Adaptive Parameter Optimization for Real-time Tracking

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

[1]: Center for Machine Perception
[2]: Center for Applied Cybernetics
Czech Technical University,
Prague, Czech Republic

zimmerk@cmp.felk.cvut.cz

Adaptive Parameter Optimization for Real-time Tracking

Karel Zimmermann[1], Tomáš Svoboda[2] and Jiří Matas[1]
[1]: Center for Machine Perception [2]: Center for Applied Cybernetics
Czech Technical University,Prague, Czech Republic
zimmerk@cmp.felk.cvut.cz

Abstract

Adaptation of a tracking procedure combined in a common way with a Kalman filter is formulated as an constrained optimization problem, where a trade-off between precision and loss-of-lock probability is explicitly taken into account. While the tracker is learned in order to minimize computational complexity during a learning stage, in a tracking stage the precision is maximized online under a constraint imposed by the loss-of-lock probability resulting in an optimal setting of the tracking procedure. We experimentally show that the proposed method converges to a steady solution in all variables. In contrast to a common Kalman filter based tracking, we achieve a significantly lower state covariance matrix. We also show, that if the covariance matrix is continuously updated, the method is able to adapt to a different situations. If a dynamic model is precise enough the tracker is allowed to spend a longer time with a fine motion estimation, however, if the motion gets saccadic, i.e. unpredictable by the dynamic model, the method automatically gives up the precision in order to avoid loss-of-lock.

1. Introduction

Visual tracking comprises motion prediction from object dynamics followed by a measurement update step based on image data. The most common update method is the Lucas-Kanade tracker [5] which estimates object pose from a predicted pose $\mathbf{x}$ by the steepest gradient method. If $\mathbf{x}$ is the only input parameter then the number of iterations computed during $\Phi_{LK}(\mathbf{x})$ evaluation is determined automatically by detection of convergence. Optionally, the number of iteration is given by the maximum allowable time t for motion estimation, then the update function is $\Phi'_{LK}(\mathbf{x};t)$. We adopt to define a measurement update function $\Phi(\mathbf{x};t,r)$ [6] estimating a motion of the object given a range r determining an initial error which can be handled by the update function. $\Phi(\mathbf{x};t,r)$ is learned to compute the LSQ estimate of an arbitrary motion within range r computing as long as is allowed by the time t. Unlike Lucas

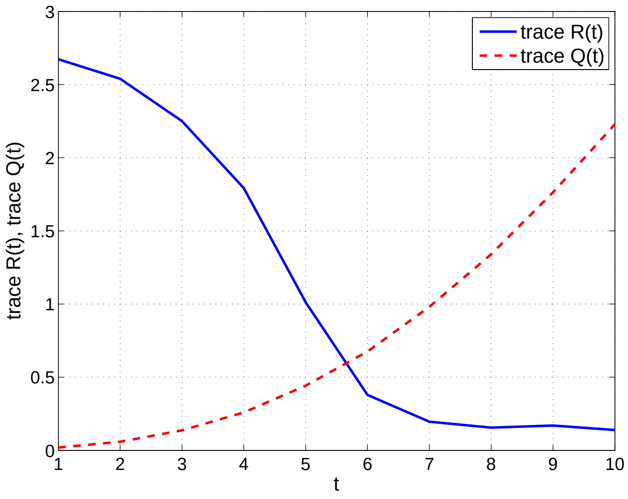


Figure 1. Covariance of the update function and linear dynamic model prediction errors. We clarify how to obtain these graphs from real data in Section 6.

and Kanade, our update function exploits the knowledge of the expected initial error resulting in more precise motion estimation for closer predictions.

In the paper, we formulate the tracker design as an optimization task where the time and range of $\Phi(\mathbf{x};t,r)$ are optimized in order to maximize the tracker precision given a loss-of-lock probability constraint. In each frame the time t minimizing the error in the following frame is estimated. Initial ranges are selected in order to satify a predefined bound on an approximated loss-of-lock probability.

For this tracking procedure, the measurement update function estimating the motion with the lowest computational complexity [1] with a given accuracy, is required. Hence, the computational complexity of the proposed update function is explicitly minimized given its desired range and precision during a learning stage.

Let us suppose that a function $\mathtt{R}(t)$, determining the covariance matrix of the measurement update error after time

[1]computational complexity directly corresponds to the computational time on a specific machine

t spent on motion estimation, is available. Since the longer time spent the better the update, the update error characteristic function $\Psi(t) = \text{tr}\,(\mathtt{R}(t))$ is bijection, see Figure 1.

Given an expected initial error ϵ, the output error is $\chi(\epsilon, t) = \Psi(\Psi^{-1}(\epsilon) + t)$. In current frame, the output error $\chi(\epsilon, t)$ decreases with the time allowing us to achieve an arbitrary precision, which is limited only by image resolution and nature of object texture. However, the longer the time spent on motion estimation the worse the prediction from the object dynamics, see Figure 1. Both the current output error $\chi(\epsilon, t)$ and dynamic prediction error characterised by $\text{tr}\,(\mathtt{Q}(t))$ significantly influence the resulting precision.

We experimentally show the improvement to a common tracking approaches [7, 9, 10], where dynamics and measurement updates are combined by the Kalman filtering [4]. If the characteristic $\text{tr}\,(\mathtt{Q}(t))$ is updated online, the system is demonstrated to automatically give up the precision when the motion gets saccadic and vice versa.

In Section 2 we formulate estimation of the time and initial range as an constrained optimization task. In Section 3 the problem is solved and incorporated into Kalman filter. The proposed algorithm is summarized in Section 4. Learning of the update method is described in Section 5. Section 6 demonstrates experiments and Section 7 summarizes the results.

2. Problem Definition

Let us suppose, we are able to express the probability distribution $\mathcal{F}_k(\epsilon; t_{k-1}, r_{k-1})$ of the prediction error ϵ in frame k for values of time and range (t_{k-1}, r_{k-1}) used in a previous frame. Then, the probability of loss-of-lock initilizing the update function $\Phi(\mathbf{x}; t, r)$ at range r for previously used values (t^*_{k-1}, r^*_{k-1}) is

$$P_{\text{loss-of-lock}}(r) = 1 - \int_{-r}^{r} \mathcal{F}_k(\epsilon; t^*_{k-1}, r^*_{k-1}) d\epsilon, \tag{1}$$

see for example Figure 2.

We measure the motion estimation error by covariance of the error distribution in the following frame $\text{tr}\,(\text{cov}_\epsilon\,(F_{k+1}(t, r)))$. This measure comprises contributions of tracking precision in the recent frame and the necessity of the range increases in the next frame. The task is to estimate parameters (t^*_k, r^*_k) which minimize error in the following frame and satisfy constraint that $P_{\text{loss-of-lock}}(r) < \epsilon$, which leads to the following constrained optimization problem:

$$\begin{aligned}(t^*_k, r^*_k) \quad &= \quad \arg\min_{t,r} \{\text{tr}\,(\text{cov}_\epsilon\,(F_{k+1}(\epsilon; t, r)))\} \\ &\quad\quad \text{subj. to:} 1 - \int_{-r}^{r} \mathcal{F}_k(\epsilon; t^*_{k-1}, r^*_{k-1})\, d\epsilon < \delta)\end{aligned} \tag{2}$$

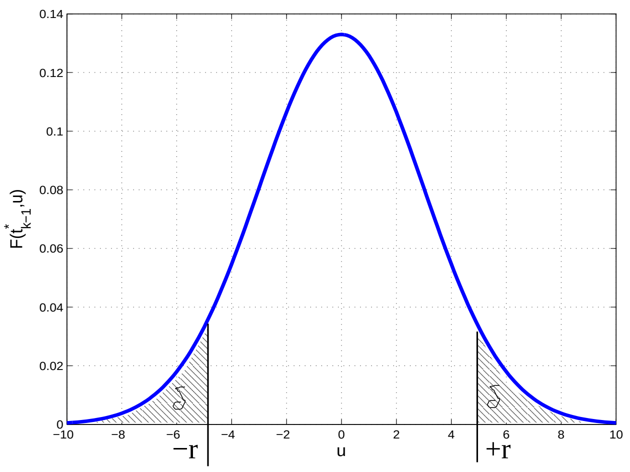


Figure 2. distribution of motion estimation error.

The smaller is the range r^*_k, the smaller area has to be searched through and the more accurate is the prediction at the same time, but the higher is $P_{\text{loss-of-lock}}(r^*_k)$. Therefore $\text{tr}\,(\text{cov}_\epsilon\,(F_{k+1}(\epsilon; t, r)))$ is nondecreasing function of r and we set up r^*_k as tight as possible to its upper bound $P_{\text{loss-of-lock}}(r^*_k) = \delta$. In consequence, problem (3) is decomposed to two separated tasks:

1. Find the smallest r satisfying the loss-of-lock constraint

$$r^*_k = \arg\min_r \left\{ r | \int_{-r}^{r} \mathcal{F}_k(\epsilon; t^*_{k-1}, r^*_{k-1})\, d\epsilon > 1 - \delta) \right\} \tag{3}$$

2. Find the time t minimizing error in the next frame for the fixed r^*_k

$$t^*_k = \arg\min_t \{\text{tr}\,(\text{cov}_\epsilon\,(F_{k+1}(\epsilon; t, r^*_k)))\}\,, \tag{4}$$

3. Estimation of the Probability Distribution Function $F_{k+1}(t, r)$ using Kalman filter

In this section, we derive covariance of probability distribution $\mathcal{F}_k$ adopting the Kalman filter principle. As we have shown, problem (2) can be decomposed into two independent optimization problems (3, 4). Because of simplicity we start with (4) and fix range $r_k = r^*_k$. In the end the computation of r^*_k is explained.

We address the general problem of an object state $\mathbf{x} \in \mathcal{R}^n$ estimation of a discrete process expressed by the linear stochastic difference equation

$$\mathbf{x}_k = \mathtt{A}\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \tag{5}$$

with a measurement

$$\mathbf{z}_k = \mathtt{H}\mathbf{x}_k + \mathbf{r}_{k-1}, \tag{6}$$

where random variables $\mathbf{q}_k \sim \mathcal{N}(0, \mathtt{Q}(t)), \mathbf{r}_k \sim \mathcal{N}(0, \mathtt{R}(t))$ represent the process and measurement noise. In contrast to Kalman filter, in our approach both covariance matrices are functions of the time spent on motion estimation in a particular frame. We further follow the idea of tracking by Kalman filter [2, 8]. Motion of an object in frame k is first approximated by a linear dynamic model

$$\mathbf{x}_{k|k-1} = \mathtt{A}\mathbf{x}_{k-1|k-1} \tag{7}$$

from a previous state $\mathbf{x}_{k-1|k-1}$, where $\mathtt{A}$ is an $n \times n$ matrix. Measurement $\mathbf{z}_k$ is estimated by a tracker Φ initialized at the position $\mathbf{x}_{k|k-1}$.

$$\mathbf{z}_k(t) = \Phi(\mathbf{x}_{k|k-1}; t, r), \tag{8}$$

The state $\mathbf{x}_k$ is approximated by

$$\mathbf{x}_{k|k}(t) = \mathbf{x}_{k|k-1} + \mathtt{K}(\mathbf{z}_k(t) - \mathtt{H}\mathbf{x}_{k|k-1}), \tag{9}$$

where $\mathtt{K}$ weights contributions of prediction by the dynamic model (7) and by the tracker (8). The error of approximations (7) has a Gaussian distribution with covariance matrices $\mathtt{Q}(t), \mathtt{R}(t)$. While tr $(\mathtt{R}(t))$ is a decreasing function of time, because the longer time we spend the better is the prediction, tr $(\mathtt{Q}(t))$ is increasing, because the longer time we spend, the smaller is the frame-rate and the harder is to predict the motion by a linear dynamic model, see Figure 1.

Covariance of a state estimation error in frame k based on measurements in frame k is

$$\mathtt{P}_{k|k}(t) = E\left\{(\mathbf{x}_k - \mathbf{x}_{k|k})(\mathbf{x}_k - \mathbf{x}_{k|k})^T\right\}. \tag{10}$$

For the sake of simplicity, we utilize the following results of common Kalman filter derivation [4] without any further explanation:

- Kalman gain $\mathtt{K}_k(t)$ minimizing tr $\left(P_{k|k}(t)\right)$ in frame k for an arbitrary t, r is

$$\mathtt{K}_k(t) = \mathtt{P}_{k|k-1}\mathtt{H}^T(\mathtt{H}\mathtt{P}_{k|k-1}(t)\mathtt{H}^T + \mathtt{R}(t))^{-1}. \tag{11}$$

- Covariance of the state estimation error in frame k is

$$\mathtt{P}_{k|k}(t) = (\mathtt{I} - \mathtt{K}_k(t)\mathtt{H})\mathtt{P}_{k|k-1}(t). \tag{12}$$

- Approximation of the covariance of the state estimation error in frame $k+1$ is

$$\mathtt{P}_{k+1|k}(t) = \mathtt{A}\mathtt{P}_{k|k}(t)\mathtt{A}^T + \mathtt{Q}(t). \tag{13}$$

Since we measure the precision by covariance matrix of the state estimation error, problem (4) is therefore replaced by

$$t_k^* = \arg\min_t \left\{\text{tr}\left(\mathtt{P}_{k+1|k}(t)\right)\right\}. \tag{14}$$

In order to find a global minimum of tr $\left(\mathtt{P}_{k+1|k}(t)\right)$, we express $\mathtt{P}_{k+1|k}(t)$ as a function independent of $\mathtt{K}_k(t)$. Substituting to Equation (13) for $\mathtt{P}_{k|k}(t)$ from Equation (12) and substituting for $\mathtt{K}_k(t)$ from Equation (11) we obtain

$$\mathtt{P}_{k+1|k}(t) = \tag{15}$$

$$\mathtt{A}\left(\mathtt{I} - (\mathtt{P}_{k|k-1}\mathtt{H}^T(\mathtt{H}\mathtt{P}_{k|k-1}(t)\mathtt{H}^T + \mathtt{R}(t))^{-1})\mathtt{H})\mathtt{P}_{k|k-1}(t)\right)\mathtt{A}^T + \mathtt{Q}(t)$$

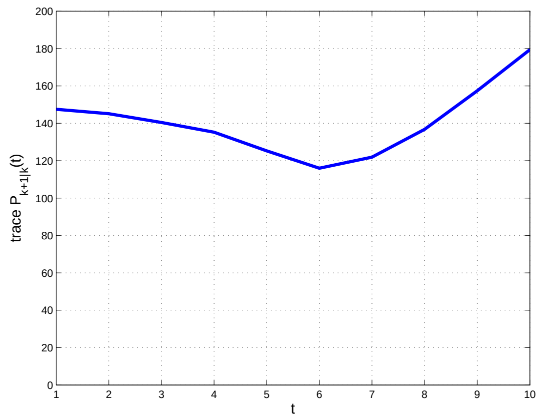


Figure 3. State covariance in the next frame tr $\left(\mathtt{P}_{k+1|k}(t)\right)$ as a function of t.

Trace of this function, see for example Figure 3, has usually one minimum determining the time t^* we are allowed to spend. As a side product, $\mathtt{P}_{k+1|k}(t^*)$ is also delivered, which allows estimation of an initial range r_{k+1}^* of the tracker in the following frame. Hence, Equation (3) simplifies to

$$r_{k+1}^* = \arg\min_r \left\{\left(\int_{-r}^{r} \mathcal{N}(u, \mathbf{0}, \mathtt{P}_{k+1|k}(t_k^*))\, du - 1 + \epsilon\right)^2\right\} \tag{16}$$

In order to avoid time consuming integration over at least 2-dimensional space, one could simply set $r_{k+1}^* \approx q.\text{tr}\left(\mathtt{P}_{k+1|k}(t_k^*)\right)$, which for example in 1D space for $q = 3$ results in $\epsilon \approx 0.2\%$.

Starting the tracking procedure in the following frame at a smaller range causes smaller state covariance allowing to estimate the motion more accurately, which again leads to decreases of the following state covariance. This process leads to a steady solution of $\lim_{k\to\infty}(\mathtt{K}_k, t_k, r_k, \mathtt{P}_{k|k})$ which is later experimentally shown in experiments.

4. Algorithm

The tracker Φ and dynamic model $\mathtt{A}, \mathtt{H}, \mathtt{Q}(t), \mathtt{R}(t)$ are determined during a learning stage described in Sections 5, 6. Tracking consists of two stages:

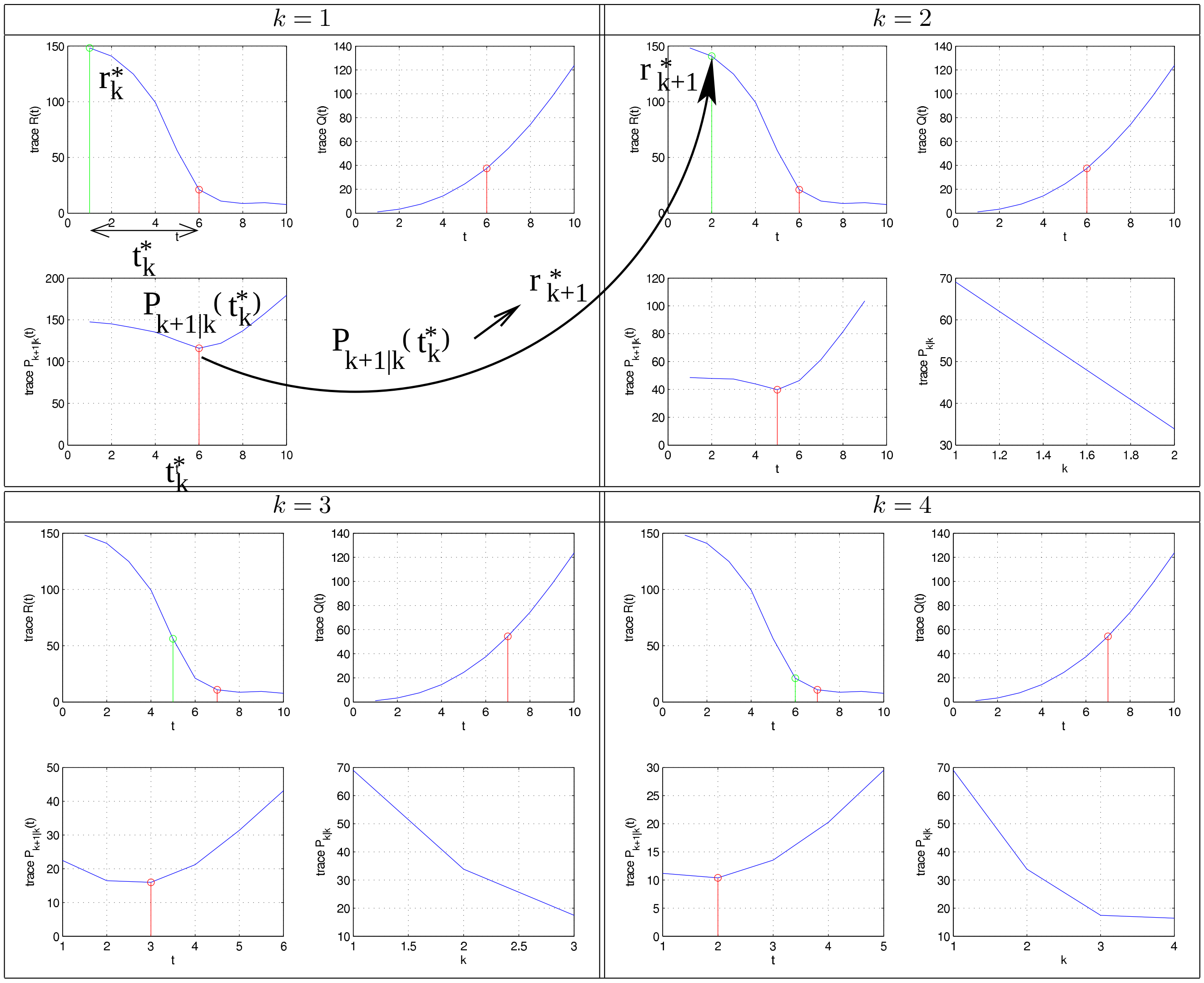


Figure 4. Visualization of Algorithm 1. Upper-left: $\mathrm{tr}\,(\mathtt{R}(t))$ as a function of t, Upper-right: $\mathrm{tr}\,(\mathtt{Q}(t))$ as a function of t, Lower-left: $\mathrm{tr}\,\left(\mathtt{P}_{k+1|k}(t)\right)$ as a function of t and its minimum, Lower-right: $\mathrm{tr}\,\left(\mathtt{P}_{k|k}(t^*_k)\right)$ as a function of k. Initial range r is denoted by green color, final time t is denoted by red color.

- **Time update**, where state $\mathbf{x}_{k|k-1}$ and covariance $\mathtt{P}_{k|k-1}$ are predicted from the motion dynamics and time t^*_k minimizing covariance $\mathtt{P}_{k+1|k}$ in the next frame is computed.

- **Measurement update**, where motion is estimated by the tracker $\Phi(\mathbf{x}_{k|k-1}; t^*_k, r^*_k)$. The final state approximation $\mathbf{x}_{k|k}$ is refined from both predictions.

The method is summarized by Algorithm 1, see also Figure 4 where the first 4 steps of the proposed algorithm are visualized. In each step functions $\mathrm{tr}\,(\mathtt{R}(t))$, $\mathrm{tr}\,(\mathtt{Q}(t))$, $\mathrm{tr}\,\left(\mathtt{P}_{k+1|k}(t)\right)$ and $\mathrm{tr}\,\left(\mathtt{P}_{k|k}(t^*_k)\right)$ are drawn. Initial range is depicted by the green color and the final time by the red color.

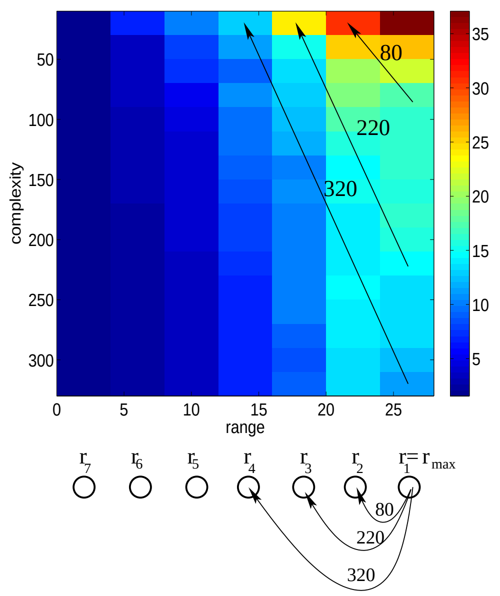
5. Learning of the tracker

In the previous sections, we have derived a tracking algorithm dealing with an arbitrary tracker $\Phi(\mathbf{x}; t, r)$, in this section we propose a tracker with clearly defined parameters range r and time t. Learning procedure which minimizes its computational complexity is also described.

The proposed tracker estimates the motion of the object from a subset of its pixels. The subset of the pixels is called *support set* $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_c\}$ The intensities observed on the support set X are collected in the *observation vector* $\mathbf{l}(X)$. Since different support sets of the same size provide different prediction errors, a subset with the smallest predictions error should be selected. Unfortunately, this problem is NP-hard. Although we have experimented with different

```
1. Estimate set of border predictors $\omega^+ = \{\varphi^+(c,r) \mid c \in C,\ r \in R\}$.
2. Construct graph $\mathcal{G} = (V \equiv R, E \subseteq R \times R, \alpha : E \to C)$:
    • for $r \in R$,
        • for $c \in C$,
            a) $v^* = \arg\min_{v \in R} v \mid \lambda(c,r) < v$
            b) $E = E \cup (r, v^*)$ and $\alpha(r, v^*) = c$
        • end
    • end
3. Dijkstra($\mathcal{G}, r_{max}$) $\Rightarrow$ cheapest path to each range $r \in R$.
4. Denote $\rho(r)$ complexity of the cheapest path to range $r$, then
   $\varphi_t^+(c_t, r_t) = \arg\min_{\varphi^+(c,r) \in \omega_T} \rho(r) + c$ is the last predictor.
5. An optimal sequential predictor is created from the sequence of predictors associated
   with the edges of the cheapest path to $r_t$ and the last predictor $\varphi_t^+(c_t, r_t)$.
```

(a) **Algorithm 2** - Estimation of the optimal sequence



(b) Construction of a graph $\mathcal{G}$

Figure 5. Construction of a graph $\mathcal{G}$ from a set of predictors ω. Edges from range r_{max} are depicted by black arrows.

```
1. Initialize covariance $P_{0|0}$, learn a tracker $\Phi$ and dynamic model $A, H, Q(t), R(t)$. Let $k = 1$
2. Get frame $k$
3. Time update equations:
    • $\mathbf{x}_{k|k-1} = A\mathbf{x}_{k-1|k-1}$
    • $t_k^* = \arg\min_{t \in \mathcal{R}} \{\mathrm{tr}\,(P_{k+1|k})\,(t)\}$
    • $P_{k|k-1}(t_k^*) = AP_{k-1|k-1}(t_k^*)A^T + Q(t_k^*)$
4. Measurement update:
    • $K_k(t_k^*) = P_{k|k-1}H^T(HP_{k|k-1}(t_k^*)H^T + R(t_k^*))^{-1}$
    • $\mathbf{z}_k(t_k^*) = \Phi(\mathbf{x}_{k|k-1}; t_k^*, r_k^*)$
    • $\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + K(t_k^*)(\mathbf{z}_k(t_k^*) - H\mathbf{x}_{k|k-1})$
    • $P_{k|k}(t_k^*) = I - K_k(t_k^*)H)P_{k|k-1}(t_k^*)$
    • $r_{k+1}^* = q.\mathrm{tr}\,(P_{k+1|k}(t_k^*))$, where $q$ is a parameter.
5. $k = k + 1$ repeat from 2.
```

Algorithm 1 - the proposed method of tracking.

algorithms for the support set selection, let us assume, that the support set has been selected for example randomly.

Let $(\mathbf{p} \circ X)$ denotes the support set transformed by a motion with parameters $\mathbf{p}$. For example, if the considered motion is a 2D translation $\mathbf{t}$, then $(\mathbf{p} \circ X) = (X + \mathbf{t}) = \{(\mathbf{x}_1 + \mathbf{t}), \ldots, (\mathbf{x}_c + \mathbf{t})\}$. There is a mapping (rendering) from parameters $\mathbf{p}$ to observations $\mathbf{l}(\mathbf{p} \circ X)$, which is not usually uniquely invertible. Nevertheless, we search for a linear mapping

$$\hat{\varphi} = H\mathbf{l}, \tag{17}$$

approximating the inverse relation $\mathbf{l}^{-1}$. This mapping assigns a p-vector of motion parameters to a c-vector of observation. All the regressors $\hat{\varphi}$ are characterized by the following parameters:

Definition: *Complexity* c is the size of a support set, which is directly proportional to the time t.

Definition: *Range* $R_{\hat{\varphi}}(r)$ of the regressor $\hat{\varphi}$ is a a circular region with radius r of motion parameters within which the regression is defined.

Definition: *Uncertainty region* of the regressor $\hat{\varphi}$ is circle

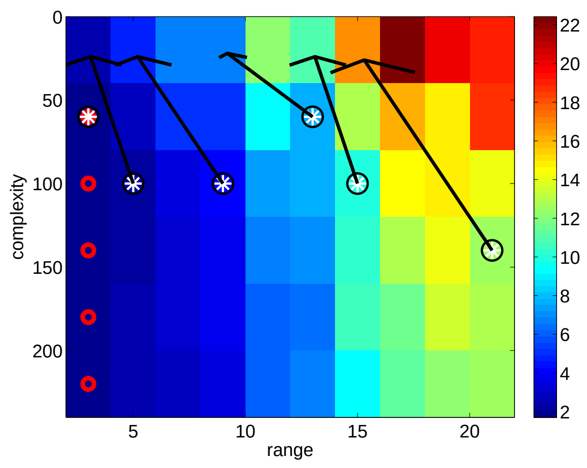
$$\Lambda_{\hat{\varphi}}(\lambda) = \left\{ \Delta\mathbf{p} \mid \Delta\mathbf{p} = \|\mathbf{p} - \hat{\varphi}\Big(\mathbf{l}(\mathbf{p} \circ X)\Big)\|_2 < \lambda,\ \forall \mathbf{p} \in R_{\hat{\varphi}} \right\}. \tag{18}$$

In the other words, the uncertainty region is the smallest circle within which all the predictions from the range $R_{\hat{\varphi}}$ lie.

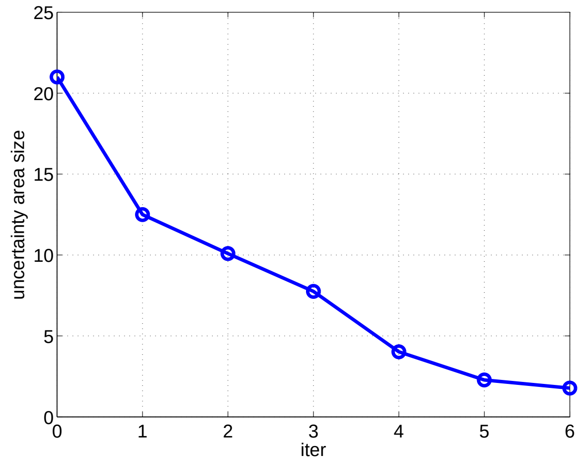
Definition: *Predictor* is an ordered pair $\varphi(c, r, \lambda) = (\hat{\varphi}, X)$ which satisfies the following requirements:

- $\forall \mathbf{p} \in R_{\hat{\varphi}}(r) : \ \mathbf{p} - \hat{\varphi}\Big(\mathbf{l}(\mathbf{p} \circ X)\Big) \in \Lambda_{\hat{\varphi}}(\lambda)$
- $|X| = c$

Let us suppose we are given a training set: the set of synthesized examples of observed intensities $\mathbf{l}^i$ and motions $\mathbf{t}^i$ around a reference point. These examples are column-wise formed into matrices $\mathtt{L}$ and $\mathtt{T}$. Size of $\mathbf{l}^i$ corresponds to a desired complexity c of $\hat{\varphi}$. Range of generated motions corresponds to a desired range r. Given the training set $(\mathtt{L}, \mathtt{T})$,



(a) path with the lowest complexity



(b) iterations

Figure 6. (a) Size of uncertainty regions (coded by colors) as a function of complexity c (vertical direction) and range r (horizontal direction) and the optimal path from the initial $r_{max}]$ to a predictor with sufficiently small uncertainty region (red circles). (b) size of uncertainty region after each iteration (iter = 0 corresponds to the range $r_{max} = 25$).

we estimate coefficients $\mathtt{H}$ of the linear regressor (17) as follows

$$\mathtt{H} = \mathtt{TL}^{+} = \mathtt{TL}^{\top}(\mathtt{LL}^{\top})^{-1}. \tag{19}$$

The higher is the complexity the better is the prediction. However, as the complexity increases towards the complete whole template the improvement steps become less and less significant. In general, for large ranges it is very difficult to achieve a good prediction with reasonable complexity. In order to overcome this limitation we develop a sequential predictor $\Phi = (\varphi_1 \dots \varphi_m)$, which is such a concatenation of predictors, which yields lower complexity for a higher precision. The vector of motion parameter $\mathbf{p}$ is predicted in m iterations as follows:

$$\begin{aligned}
\mathbf{p}_1 &= \hat{\varphi}_1\big(\mathbf{l}(X_1)\big),\\
\mathbf{p}_2 &= \hat{\varphi}_2\big(\mathbf{l}(\mathbf{p}_1 \circ X_2)\big),\\
\mathbf{p}_3 &= \hat{\varphi}_3\big(\mathbf{l}(\mathbf{p}_2 \circ \mathbf{p}_1 \circ X_3)\big),\\
&\ \vdots \\
\mathbf{p}_m &= \hat{\varphi}_m\Big(\mathbf{l}\big((\bigcirc_{i=1}^{h-1} \mathbf{p}_i) \circ X_m\big)\Big),\\
\mathbf{p} &= \bigcirc_{i=1}^{h} \mathbf{p}_i,
\end{aligned} \tag{20}$$

While the first vector of motion parameters $\mathbf{p}_1$ is directly predicted from the intensities observed in the support set X_1, the following predictors refine on the preceding predictions. The advantage is that each predictor in a sequence is more and more specific, using range corresponding to the accuracy of the preceding predictor.

Obviously, we consider only those sequential predictors which satisfy $r_{i+1} \geq \lambda_i,\ i = 1 \dots m-1$. It means that, the range of each particular predictor must at least accommodate the uncertainty region of its predecessor. Uncertainty region of the sequential predictor is understood as the uncertainty region of the last predictor and its range as the range of the first predictor.

Definition: *Sequential predictor* is an m-tuple $\Phi(\mathbf{x}; t, r) = (\varphi_1(c_1, r_1, \lambda_1), \dots, \varphi_m(c_m, r_m, \lambda_m))$ of predictors $\varphi_i \in \omega$ such that $\forall r_{i+1} \geq \lambda_i, i = 1 \dots m-1$. Uncertainty region of the sequential predictor is λ_m and its range is $r = r_1$. Time t is implementation and machine specific value which is directly proportional to $\sum c_i$.

We are looking for a sequential predictor

$$\Phi^*(\mathbf{x}; t, r) = \arg\min_{\Phi \in \Omega} \sum_{i=1}^{m} c_i. \tag{21}$$

where Ω is the set of sequential linear predictors with regressors learned by Equation (5).

Let ω denote a set of linear predictors $\varphi(c, r)$ with regressors learned by LSQ method[2], Equation , for some discretized values of complexities $c \in C$ and ranges $r \in R$. Figure 6a shows sizes $\lambda(c, r)$ (coded by colors) of their uncertainty regions as a function of complexity $c \in C$ (vertical direction) and range $r \in R$ (horizontal direction). Given the set ω, desired range r_{max} and uncertainty region λ_{min}, we search for an ordered subset of ω that constitutes sequential predictor Φ^*, Equation (21). Since the range of the sequential predictor corresponds to the range $r_1 = r_{max}$ of the first predictor in the sequence, the first predictor must lie in the corresponding (usually the most right) column. Complexity c_1 of the first predictor is unknown. Color at the place

[2]Since the learning method uniquely determines the uncertainty region, λ will not be further involved among parameters.

(c_1, r_1) in Figure 6a corresponds to the size of uncertainty region λ_1. If a complexity c_1 is chosen the current predictor $\varphi^+(c_1, r_1)$ is determined and the following predictor can be selected from those with the range greater than the achieved uncertainty region. We consider only the predictor with the smallest range because higher ranges cannot provide smaller complexities. In such a way, a sequence with the last predictor with the size of uncertainty region λ_m smaller than λ_{min} is constructed. Furthermore, we search for a sequence consisting of such predictors which converges to the sufficiently small uncertainty regions with the lowest complexity.

We formulate the previous problem as a searching of the cheapest path in graph $\mathcal{G} = (V \equiv R, E \subseteq R \times R, \alpha : E \to C)$, where R is the set of considered ranges and C is the set of considered complexities and α assigns cost to each edge, see Figure 5. In each range (vertex), a set of edges with different complexities starts. Edges correspond to the predictors learned for the same range but with different complexities. The higher is the complexity the smaller is the achieved uncertainty region and the smaller is its target range. Dijkstra algorithm estimates the cheapest path in the graph, as a result the cheapest path to each range is found. Some of these ranges include a target predictors, i.e. those with $\omega_T = \{\varphi(c, r) \mid \lambda(c, r) < \lambda_{min}\}$ (depicted by red circles in Figure 6b. The one which creates the cheapest path in conjunction with the path to the corresponding range is added to the path. The predictors corresponding to the edges of the resulting path create sequential predictor (21). The method is summarized in Algorithm 3.

6. Experiments

In our implementation, the tracker is an optimal sequential predictor [6], which is a sequence of linear predictors estimated during a learning stage in an optimal way. Instead, you can imagine any other tracker, e.g. the Lucas-Kanade tracker [5, 1] or Meanshift [3] initialized at a certain position covering a range r and using number of iterations corresponding to time t.

Experiment in Section 6.1 shows some results achieved by Algorithm 1 on the tracking of a planar patch shown in Figure 7a. Section 6.2 shows results on a different sequence Figure 7b, achieved with an extension of the proposed method, where state covariance matrix was updated.

6.1. Verification and implementation

We verify our method on a sequence with length of $m = 300$ frames. Tracked object is a planar 50×50-pixels patch, see Figure 7a. We learned an optimal linear sequential predictor estimating its 2D translation in a range of 25 pixels.

Considering state vector $\mathbf{x} = \begin{bmatrix} p_x & p_y & v_x & v_y \end{bmatrix}^T$, where p_x, p_y are positions and v_x, v_y are velocities, linear dynamic model is estimated from a training sequence $\mathbf{x}_1, \dots \mathbf{x}_m$ as follows:

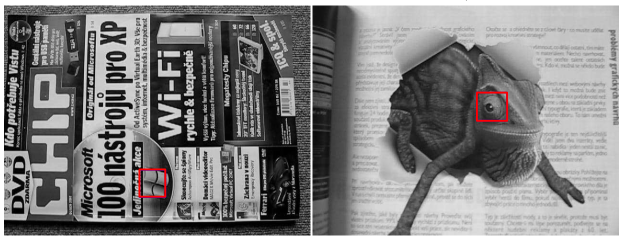

(a) (b)

Figure 7. Tracked patch in (a) the first and (b) second experiment.

$$\mathtt{A} = [\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_m]\, [\mathbf{0}, \mathbf{x}_1 \dots \mathbf{x}_{m-1}]^+ ,$$

where $+$ denotes pseudo-inverse operation and $\mathbf{0}$ is vector of zeros. Also $\mathtt{R}(t)$ and $\mathtt{Q}(\mathtt{t})$ for some discretized values $t \in T \subset \mathcal{R}$ are computed for a training sequence.

We conclude that the proposed method converges to a steady solution with $\operatorname{tr}\left(\mathtt{P}_{k|k}\right)$ almost $2\times$ smaller than a common Kalman tracking approaches, see Figure 8. Note that, the visualization (Figure 4) of the first 4 steps of the Algorithm 1 was obtained during this experiment. The reader is also encouraged to watch attached videos.

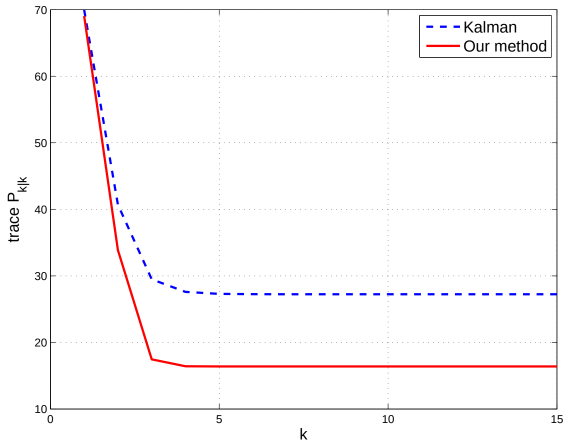


Figure 8. Convergence of $\operatorname{tr}\left(\mathtt{P}_{k|k}\right)$ to a steady solution.

6.2. Online $\mathtt{Q}(t)$ updating

Covariance matrix of dynamic model prediction error $\mathtt{Q}(t)$ characterizes an ability of the Equation (7) to predict the motion of target. Consequently, the value of $\operatorname{tr}\left(\mathtt{Q}(t)\right)$ significantly influences precision of tracking: If the linear model is precise the algorithm is allowed to spend longer time with fine motion estimation, however, if the linear model fails, for example because of saccadic motion of the target, the method automatically gives up the precision in order to avoid loss-of-lock. We therefore extend Algorithm 1 to the method which automatically updates $\mathtt{Q}(t)$. The results on a testing sequence with 1323 frames, where the first 400 are well predictable by Equation (7) and in the rest saccadic unpredictable motions are present, are shown in Figure 9, where position p_x is depicted as a function of frames, the sizes of blue circles correspond to tr $\left(P_{k|k}\right)$.

We also conclude that an unpredictable motion causes an increase in frame-rate and inevitably a decrease in precision. Frame-rate and tr $\left(P_{k|k}\right)$ are visualized in Figure 10.

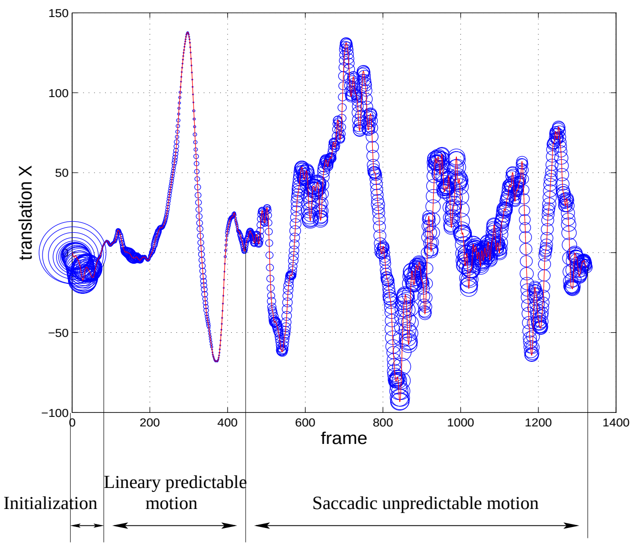


Figure 9. If the linear model fails, for example because of saccadic motion of the target, the method automatically gives up the precision in order to avoid loss-of-lock. Size of blue markers is proportional to tr $\left(\mathsf{P}_{k|k}(t_k^*)\right)$.

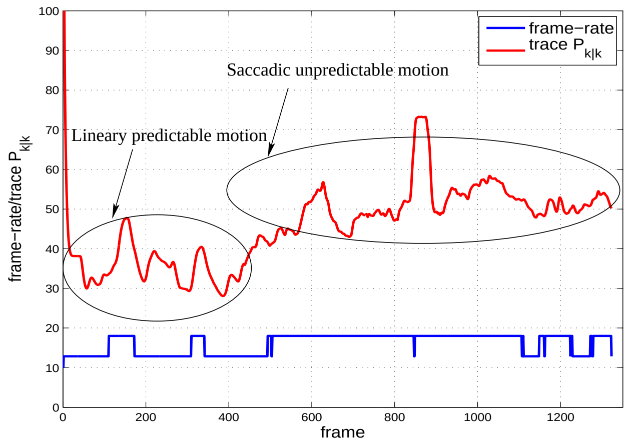


Figure 10. Covariance and frame-rate as a function of time.

7. Conclusions

We have shown an adaptive extension to a common Kalman tracking approach, where time spent with motion estimation and initial range of the tracker are optimized. The task is formulated as a maximization of tracking precision given a bound on loss-of-lock probability. We showed, that as well as the common approach, the proposed process leads to the steady solution of all variables, however, significantly smaller steady solution of a state covariance matrix is achieved. We also proposed learnable updating method, where computational complexity is minimized during a learning stage resulting in a fast update function suitable for real-time applications.

Acknowledgement

K. Zimmermann was supported by The Czech Academy of Sciences project 1ET101210407. T. Svoboda was supported by the EC project FP6-IST-027787 DIRAC. J. Matas was supported by The Grant Agency of Czech Republic under project 102/07/1317.

References

[1] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. 7

[2] A. Blake and M. Isard. *Active Contours*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998. 3

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002. 7

[4] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the ASME - Journal of Basic Engineering, 82(D):35-45*, 1960. 2, 3

[5] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981. 1, 7

[6] J. Matas, K. Zimmermann, T. Svoboda, and A. Hilton. Learning efficient linear predictors for motion estimation. In *Indian Conference on Computer Vision, Graphics and Image Processing*, LNCS4338, pages 445–456. Springer-Verlag, 2006. 1, 7

[7] Y. Satoh, T. Okatani, and K. Deguchi. A color-based tracking by kalman particle filter. In *International Conference on Pattern Recognition*, volume 3, pages 502–505, Washington, DC, USA, 2004. IEEE Computer Society. 2

[8] G. Welch and G. Bishop. An introduction to the kalman filter. In *Special Interest Group for Computer Graphics*, 2001. 3

[9] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *International Conference on Computer Vision*, volume 1, pages 44–52, 2003. 2

[10] Z. Zhu, Q. Ji, and K. Fujimura. Combining kalman filtering and mean shift for real time eye tracking under active ir illumination. volume 4, pages 318–321, 2002. 2