Simultaneous learning of motion and appearance

Karel Zimmermann[1,2] , Tomáš Svoboda[1], Jiří Matas[1]

[1]: Czech Technical University
FEL/CMP
Prague, Czech republic
svoboda@cmp.felk.cvut.cz

[2] : Katholieke Universiteit Leuven
ESAT/PSI/Visics
Leuven, Belgium
kzimmerm@esat.kuleuven.be

Abstract. A new learning method for motion estimation of objects with significantly varying appearance is proposed. Varying object appearance is represented by a low dimensional space of appearance parameters. The appearance mapping and motion estimation method are optimized simultaneously. Appearance parameters are estimated by unsupervised learning. The method is experimentally verified by a tracking application on sequences which exhibit strong variable illumination, non-rigid deformations and self-occlusions.

1 Introduction

Visual tracking is often formulated as iterative motion estimation with possible updates of the object appearance. The optional online appearance update may allow for longer tracks but also makes the procedure prone to failure. We propose an algorithm that learns both motion prediction and appearance changes from training data.

The most natural approach to tracking is alignment of a *template* image $\mathbf{J}$ with image data $\mathbf{I}$ by an online optimization of a criterion function like the sum of square errors [1] $\|\mathbf{I} - \mathbf{J}\|^2$ or mutual information [2], see Figure 1a. Since the tracker has usually no prior information about the possible changes of the object appearance, the template is either not updated at all or updated from the last known position in terms of partial or whole template replacement by the aligned image [2–4]. However, such *hard* appearance update often results in drifting and consequently to the loss-of-lock.

If the changes of the object appearance could be explained by a reasonably small number of parameters $\boldsymbol{\theta}$, e.g. affine transformation [5], than the template is updated by another online optimization, which searches for the best template warp, see for example Figure 1b. If such appearance parameterization describes well all possible appearance variations and the motions are slow enough to initialize the optimization within the basin of attraction, the system works usually well. However, the appearance changes are often caused by a non-rigid deformation or non-trivial illumination change, which do not allow a simple parameterization.

Facing these problems many authors [6–10] propose a learning stage, where the motions and/or possible object appearances are learned. Jurie and Dhome [10] suggest to learn a linear mapping between observed image intensities and corresponding motion $\mathbf{t}$. During the learning stage a training image is perturbed by random motion parameters and the least square method estimates coefficients $\mathtt{H}$ of the searched linear mapping. During the tracking stage, the motion is estimated as the linear function of the difference

image $(\mathbf{I} - \mathbf{J})$ between the image data and template, see Figure 1c. Jurie's approach avoids the problems of the basin of attraction, but it allows only the hard template update.

Cootes et al. [8] realized that once a training set is available, it is reasonable to use it also for the learning of the appearance. They proposed a paradigm where both the motion and appearance are projected by the PCA [11] to the lower dimensional space of some parameters. Given an image the learned linear mapping with coefficients $\mathtt{H}$ estimates the parameters which are used for both the template update and motion estimation by PCA back-projection, see Figure 1d.

Observing that

- the mapping between input image $\mathbf{I}$ and the output motion parameters $\mathbf{t}$ is linear, see dashed line in Figure 1d denoting the direct route, and
- the mapping between input image $\mathbf{I}$ and appearance parameters $\boldsymbol{\theta}$ is also linear,

we generalized the tracking approach to Figure 1e. While the *tracker* φ aligns the model with the current image, the *appearance encoder* γ encodes a current appearance into parameters $\boldsymbol{\theta}$, which adjust the tracker for the current object appearance. Since the learning process estimates both mappings simultaneously, the learned appearance encoder γ projects, in contrast to the PCA, the current object appearance to a manifold, the coordinates of which are the most convenient for the tracker adjustment. In the rest we describe tracking and learning of the *tracking system* depicted in Figure 1e.

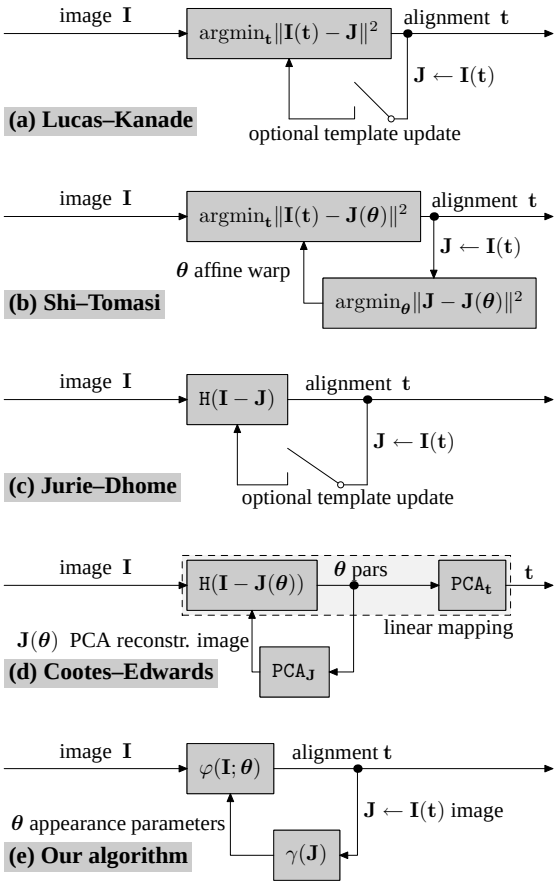


Fig. 1. State-of-the-art summary and the proposed approach.

Section 2 describes acquisition of a training set, i.e., the set of the examples consisting of the intensities incoming into the tracker, the motion parameters to be estimated by the tracker and the intensities incoming into the appearance encoder. In Section 3, we define the criterion function to be minimized as the squared Euclidean distance (i.e. L_2-norm). In Section 4 the learning of φ and γ minimizing the criterion is described. Roughly speaking, the learning is the least squares bilinear function fitting

problem, the minimum of which is searched by an exact line-search algorithm [12]. Section 5 presents experiments demonstrating an improvement to the state-of-the-art. The conclusions are in Section 6.

2 Training set construction

Let the object position is represented by a *reference point* (e.g., center of gravity of object pixels). Let us suppose we are given a ground truthed sequence of images, where the position of the object reference point is denoted. Given a predefined range of motions R within which the tracker is assumed to operate (e.g., radius of desired basin of attraction), the training set consists of:

- *tracker input* images $\mathbf{I}^i$, randomly (uniformly) generated within R,
- corresponding *tracker output* motion parameters $\mathbf{t}^i$ (e.g., translations),
- and *appearance encoder input* images $\mathbf{J}^i$, i.e., the images aligned by the tracker with a limited accuracy.

We perturb neighborhood of reference point in each particular frame of the sequence by motion parameters $\mathbf{t}^i$ randomly generated inside the range, creating the set of synthesized examples of intensities $\mathbf{I}^i$, see Figure 2. These examples are column-wise stored in matrices $\mathtt{I} = \left[\mathbf{I}^1 \dots \mathbf{I}^d\right]$ and $\mathtt{T} = \left[\mathbf{t}^1 \dots \mathbf{t}^d\right]$.

The alignment estimated by the tracker is never perfect. Let us assume for a moment, that the accuracy of the alignment by the tracker is known in advance[1]. We perturb neighborhood of reference points within the range determined by the tracker accuracy creating the set of images $\mathtt{J} = \left[\mathbf{J}^1 \dots \mathbf{J}^d\right]$ entering to the appearance encoder. Ordered triple $(\mathtt{I}, \mathtt{J}, \mathtt{T})$ of such matrices is called a *training set*.

3 Optimization problem definition

In this section the following notation is used:

Given a matrix $\mathtt{H}$:

- $\mathbf{h}_j^\top$ is row vector corresponding to j-th row,
- $\mathbf{h}^i$ is column vector denoting its i-th column,
- $h_j^i = [\mathtt{H}]_{i,j}$ denotes element at coordinates $[i, j]$,
- $\mathtt{H}^k$ is approximation of $\mathtt{H}$ in k-th iteration,
- $\|\mathtt{H}\|_F^2 = \sum_{i,j} |h_j^i|^2$ is squared Frobenius norm of $\mathtt{H}$,
- $\mathbf{1}_m = [1 \dots 1]^\top$ is $m \times 1$ unit vector,
- $\otimes$ is Kronecker product,
- $\bullet$ is entry-wise product, sometimes called Hadamard product.

[1] In practice, the accuracy has to be estimated by iterating the learning process. However, for the sake of simplicity, it is assumed to be known.

Fig. 2. Image is perturbed by the motion parameters within a predefined range in order to create a set of synthesized examples of observed intensities $\mathbf{I}^i$ and motions $\mathbf{t}^i$. Red dashed square denotes image data observed at translation $\mathbf{t} = (0,0)^\top$, blue square denotes image data observed at translation $\mathbf{t} = (15,-25)^\top$. Reference point is the tip of the nose. Different appearances are encoded by different appearance parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$.

Because of simplicity, we restrict φ and γ to be from a class of linear mappings. Note that *data lifting* makes the same method work for any mapping formed as the linear combination of arbitrary intensity functions. For example, arbitrary polynomial mapping is equivalent to the linear mapping on the embedded space of monomials. It means, that we just transform the training set to the space of monomials and the rest of the learning process is the same.

Definition 1. Learned Linear Predictor *(LLiP) is ordered pair* $(\mathtt{H}, X)$ *which computes motion parameters* $\mathbf{t}$ *from the vector of image intensities observed on the set of pixels* X *as follows:*

$$\mathbf{t} = \mathtt{H}\mathbf{I}(X).$$

We further refer to the set of pixels as to the support set *and the linear mapping is called* regressor.

Note that the support set selection problem is beyond the scope of this paper and we select it randomly.

We extend the LLiP to the form allowing parameterization by the appearance parameters, therefore Parameter Sensitive LLiP (PLLiP) is defined as follows:

Definition 2. Parameter sensitive Learned Linear Predictor *(PLLiP) is LLiP with regressor* $\mathtt{H} = (\mathtt{H}_0, \ldots, \mathtt{H}_m)$*, which computes motion parameters* $\mathbf{t}$ *from observed image* $\mathbf{I}$ *given current appearance parameters* $\boldsymbol{\theta} = (\theta_1 \ldots \theta_m)^\top$ *as follows*

$$\mathbf{t} = (\mathtt{H}_0 + \theta_1\mathtt{H}_1 + \cdots + \theta_m\mathtt{H}_m)\mathbf{I}, \tag{1}$$

Referring to Figure 1e, we define *appearance encoder* to be the LLiP with regressor

$$\boldsymbol{\theta} = \gamma(\mathbf{J}) = \mathtt{G}\mathbf{J}, \tag{2}$$

which encodes current object appearance from previously aligned image $\mathbf{J}$ into parameters $\boldsymbol{\theta}$.
Tracker is PLLiP with regressor

$$\mathbf{t} = \varphi(\mathbf{I}; \boldsymbol{\theta}) = (\mathtt{H}_0 + \theta_1 \mathtt{H}_1 + \cdots + \theta_m \mathtt{H}_m)\mathbf{I}$$

parameterized by the appearance parameters $\boldsymbol{\theta}$.

Definition 3. Tracking system *is ordered pair* $(\mathtt{H}, \mathtt{G})$ *corresponding to tracker (PLLiP) and appearance encoder (LLiP) connected according to Figure 1e.*

Definition 4. Prediction error $e(\mathtt{H}, \mathtt{G})$ *of tracking system* $(\mathtt{H}, \mathtt{G})$ *on training set* $(\mathtt{I}, \mathtt{J}, \mathtt{T})$ *is*

$$e(\mathtt{H}, \mathtt{G}) = \sum_{i=1}^{d} \left\| \Big(\mathtt{H}_0 + \underbrace{(\mathbf{g}_1^\top \mathbf{J}^i)}_{\theta_1} \mathtt{H}_1 + \cdots + \underbrace{(\mathbf{g}_m^\top \mathbf{J}^i)}_{\theta_m} \mathtt{H}_m \Big) \mathbf{I} - \mathbf{t}^i \right\|_2^2. \tag{3}$$

Definition 5. Optimal tracking system *with respect to training set* $(\mathtt{I}, \mathtt{J}, \mathtt{T})$ *is tracking system*

$$(\mathtt{H}^*, \mathtt{G}^*) = \operatorname*{argmin}_{\mathtt{H}, \mathtt{G}} e(\mathtt{H}, \mathtt{G}). \tag{4}$$

The matrices $\mathtt{H}, \mathtt{G}, \mathtt{I}, \mathtt{J}, \mathtt{T}$ have dimensions $(p(m+1) \times n), (m \times n), (n \times d), (n \times d), (n \times p)$, respectively, where

- p is the number of tracked motion parameters (e.g., 2D-translation $\Rightarrow p = 2$),
- n is the number of used pixels,
- m is the number of appearance parameters,
- d is number of training examples.

Computational complexity of the tracking procedure corresponds to the number of all elements in the matrices $(\mathtt{H}, \mathtt{G})$ since the motion in particular frame requires approximately such number of multiplications and additions.

Definition 6. Complexity $C(\mathtt{H}, \mathtt{G}) \in \mathbb{R}$ *of tracking system* $(\mathtt{H}, \mathtt{G})$ *is*

$$C(\mathtt{H}, \mathtt{G}) = p(m+1)n + mn. \tag{5}$$

4 Learning the tracking system

In this section, we propose an iterative algorithm, which searches for the optimal tracking system with respect to a training set, i.e., it solves problem (4). This is an unconstrained optimization problem, where bilinear function is fitted in the least squares sense into a high dimensional data. We show later that problem (4) has a closed-form solution in $\mathtt{H}$ (respectively $\mathtt{G}$) if $\mathtt{G}$ (respectively $\mathtt{H}$) is fixed. Therefore the solution is searched

by an *exact line-search method*[2], which successively minimize criterion (4) along the direction $\mathtt{H}$ and $\mathtt{G}$.

In the very beginning of the learning process, the appearance encoder $\mathtt{G}$ is randomly initialized. Naturally, it does not provide any reasonable appearance parameters and its influence is negligible. Given random matrix $\mathtt{G}^0$, minimum of criterion function (3) over $\mathtt{H}$ has a closed-form solution $\mathtt{H}^0$. Given $\mathtt{H}^0$, the minimum over $\mathtt{G}$ has also a closed-form solution $\mathtt{G}^1$. In that manner, the error is iteratively minimized until a stopping condition is satisfied. In our implementation the learning stops if a relative error difference is smaller than some threshold ϵ. Number of appearance parameters m, desired complexity C (or equivalently number of pixels n), learning threshold ϵ and training set $(\mathtt{I}, \mathtt{J}, \mathtt{T})$ (optionally range of the tracker) are the only inputs to the learning procedure, which is summarized in Algorithm 1.

1. Randomly initialize matrix $\mathtt{G}^0$ and set number of current iteration $k = 1$.
2. Find $\mathtt{H}^k = \operatorname{argmin}_{\mathtt{H}} e(\mathtt{H}, \mathtt{G}^{k-1})$.
3. Find $\mathtt{G}^k = \operatorname{argmin}_{\mathtt{G}} e(\mathtt{H}^k, \mathtt{G})$.
4. Recompute error $e^k = e(\mathtt{H}^k, \mathtt{G}^k)$.
5. If $\frac{e^k - e^{k-1}}{e^k} < \epsilon$ stop, otherwise $k = k + 1$ and goto 2.

Algorithm 1 Exact line-search algorithm for problem (4)

In order to derive closed-form solution of step 2, we rewrite criterion function (3) as follows:

$$\begin{aligned} e(\mathtt{H}, \mathtt{G}) &= \left\| \mathtt{H}_0 \mathtt{I} + [\mathtt{H}_1 \dots \mathtt{H}_\mathtt{m}] \Big((\mathbf{1}_m \otimes \mathtt{I}) \bullet (\mathtt{GJ} \otimes \mathbf{1}_n) \Big) - \mathtt{T} \right\|_F^2 \\ &= \left\| \mathtt{H} \left[(\mathbf{1}_{m+1} \otimes \mathtt{I}) \bullet \left(\begin{bmatrix} \mathbf{1}_d^\top \\ \mathtt{GJ} \end{bmatrix} \otimes \mathbf{1}_n \right) \right] - \mathtt{T} \right\|_F^2 = \left\| \mathtt{HA} - \mathtt{T} \right\|_F^2, \end{aligned} \tag{6}$$

where

$$\mathtt{A} = (\mathbf{1}_{m+1} \otimes \mathtt{I}) \bullet \left(\begin{bmatrix} \mathbf{1}_d^\top \\ \mathtt{GJ} \end{bmatrix} \otimes \mathbf{1}_n \right) \tag{7}$$

is $(m+1)n \times d$ matrix. If the training set has

$$d \geq (m+1)n \tag{8}$$

independent samples. The closed-form solution of step 2 is

$$\mathtt{H}^k = \mathtt{TA}^+, \tag{9}$$

where $\mathtt{A}^+$ denotes pseudo-inverse of $\mathtt{A}$. Note, that condition (8) may not assure a reasonable behavior on testing data. It sets the lower bound. However, we usually generate five or ten times more training examples to make the tracker robust.

[2] An exact line-search method finds the length of step which minimizes a criterion in a descent direction, see [12] for details.

In order to derive closed-form solution of the step 3, we rewrite the criterion function (3) as follows:

$$e(\mathtt{H},\mathtt{G}) = \left\| \mathbf{g}^\top \mathtt{B} - \mathtt{C} \right\|_F^2, \tag{10}$$

where

$$\begin{aligned}
\mathbf{g}^\top &= [\mathbf{g}_1^\top \dots \mathbf{g}_m^\top], \\
\mathtt{B} &= \begin{bmatrix} \mathtt{I} \bullet \mathbf{h}_{11}^\top \mathtt{J} \otimes \mathbf{1}_n & \dots & \mathtt{I} \bullet \mathbf{h}_{1p}^\top \mathtt{J} \otimes \mathbf{1}_n \\ \vdots & \ddots & \vdots \\ \mathtt{I} \bullet \mathbf{h}_{m1}^\top \mathtt{J} \otimes \mathbf{1}_n & \dots & \mathtt{I} \bullet \mathbf{h}_{mp}^\top \mathtt{J} \otimes \mathbf{1}_n \end{bmatrix}, \\
\mathtt{C} &= \left[\mathbf{t}_1^\top - \mathbf{h}_{01}^\top \mathtt{J} \dots \mathbf{t}_p^\top - \mathbf{h}_{0p}^\top \mathtt{J} \right],
\end{aligned}$$

where $\mathbf{h}_{mp}^\top$ denotes a p-th row of $\mathtt{H}_m$. The closed-form solution of step 3 is

$$\mathbf{g}^\top = \mathtt{CB}^+ \text{ reshaped to } \mathtt{G}^k. \tag{11}$$

Since the matrix $\mathtt{B}$ has dimension $mn \times dp$, its pseudo-inverse requires

$$dp \geq mn \tag{12}$$

Notice, that this condition is clearly weaker than the condition (8) for every $p \geq 1$, therefore it need not be considered any further.

4.1 Convergence of Algorithm 1

We observed, that the proposed algorithm converges to the solutions, which has the same criterion value. If criterion (4) was a convex or quasi-convex function it would be simple to prove that Algorithm 1 converges to a global minimum. However, the criterion is neither convex nor quasi-convex. We propose a conjecture, that every local minimum of the criterion is simultaneously global. Using MAPLE, we analytically prove that the conjecture is true, but we were not able to generalize it for arbitrary number of variables. In this proof, we equaled the criterion gradient to zero and symbolically solve the algebraic system. The solution consists of a few manifolds but only one of them has a semidefinite Hessian, i.e., it creates local minima. This solution is substituted to the criterion showing that the criterion is constant along it.

In Section 5.3 the convergence is verified experimentally. It is shown that every local minima of criterion (4) within which the algorithm is trapped is global, i.e., the same criterion value is achieved every time and the solution is independent of the algorithm initialization.

5 Experiments

Three experiments are presented in this section. The first experiment (Section 5.1) compares *simple tracking system* which consists only from a single LLiP and the proposed *parameter sensitive* tracking system. It shows that the parameter sensitive system

achieves significantly smaller prediction error if the nature of the appearance changes allows some reasonable parameterization. The second experiment (Section 5.2) demonstrates some interesting properties of the appearance encoder. The third experiment (Section 5.3) shows the convergence properties of Algorithm 1.

5.1 Results on real sequences

In this experiment, we compare prediction error of simple tracking system and parameter sensitive tracking system of the same complexity. Note, that the simple tracking system is a special case of the parameter sensitive tracking system. Just for clarity we state that: simple tracking system estimates the motion parameters by LLiP with regressor $\mathtt{H}_s$. It is learned given a training set $(\mathtt{I}_s, \mathtt{T}_s)$ as follows:

$$\mathtt{H}_s = \mathtt{T}_s \mathtt{I}_s^+ \tag{13}$$

and its complexity

$$C_s = p_s n_s \tag{14}$$

is number of elements of $\mathtt{H}_s$.

Training/testing errors and complexities for different objects are presented in Table 1. The first three objects CUP, BASIL and SIBIL are taken from a British sitcom Fawlty Towers, see the first three rows in Figure 3. The other three object demonstrate variability of the appearance changes which can be cope by the tracker. In particular, fourth object HEAD 1 is a human head, where different expressions and illuminations influence its appearance, see fourth and fifth row in Figure 3. The fifth object HEAD 2 is human head, the appearance of which changes due to out-of-plane rotations, see third row of Figure 4. The object FLOWER is a flower with appearance strongly affected by non-trivial variable illumination and non-rigid deformations, see sixth row in Figure 3.

Notice that in all cases the prediction error of PLLiP was significantly smaller than the error LLiP despite of its lower complexity. Loss-of-lock events are denoted by "X". We used 5000-20000 training examples generated from 5-50 training images. Testing is performed on the sequences with 100-400 frames. The tracker estimated only translation ($p = 2$) in the range within the radius of 20 pixels, other degrees of freedom were represented by 1-5 appearance parameters. Non-optimized Matlab implementation of the learning procedure performing 10-40 iterations requires about 20-300 seconds on an average machine (1xK8 3200+ MHz). Note, that the learning time mainly depends on the number of training examples and the number of appearance parameters. The motion estimation time in the tracking procedure is negligible (say smaller than 1ms) in contrast to the time required for image capturing and its visualization.

We first discuss results from the Fawlty Towers sequences, where only one appearance parameter is used and the complexity of the parameter sensitive tracking system is two times smaller than the complexity of the simple tracking system. Despite of the lower complexity the error is smaller about approximately 30%. In the CUP experiment (first row in Table 1), the simple tracker lost the target during the fast motion (see

[1] Medical data presented in the last row are provided by Dr. Utz Kappert, Department of Cardiac Surgery, Heart Center Dresden University Hospital at the University of Technology Dresden.

Fig. 3. Objects[1] **with variable appearance** caused by illumination, non-rigid deformations and self-occlusions.

Object	Parameter sensitive				Simple		
	$\text{Error}_{\text{train}}$	$\text{Error}_{\text{test}}$	C	m	$\text{Error}_{\text{train}}$	$\text{Error}_{\text{test}}$	C_s
CUP	5.1	5.3	605	1	7.2	X	1352
BASIL	4.9	5.0	605	1	7.0	7.8	1352
SIBIL	4.9	7.4	605	1	7.0	X	1352
HEAD 1	2.7	3.4	1859	2	6.4	8.0	1922
HEAD 2	5.0	5.5	1248	2	7.5	8.5	1300
HEAD 2	4.3	4.5	1716	3	7.3	8.2	1860
FLOWER	3.3	3.6	2873	5	4.0	4.3	3362

Table 1. Comparison of PLLiPs and LLiPs: Prediction error is presented in pixels and the complexity follows definitions (5) and (14).

blurred image in Figure 3 1st row, 3rd column). In the BASIL experiment (second row in Table 1 and in Figure 3), both trackers succeeded in tracking however, the parameter sensitive tracker was more accurate. In the third experiment (third row in Table 1 and in Figure 3), the trackers learned on BASIL were used for tracking of SIBIL. While the simple tracker lost BASIL in the very beginning of the testing sequence the parameter sensitive tracker succeeded in tracking. Of course, the achieved error was higher.

The other three experiments demonstrates appearance changes which can be efficiently compensated by the appearance encoder. While in the HEAD 1 experiment (fourth row in Table 1), the error achieved by the proposed tracker is more than two times smaller, in the remaining experiments the improvement is not as significant. In the HEAD 2 experiment (fifth and sixth row in Table 1), small out-of-plane rotations are presumably interchangeable with translations, which consequently do not allow a unique interpretation of the observed image data. The rotation may be misinterpreted as a translation, and the image entering to the appearance encoder is not correctly aligned. Incorrect alignment yields incorrect appearance parameters and the error increases. Still, even in the worst case, the adaptive tracker did not produce worse results than the static one. In the FLOWER experiment (seventh row in Table 1), the difference between the parameter sensitive and static tracker is the smallest (about 20%) because the appearance changes are chaotic. Many mutual self-occlusions and shadows are casted by waving leaves, which does not allow for a reasonable parameterization.

5.2 Properties of the appearance encoder

This experiment demonstrates relationship between the current object appearance and appearance parameters estimated by the learned appearance encoder. First and second rows of Figure 4 show four frames from the BASIL sequence and corresponding course of one-dimensional appearance parameter. The lowest values are associated with the head profile, middle values correspond to the front view and the highest values correspond to the partial head occlusion by the white cup. Third row of Figure 4 shows a sequence, where human head turns around. Appearance parameters obtained by the parameter encoder are depicted in the second row of Figure 4. Notice, that these pa-

rameters naturally correspond almost exactly to the out-of-plane rotation, without any explicit knowledge about that evidence.

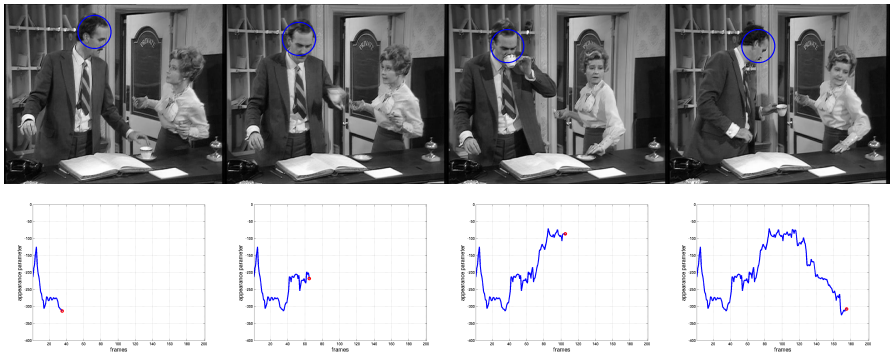


Fig. 4. Values of appearance parameters: Pose parameters are only translations, the other degrees of freedom are modeled as appearance changes. Selected frames from sequences (first row) and the corresponding output of one-dimensional (second row) appearance encoder. The red circle denotes the current appearance parameters, the blue line shows history.

5.3 Convergence of Algorithm 1

We study convergence of Algorithm 1 in this section. In order to demonstrate the convergence, we experimentally show that every local minima of criterion (4) within which the algorithm is trapped is global, i.e., the same criterion value is achieved every time.

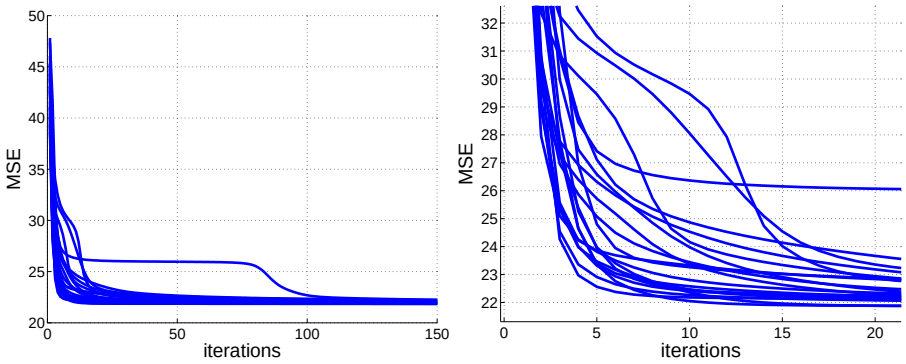
In the experiment, Algorithm 1 was one thousand times randomly initialized from uniform distribution and 150 iterations on 5000 training examples generated from sequence HEAD 1 were computed. Covariance of achieved values of the criterion function (3), i.e., Mean Square prediction Errors (MSE) is 3.4×10^{-3}. See Figure 5 for twenty convergence examples. From the detail depicted in Figure 5b, we conclude that 20 iterations are for most of the initializations sufficient.

6 Conclusions

We proposed a learnable tracking procedure suitable for objects with significantly varying appearance. The method was experimentally verified on the challenging sequences which exhibit strong variable illumination, non-rigid deformations and self-occlusions. The results were compared to the parameter insensitive tracking system, which was shown to be a special case of our approach. We demonstrated that on the same or lower complexity a smaller prediction error is achieved.

Acknowledgement

K. Zimmermann was supported by EC project FP6-IST-027787 DIRAC. T. Svoboda was supported by EC project FP6-IST-027787 DIRAC and Czech Academy of Sciences project 1ET101210407. J .Matas was supported by Czech Science Foundation Project 102/07/1317.



(a) convergence of Algorithm 1 (b) detail

Fig. 5. Covergence analysis of the learning procedure: Twenty convergence examples of randomly initialized algorithm.

References

1. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: International Journal of Computer Vision and Artificial Intelligence. (1981) 674–679
2. Dowson, N., Bowden, R.: N-tier simultaneous modelling and tracking for arbitrary warps. In: British Machine Vision Conference. Volume 2. (2006) 569–578
3. Ellis, L., Dowson, N., Matas, J., Bowden, R.: Linear predictors for fast simultaneous modelling and tracking. In: Proceedings of 11th IEEE International Conference on Computer Vision, workshop on Non-rigid registration and tracking through learning, Rio de Janeiro, Brazil, IEEE computer society (2007)
4. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. IEEE Transaction on Pattern Analysis Machine Intelligence **26**(6) (2004) 810–815
5. Shi, J., Tomasi, C.: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition. (1994) 593 – 600
6. Black, M.J., Jepson, A.D.: Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In Buxton, B.F., Cipolla, R., eds.: 4th European Conference on Computer Vision. Volume 1064 of Lecture Notes in Computer Science., Springer (1996) 329–342
7. Cascia, M.L., Sclaroff, S., Athitsos, V.: Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models. IEEE Transaction of Pattern Analysis Machine Intelligence **22**(4) (2000) 322–336
8. Cootes, T., Edwards, G., Taylor, C.: Active appearance models. IEEE Transaction on Pattern Analysis and Machine Intelligence **23**(6) (2001) 681–685
9. Hager, G.D., Belhumeur, P.N.: Efficient region tracking with parametric models of geometry and illumination. IEEE Trans. Pattern Analysis and Machine Intelligence **20**(10) (1998) 1025–1039
10. Jurie, F., Dhome, M.: Real time robust template matching. In: British Machine Vision Conference. (2002) 123–131
11. Fukunaga, K.: Statistical Pattern Recognition. Academic Press, New York, NY, USA (1990)
12. Gould, N.I.M., Leyffer, S.: An introduction to algorithms for nonlinear optimization. In: Frontiers in Numerical Analysis, Berlin, Springer Verlag (2003) 109 – 197