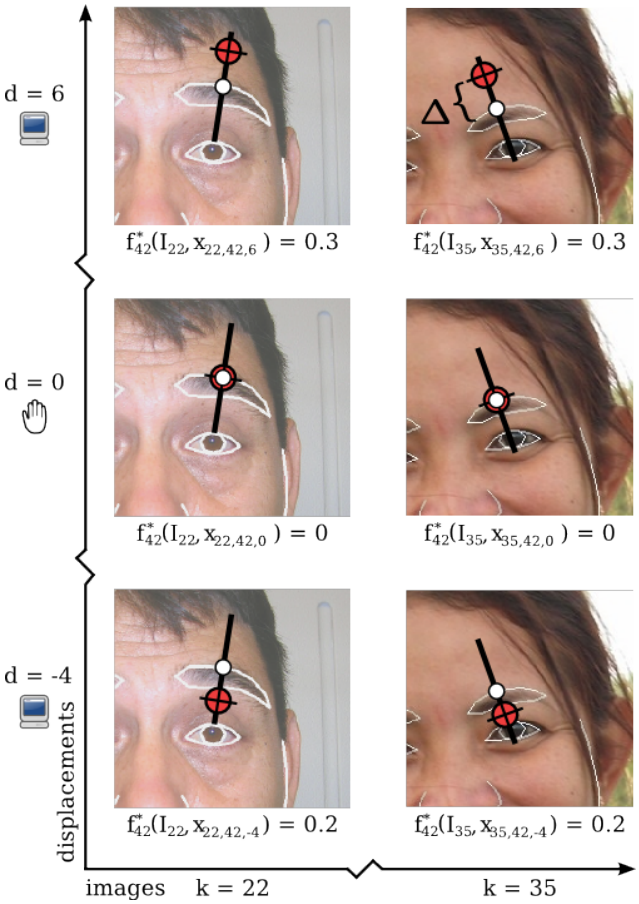

Figure 4. In each of the K images, each of the N contour points is annotated with $2D{+}1$ displacements. Manual annotation is only necessary for $d{=}0$ (middle row). The other displacements are computed automatically. The upper right image shows the learning radius Δ. The unit of the ideal objective function values and Δ is the interocular measure.

the preferred parameters of our face model for one image. Therefore, the time requirements remain predictable. Figure 1 shows three images that are annotated with the preferred parameters of our face model.

4.2. Generating Further Image Annotations

According to P1, the ideal objective function returns the minimum $f_n^\star(I,\boldsymbol{x}){=}0$ for all image annotations. This data is not sufficient to learn f_n^ℓ, because training data must also contain image annotations, for which $f_n^\star(I,\boldsymbol{x}){\neq}0$. 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 $f_n^\star(I,\boldsymbol{x}_{k,n,0}){=}f_n^\star(I,\boldsymbol{c}_n(\boldsymbol{p}_{I_k}^\star)){=}0$. The other columns depict the displacements $\boldsymbol{x}_{k,n,d\neq0}$ with $f_n^\star(I,\boldsymbol{x}_{k,n,d\neq0}){>}0$ as defined by P1. At these displacements the values of $f_n^\star$ 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.

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 [7]. 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 [13], 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.

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.

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

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

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

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.

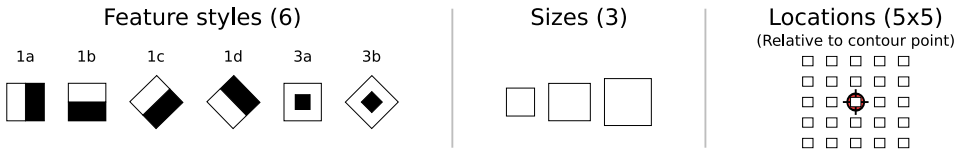

Figure 5. The set of $A = 6 \cdot 3 \cdot 5 \cdot 5 = 450$ features utilized for learning the objective functions.

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 [14]. 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 evaluated at an arbitrary pixel $\boldsymbol{x}$ of an arbitrary image I.

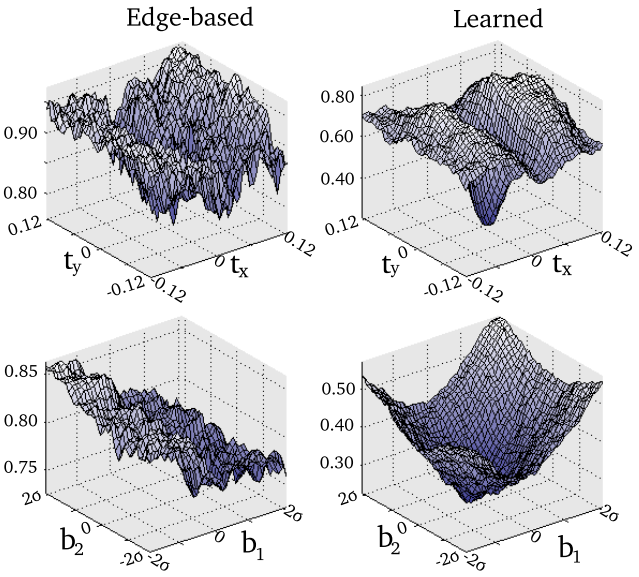

Figure 6. The behavior of the learned objective function varying pairs of parameters; the translation t_x and t_y and two deformation parameters b_1 and b_2.

5. Experimental Evaluation

This section evaluates our approach in order to prove its appropriateness and accuracy using a publicly available image database for comparison purpose.

Figure 6 visualizes how the value of the global objective function depends on specific pairs of model parameters, such as the translation t_x and t_y and the deformation parameters b_1 and b_2, where b_1 determines the rotation angle of the face model, and b_2 opens and closes its mouth. As proposed by Cootes et al. [1] we vary the deformation parameters between -2σ and 2σ of the deviation of the examples used for training the model. As expected, the learned global objective function f^ℓ is closer to be ideal than the designed edge-based approach f^e. Its plateaus with many local minima arise from the fact that they are outside of the learning radius Δ. In these areas, the result of the function is undefined.

In a further experiment, we evaluate our approach on the BioID database [8]. Figure 7 shows the result of model fitting with learned objective functions (solid line). The x-axis indicates the point-to-point distance between the fitted model and the manually specified model and the y-axis indicates the cumulative percentage of the distances. Model fitting with learned objective functions greatly improves the initial face localization (dashed line) that is conducted with the approach of Viola et al. [13]. 95% of all faces are fitted within a distance measure of 0.12 by applying the learning approach. Applying only face localization the distance measure for locating 95% of the faces is 0.16. That corre-

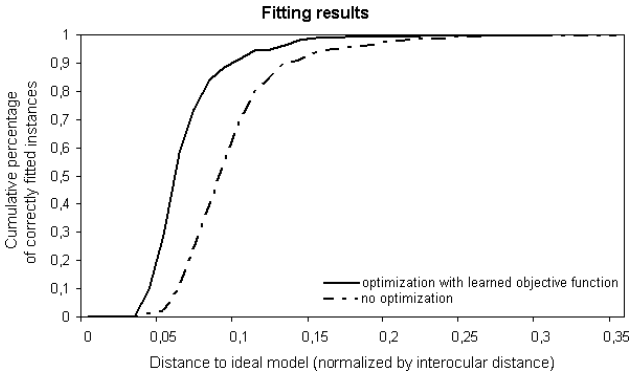

Figure 7. The initial position of the face model (dashed line) is highly improved by fitting it with a learned objective function (solid line).

sponds to an up to 30% higher deviation from the annotated model parameters. The set-up of this experiment is similar to the one of [3] w.r.t. the utilized images and the obtained results. Their approach conducts template matching in order to determine facial feature points. They achieved the fitting of 90% of the face models within a distance of 0.075 and 96% within a distance of 0.15.

6. Related Work

The approach of Ginneken et al. [5] is most similar to our work. They consider objective functions to be ideal if they fulfill properties similar to P1 and P2. Also, training images annotated by humans 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 preferred model parameters.

7. Summary and Outlook

Accurate objective functions are essential for accurate model fitting. However, their calculation rules are usually designed by hand and therefore far from ideal. In this paper, we propose to learn robust objective functions from examples provided by humans. First, we formalize the properties of ideal objective functions and give a concrete example of such a function. This ideal objective function requires human interaction to annotate a set of example image with the preferred model parameters. Then, we learn a new objective function from these image annotations. The resulting objective function is 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.

For evaluation, we compare this approach to a state-of-the-art approach. Using a publicly available image database, we verify that learned objective functions enable fitting algorithms to robustly determine the best fit.

In our ongoing research, we are applying our human-assisted approach to three-dimensional face models, as well as to face tracking in image sequences.

References

[1] T. F. Cootes and C. J. Taylor. Active shape models – smart snakes. In *Proc. of the 3rd British Machine Vision Conference 1992*, pages 266 – 275. Springer Verlag, 1992.

[2] T. F. Cootes and C. J. Taylor. Statistical models of appearance for computer vision. Technical report, U of Manchester, Wolfson Image Analysis, Imaging Science and Biomedical Eng., Manchester M13 9PT, UK, 2004.

[3] D. Cristinacce and T. F. Cootes. Facial feature detection and tracking with automatic template selection. In *7th IEEE International Conference on Automatic Face and Gesture Recognition, Southampton, UK*, pages 429–434, 2006.

[4] G. J. Edwards, T. F. Cootes, and C. J. Taylor. Face recognition using active appearance models. In H. Burkhardt and B. Neumann, editors, *5th European Conference on Computer Vision*, volume LNCS-Series 1406–1607, pages 581–595, Freiburg, Germany, 1998. Springer-Verlag.

[5] B. Ginneken, A. Frangi, J. Staal, B. Haar, and R. Viergever. Active shape model segmentation with optimal features. *IEEE Transactions on Medical Imaging*, 21(8):924–933, 2002.

[6] D. Grest, D. Herzog, and R. Koch. Human model fitting from monocular posture images. In *Proc. of VMV 2005*, Erlangen, Germany, November 2005.

[7] R. Hanek. *Fitting Parametric Curve Models to Images Using Local Self-adapting Seperation Criteria*. PhD thesis, Department of Informatics, Technische Universität München, 2004.

[8] O. Jesorsky, K. J. Kirchberg, and R. Frischholz. Robust face detection using the hausdorff distance. In *AVBPA '01: Proceedings of the Third International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 90–95, Halmstad, Sweden, 2001. Springer-Verlag.

[9] V. Lepetit and P. Fua. Monocular model-based 3D tracking of rigid objects. *Found. Trends. Comput. Graph. Vis.*, 1(1):1–89, 2006.

[10] S. Romdhani. *Face Image Analysis using a Multiple Feature Fitting Strategy*. PhD thesis, University of Basel, Computer Science Department, Basel, CH, 1 2005.

[11] S. Romdhani, P. Torr, B. Scholkopf, and A. Blake. Computationally efficient face detection. In *Eighth IEEE International Conference on Computer Vision*, volume 2, pages 695–700, 2001.

[12] M. B. Stegmann and K. Skoglund. On automating and standardising corpus callosum analysis in brain MRI. In *Proceedings Svenska Symposium i Bildanalys, SSBA 2005, Malmø, Sweden*, pages 1–4. SSBA, mar 2005.

[13] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[14] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.