Extended Lucas-Kanade Tracking

Shaul Oron[1], Aharon Bar-Hillel[2], and Shai Avidan[3]

[1] Tel Aviv University, Israel
shauloro@post.tau.ac.il
[2] Microsoft Research
aharonb@microsoft.com
[3] Tel Aviv University, Israel
avidan@eng.tau.ac.il

Abstract. The Lucas-Kanade (LK) method is a classic tracking algorithm exploiting target structural constraints thorough template matching. Extended Lucas Kanade or ELK casts the original LK algorithm as a maximum likelihood optimization and then extends it by considering pixel object / background likelihoods in the optimization. Template matching and pixel-based object / background segregation are tied together by a unified Bayesian framework. In this framework two log-likelihood terms related to pixel object / background affiliation are introduced in addition to the standard LK template matching term. Tracking is performed using an EM algorithm, in which the E-step corresponds to pixel object/background inference, and the M-step to parameter optimization. The final algorithm, implemented using a classifier for object / background modeling and equipped with simple template update and occlusion handling logic, is evaluated on two challenging data-sets containing 50 sequences each. The first is a recently published benchmark where ELK ranks 3rd among 30 tracking methods evaluated. On the second data-set of vehicles undergoing severe view point changes ELK ranks in 1st place outperforming state-of-the-art methods.

1 Introduction

The famous Lucas-Kanade (LK) algorithm[19] is an early, and well known, algorithm that takes advantage of object structural constraints by performing template based tracking.

Structure is a powerful cue which can be very beneficial for reliable tracking. Early methods performing template matching [19,21,20,7] later evolved and inspired the use of multiple templates and sparse representations to represent target appearance [30,5,14,24], and for known target classes a more complex use of structure can be made [27]. Learning based tracking methods can also use template matching, some examples are by target appearance mining [15] or exemplar based classification [4]. Some methods disregard target structure, for example by performing pixel-wise classification [3,11] or using histogram representations [6]. Although this can be beneficial in cases where targets are highly deformable, these methods are in most cases very pron to drift as they do not enforce any structural constraints.

Using structure, in the form of template matching, can help tracking algorithms avoid drift and maintain accurate tracking through target and scene appearance changes.

D. Fleet et al. (Eds.): ECCV 2014, Part V, LNCS 8693, pp. 142–156, 2014.

These changes can be related to in/out-of-plane rotation, illumination changes, motion blur, rapid camera movement, occlusions, target deformations and more. The drift problem is extremely difficult since tracking is performed without user intervention, apart from some initialization in the first frame, which is usually a rectangle bounding the target region.

One of the problems arising when using target templates, bounded by a rectangle, is the inclusion of background pixels in the target image. When matching the template, one is also required to match the included background pixels which can ultimately lead to drift. Our proposed method therefore attempts to perform template matching, using the structural cue, while requiring object / background consistencies between template and image pixels.

The contribution of this work is a novel template tracking algorithm we denote Extended Lucas Kande or ELK. Inspired by the famous LK algorithm our algorithm extends the original one to accounts for pixels object / background likelihood. We first cast the original LK problem is terms of probabilistic inference demonstrating the loss function minimizing the sum-of-square-difference (SSD) between image and template is equivalent to maximum likelihood estimation under Gaussian noise assumption. We then introduce hidden variables related to image and template pixels object / background likelihoods. We derive an extension of the original LK algorithm which includes 2 additional log-likelihood terms in the loss function. These terms enforce that object / background template pixels are matched to object / background image pixels respectively. In addition, from this derivation emerge pixel weights, used in the template matching term computation, as well as a factor regularizing between the template matching term and the object / background log-likelihood terms, which can be used to regularize between ordered template matching and disordered probability mode matching. We derive an estimation-maximization (EM) algorithm which enables maximizing the loss function and inferring the hidden variables.

We implement this new algorithm using a boosted stumps classifier for object / background modeling and equip it with a simple occlusion handling logic. The resulting algorithm achieves results comparable to state-of-the-art methods on a challenging tracking benchmark ranking in 3rd place among 30 trackers evaluated.

2 Extended Lukas Kanade Tracking

The Lucas-Kanade (LK) tracking algorithm works quite well when the template to be tracked consists entirely of pixels belonging to the object. Problems arise when background pixels are added to the template which cause the algorithm to drift.

To combat that we propose a Bayesian model that combines template matching (i.e., regular LK) with objecthood reasoning at the pixel level. Tracking is performed by finding a $2D$ transformation that maximizes the model likelihood.

We start in section 2.1 by introducing notation and casting traditional template matching, as done by the LK algorithm, in a probabilistic framework. In section 2.2 we extend the probabilistic framework to include pixel objecthood reasoning, by introducing a model including foreground/background hidden variables. In section 2.3 we derive an EM formulation for inferring the hidden objecthood variables and optimizing some of

the model parameters, not including the tracking transformation. Finally, in section 2.4, we show how the tracking transformation can be found as part of the EM M-step, using an extension of the traditional LK algorithm.

2.1 Template Matching and Traditional LK

We wish to track template $\mathbf{T}$ with the set of pixels $\mathbf{P} = \{\mathbf{p}\}_{i=1}^{n}$, where $\mathbf{p} = (x, y)$ is the 2D pixel location, and $\mathbf{T}(\mathbf{p})$ denotes pixel $\mathbf{p}$ of template $\mathbf{T}$. Let $\mathbf{I}$ denote the image at time t. We say that image pixel $\mathbf{I}(W(\mathbf{p};\omega))$ is mapped to template pixel $\mathbf{T}(\mathbf{p})$ by the transformation W with parameter vector ω. Examples are 2D translation or similarity transformation. The algorithm assumes an estimated position of $\mathbf{T}$ in the image at time $t-1$ is known and given by ω^{t-1}, i.e the set of pixels $\mathbf{I}(W(\mathbf{P};\omega^{t-1}))$ is a noisy replica of $\mathbf{T}$.

Given template $\mathbf{T}$, the estimated previous position ω^{t-1}, and a new image $\mathbf{I}$, LK looks for an update $\Delta\omega$ s.t. $\omega = \omega^{t-1} + \Delta\omega$ with $\Delta\omega$ minimizing:

$$\Delta\omega = \arg\min \sum_{\mathbf{p}\in\mathbf{P}} (\mathbf{I}(W(\mathbf{p}, \omega^{t-1} + \Delta\omega)) - \mathbf{T}(\mathbf{p}))^2 \tag{1}$$

Using a Gauss-Newton method.

This algorithm has a natural probabilistic interpretation. Assuming a Gaussian independent pixel noise we look for the maximum likelihood pixel set $\mathbf{I}(W(\mathbf{P};\omega^{t-1} + \Delta\omega))$:

$$\begin{aligned} &\max_{\Delta\omega} logP(\mathbf{I}(W(\mathbf{P};\omega^{t-1} + \Delta\omega))|\mathbf{T}) \\ &= \max_{\Delta\omega} \textstyle\sum_{\mathbf{p}\in\mathbf{P}} logG(\mathbf{I}(W(\mathbf{p}, \omega^{t-1} + \Delta\omega)) - \mathbf{T}|0, \sigma) \\ &= -\tfrac{1}{2}|\mathbf{P}|log(2\pi\sigma^2) \; - \tfrac{1}{2\sigma^2} \min_{\Delta\omega}[\textstyle\sum_{\mathbf{p}\in\mathbf{P}} (\mathbf{I}(W(\mathbf{p}, \omega^{t-1} + \Delta\omega)) - \mathbf{T}(\mathbf{p}))^2] \end{aligned} \tag{2}$$

Where $G(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu)^2)$ is the Gaussian density function. In words, we assume that the (log) probability of the image given the template is (log) Gaussian. Since the optimization in equations (1) and (2) are the same w.r.t the optimal $\Delta\omega$, we see that LK is equivalent to searching for a maximum likelihood pixel set.

Note that in an important sense, this is not a traditional ML formulation: usually the data to explain is *fixed* and we learn model parameters which makes it the most likely. Here the model has no 'traditional' parameters (it has σ, but it is not relevant to the optimization), and the data to explain is what we optimize over, by treating the transformation ω as a parameter and optimizing over it.

2.2 Template Matching with Objecthood Inference

In this section we present a graphical model that combines rigid template matching with pixel based foreground/background reasoning. The model is presented as a Bayesian network with an added event constraint. It is then simplified to a graphical model over 4 variables: the pixel values of template and image, and hidden variables determining their foreground/background affiliation.

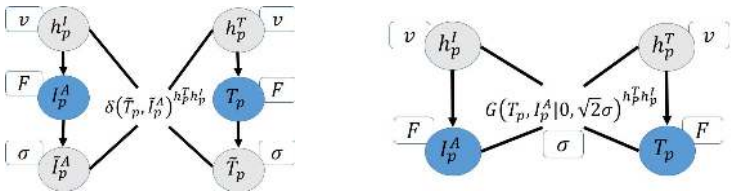


Fig. 1. Graphical Model: (a) before and (b) after simplification. (a) We observe image $\mathbf{I}$ and template $\mathbf{T}$ and wish to estimate the hidden variables: $h_p^{\mathbf{I}}$ (hidden binary variable, is the image pixel an object or background?), $\tilde{\mathbf{I}}_p$ (hidden image), $h_p^{\mathbf{T}}$ (hidden binary variable, is the template pixel an object or background?), $\tilde{\mathbf{T}}_p$ (hidden template). (b) After simplification, the hidden template $\tilde{\mathbf{T}}_p$ and image $\tilde{\mathbf{I}}_p$ vanish, and all we have to estimate are the binary variables $h_p^{\mathbf{I}}$ and $h_p^{\mathbf{T}}$. The match between $\mathbf{T}$ and $\mathbf{I}$ is assumed to come from a Gaussian distribution G with 0 mean and σ variance.

The Model. The image $\mathbf{I}$ and the template $\mathbf{T}$ we observe are assumed to be noisy measurements of some hidden and noisy source image $\tilde{\mathbf{I}}$ and template $\tilde{\mathbf{T}}$. We further know that pixels in the template and the image can belong to the object or the background. Clearly, we would like to make sure that when we match pixels in the template to pixels in the image we match object pixels and not background pixels.

To do that, let $h(\mathbf{p})$ be a binary variable that determines if the pixel belongs to the background (i.e., is 0) or object (i.e., is 1). This gives us four variables per pixel: the pixel in the hidden template $\tilde{\mathbf{T}}(\mathbf{p})$, its corresponding pixel in the hidden image $\tilde{\mathbf{I}}(W(\omega, \mathbf{p}))$, and their binary object/background binary variables $h_p^{\mathbf{T}} = h(\tilde{\mathbf{T}}(\mathbf{p}))$ and $h_p{}^{\mathbf{I}} = h(\tilde{\mathbf{I}}(W(\omega, \mathbf{p})))$, respectively. For brevity, from now on we will denote $\mathbf{T}_p = \mathbf{T}(\mathbf{p})$ and $\mathbf{I}_p = \mathbf{I}(W(\mathbf{p}, \omega))$.

The Connections between the hidden variables and the observables $\mathbf{T}_p$ and $\mathbf{I}_p$ are given by the graphical model in Figure 1(Left), which is replicated for each pixel $\mathbf{p} \in \mathbf{P}$. The prior probabilities of pixels (both template and image) to be foreground are Bernoulli distributed with a parameter v, i.e. $P(h = 1) = v,\ P(h = 0) = 1 - v$. The pixel appearance models, with parameters shared between template and image, are given by $P(\mathbf{I}_p|h_p^{\mathbf{I}}, F), P(\mathbf{T}_p|h_p^T, F)$. We denote by F the parameters of this conditional probability. For example, we can implement F using two discrete histograms of pixel values, one for the object and one for the background, or we can use a discriminative model. Finally, we let $P(\tilde{\mathbf{T}}_p|\mathbf{T}_p) = G(\tilde{\mathbf{T}}_p|\mathbf{T}_p, \sigma), P(\tilde{\mathbf{I}}_p|\mathbf{I}_p) = G(\tilde{\mathbf{I}}_p|\mathbf{I}_p), \sigma)$ be Gaussian connections.

The model described up until now is a standard Bayesian network. However, in the space spanned by this network, we are interested in the subspace obeying the following condition: if both template $\mathbf{T}_p$ and image $\mathbf{I}_p$ are object pixels, than $\tilde{\mathbf{T}}_p$ and $\tilde{\mathbf{I}}_p$ are identical, i.e. $\mathbf{T}_p$ and $\mathbf{I}_p$ are noisy replica of the same source. Denoting this event by Ω, we are interested in:

$$P_\Omega(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p, \tilde{\mathbf{T}}_p, \tilde{\mathbf{I}}_p) = P(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p, \tilde{\mathbf{T}}_p, \tilde{\mathbf{I}}_p) 1_\Omega \tag{3}$$

where 1_Ω is given by :

$$1_\Omega = \begin{cases} 1, & h_p^{\mathbf{T}} = 0 \; or \; h_p^{\mathbf{I}} = 0 \\ \delta(\tilde{\mathbf{T}}_p - \tilde{\mathbf{I}}_p) & h_p^{\mathbf{T}} = 1 \; and \; h_p^{\mathbf{I}} = 1 \end{cases} = \delta(\tilde{\mathbf{T}}_p - \tilde{\mathbf{I}}_p)^{h_p^{\mathbf{T}} h_p^{\mathbf{I}}} \tag{4}$$

with $\delta(\cdot)$ denoting the Dirac delta. The event-restricted joint probability we consider is hence:

$$\begin{aligned} &P_\Omega(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p, \tilde{\mathbf{T}}_p, \tilde{\mathbf{I}}_p) = \\ &P(h_p^{\mathbf{T}}) P(h_p^{\mathbf{I}}) P(\mathbf{T}_p | h_p^{\mathbf{T}}) P(\mathbf{I}_p | h_p^{\mathbf{I}}) G(\tilde{\mathbf{T}}_p | \mathbf{T}_p) G(\tilde{\mathbf{I}}_p | \mathbf{I}_p) \delta(\tilde{\mathbf{T}}_p - \tilde{\mathbf{I}}_p)^{h_p^{\mathbf{T}} h_p^{\mathbf{I}}} \end{aligned} \tag{5}$$

The model parameters are $\Theta = \{v, F, \sigma, \omega\}$. Like in the formalism presented for traditional LK, we optimize here not only over traditional model parameters $\{v, F, \sigma\}$, but also over the data we explain via the choice of ω.

Model Simplification. We can simplify the model by integrating $\tilde{\mathbf{T}}, \tilde{\mathbf{I}}$ out:

$$\begin{aligned} P_\Omega(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p) = \int\int P_\Omega(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p, \tilde{\mathbf{T}}_p, \tilde{\mathbf{I}}_p) d\tilde{\mathbf{T}} d\tilde{\mathbf{I}} = \\ P(h_p^{\mathbf{T}}) P(h_p^{\mathbf{I}}) P(\mathbf{T}_p | h_p^{\mathbf{T}}) P(\mathbf{I}_p | h^{\mathbf{I}}) \\ \times \int\int G(\tilde{\mathbf{T}}_p | \mathbf{T}_p, \sigma) \cdot G(\tilde{\mathbf{I}}_p | \mathbf{I}_p, \sigma) \delta(\tilde{\mathbf{T}}_p - \tilde{\mathbf{I}}_p)^{h_p^{\mathbf{T}} h_p^{\mathbf{I}}} d\tilde{\mathbf{T}} d\tilde{\mathbf{I}} \end{aligned} \tag{6}$$

If $h_p^{\mathbf{T}} = 0$ or $h_p^{\mathbf{I}} = 0$, the double integral decomposes into two independent integrals of Gaussian CDF, hence it is 1. If $h_p^{\mathbf{T}} = 1,\ h_p^{\mathbf{I}} = 1$ the double integral collapses into a single integral of a product of Gaussians. Such a product of two Gaussians is a scaled Gaussian, with the scaling factor itself a Gaussian $G(\mathbf{T}_p - \mathbf{I}_p | 0, \sqrt{2}\sigma)$ [8] in the means $\mathbf{T}_p, \mathbf{I}_p$. Therefore, the double integral at the end of the equation above simplifies to $G(\mathbf{T}_p - \mathbf{I}_p | 0, \sqrt{2}\sigma)^{h_p^{\mathbf{T}} h_p^{\mathbf{I}}}$. Following the elimination of $\tilde{\mathbf{T}}_p, \tilde{\mathbf{I}}_p$ we get a simpler model structure over 4 variables:

$$\begin{aligned} P_\Omega(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p) = \\ P(h_p^{\mathbf{T}}) P(h_p^{\mathbf{I}}) P(\mathbf{T}_p | h_p^{\mathbf{T}}) P(\mathbf{I}_p | h_p^{\mathbf{I}}) G(\mathbf{T}_p - \mathbf{I}_p | 0, \sqrt{2}\sigma)^{h_p^{\mathbf{T}} h_p^{\mathbf{I}}} \end{aligned} \tag{7}$$

This simplified model is described in the graphical model is Figure 1(Right)

2.3 An EM Formulation

As in traditional LK, we are given a template $\mathbf{T}$, the estimated position in the previous frame ω^{t-1}, and a new image $\mathbf{I}^t$ (we will omit the t super script for notation simplicity),

and we look for an update $\omega = \omega^{t-1} + \Delta\omega$ giving us the maximum-likelihood pixel set:

$$\max_{\Theta} \log P(\mathbf{T}, \mathbf{I}_\omega|\Theta) = \max_{\Delta\omega} \max_{v,F,\sigma} \log P(\mathbf{T}, \mathbf{I}_{\omega^{t-1}+\Delta\omega}|v, F, \sigma) \tag{8}$$

Assuming pixel independence we have

$$\begin{aligned} \log P(\mathbf{T}, \mathbf{I}_\omega) &= \sum_{\mathbf{p}\in\mathbf{P}} \log P(\mathbf{T}_p, \mathbf{I}_p) \\ &= \sum_{\mathbf{p}\in\mathbf{P}} \sum_{h_p^{\mathbf{T}}\in\{0,1\}} \sum_{h_p^{\mathbf{I}}\in\{0,1\}} \log P(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p) \end{aligned} \tag{9}$$

With $P(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}, \mathbf{T}_p, \mathbf{I}_p)$ given by Eq. 7. Expressions like this, containing a summation inside the log function are not optimization-friendly in a direct manner, so we resort to EM [9] optimization. In our case the parameters are $\Theta = \{v, F, \sigma, \omega\}$, the hidden variables are:

$$H = \{H^{\mathbf{T}}, H^{\mathbf{I}}\} = \{h_p^{\mathbf{T}}, h_p^{\mathbf{I}}\}_{\mathbf{p}\in\mathbf{P}}, \tag{10}$$

and the observables are

$$O = \{\mathbf{T}, \mathbf{I}\} = \{\mathbf{T}_p, \mathbf{I}_p\}_{\mathbf{p}\in\mathbf{P}}. \tag{11}$$

Following the EM approach, we will optimize Θ by

$$\Theta_{new} = \arg\max_{\Theta} E_{old} \log P(H^{\mathbf{T}}, H^{\mathbf{I}}, \mathbf{T}, \mathbf{I}) \tag{12}$$

where $E_{old} = E_{P(H^{\mathbf{T}}, H^{\mathbf{I}}|\mathbf{T},\mathbf{I},\Theta_{old})}$.

E-step: Given known $\mathbf{T}, \mathbf{I}, \Theta_{old}$, the distribution $P(H^{\mathbf{T}}, H^{\mathbf{I}}|\mathbf{T}, \mathbf{I}, \Theta_{old})$ over the hidden variables is easy to infer. $P(H^{\mathbf{T}}, H^{\mathbf{I}}|\mathbf{T}, \mathbf{I}, \Theta_{old})$ decomposes into a product of

$$P(h_p^{\mathbf{T}}, h_p^{\mathbf{I}}|\mathbf{T}_p, \mathbf{I}_p, \Theta_{old}) \tag{13}$$

for each pixel $\mathbf{p}$, using pixel independence assumption. Since $h_p^{\mathbf{T}}, h_p^{\mathbf{I}}$ are discrete binary variables, we can get the conditional probability for any pixel $\mathbf{p}$, hidden values $b_1, b_2 \in \{0, 1\}$, by:

$$P(h_p^{\mathbf{T}} = b_1, h_p^{\mathbf{I}} = b_2|\mathbf{T}_p, \mathbf{I}_p) = \frac{P(h_p^{\mathbf{T}}=b_1, h_p^{\mathbf{I}}=b_2, \mathbf{T}_p, \mathbf{I}_p)}{\sum_{a_1\in\{0,1\}} \sum_{a_2\in\{0,1\}} P(h_p^{\mathbf{T}}=a_1, h_p^{\mathbf{I}}=a_2, \mathbf{T}_p, \mathbf{I}_p)} \tag{14}$$

Computing the conditional distribution hence requires only evaluating Eq. 7 four times, for the four possible combinations of $h_p^{\mathbf{T}}, h_p^{\mathbf{I}}$ values. We will see below that the objecthood probability $P(h_p^{\mathbf{T}} = 1, h_p^{\mathbf{I}} = 1|\mathbf{T}_p, \mathbf{I}_p)$ has the role of template-matching pixel weights in the optimization of ω and σ. For the optimization of other parameters, the probabilities $P(h_p^{\mathbf{I}}|\mathbf{T}_p, \mathbf{I}_p)$,$P(h_p^{\mathbf{T}}|\mathbf{T}_p, \mathbf{I}_p)$ are used, and they are obtained from $P(h_p^{\mathbf{T}} = 1, h_p^{\mathbf{I}} = 1|\mathbf{T}_p, \mathbf{I}_p)$ by simple marginalization.

M-step: For notation convenience, let $P_{old}(H^{\mathbf{T}}, H^{\mathbf{I}}) = P(H^{\mathbf{T}}, H^{\mathbf{I}}|\mathbf{T}, \mathbf{I}, \Theta_{old})$. For a single pixel we have to maximize the expectation of the log of Eq. 7 which, after some manipulation leads to the following update equations:

$$v^{new} = \frac{\sum_{\mathbf{p}\in\mathbf{P}} P_{old}(h_p^{\mathbf{I}}) + P_{old}(h_p^{\mathbf{T}})}{2|\mathbf{P}|} \tag{15}$$
$$\sigma^{new} = \sqrt{\frac{\sum_{\mathbf{p}\in\mathbf{P}} P_{old}(h_p^{\mathbf{I}}=1, h_p^{\mathbf{T}}=1)(\mathbf{T}_p - \mathbf{I}_p)^2}{\sum_{\mathbf{p}\in\mathbf{P}} P_{old}(h_p^{\mathbf{I}}=1, h_p^{\mathbf{T}}=1)}}$$

As stated before, F can be implemented using two histograms $F = (F^0, F^1)$, with $F^0(c)$, $F^1(c)$ keeping the frequency of pixel value c according to figure (F^0) and background (F^1) histogram respectively. In that case, the update rule would be

$$F^l(c) = \frac{\sum_{p:p=c} P_{old}(h_p^I = l) + P_{old}(h_p^T = l)}{2|\mathbf{P}|} \quad for\ l \in \{0,1\} \tag{16}$$

However, we use instead a discriminative model. In this case we use the previous pixel weights $P_{old}(h_p^{\mathbf{I}}), P_{old}(h_p^{\mathbf{T}})$ as pixel weights when training the parameters F of the model. As for the transformation parameters ω, gathering the terms dependent on it from the expected log-likelihood gives the following optimization problem:

$$\max_{\omega} \sum_{b\in\{0,1\}} \sum_{\mathbf{p}\in\mathbf{P}} [P_{old}(h_p^{\mathbf{I}} = b) \log P(\mathbf{I}_p|h_p^{\mathbf{I}} = b)) \tag{17}$$
$$-\frac{1}{4\sigma^2} \sum_{\mathbf{p}\in\mathbf{P}} P_{old}(h_p^{\mathbf{I}}=1, h_p^{\mathbf{T}}=1)(\mathbf{T}_p - \mathbf{I}_p)^2]$$

We see that ω has to optimize a balance of two terms: The first is a foreground (and background) likelihood term, demanding that foreground pixels will correspond to the foreground appearance model (and similarly for background pixels). The second term requires rigid template matching and traditional LK, but only for pixels with high probability of being foreground. The relative weight of the two terms depends on σ, so adaptively changing this parameter moves the emphasis between appearance-based orderless matching and rigid template matching. We next see that the Gauss Newton optimization technique used in standard LK can be extended for the new objective function.

2.4 ELK Optimization Algorithm

maximizing Equation 17 in the context of a forward-additive LK algorithm is straightforward. Given image $\mathbf{I}$ taken at time t, we use F, the parameters of the conditional distribution, to obtain a log probability images for foreground $\mathbf{I_1} = \log P(\mathbf{I}|h^{\mathbf{I}} = 1)$ and background $\mathbf{I_0} = \log P(\mathbf{I}|h^{\mathbf{I}} = 0)$. Then, we use the standard first order Taylor expansion to approximate each of them and arrive at the following objective function that we wish to maximize:

$$L(\Delta\omega) = \sum_{\mathbf{p}\in\mathbf{P}} \{\mathbf{Q_0}(\mathbf{p})[\mathbf{I_0}(\mathbf{p},\omega) + \nabla\mathbf{I_0}\frac{dW(\omega)}{d\omega}\Delta\omega] \tag{18}$$
$$+\mathbf{Q_1}(\mathbf{p})[\mathbf{I_1}(W(\mathbf{p},\omega)) + \nabla\mathbf{I_1}\frac{dW(\omega)}{d\omega}\Delta\omega]$$
$$-\mathbf{Q}(\mathbf{p})[\mathbf{T}(\mathbf{p}) - \mathbf{I}(W(\mathbf{p},\omega)) - \nabla\mathbf{I}\frac{dW(\omega)}{d\omega}\Delta\omega]^2\}$$

where

$$\mathbf{Q_0}(\mathbf{p}) = P_{old}(h^{\mathbf{I}}(W(\mathbf{p},\omega)) = 0) \tag{19}$$
$$\mathbf{Q_1}(\mathbf{p}) = P_{old}(h^{\mathbf{I}}(W(\mathbf{p},\omega)) = 1)$$
$$\mathbf{Q}(\mathbf{p}) \;\; = \frac{1}{4\sigma^2}P_{old}(h^{\mathbf{I}}(W(\mathbf{p},\omega)) = 1, h^{\mathbf{T}}(\mathbf{p}) = 1)$$

Equation 18 is an extension of the standard LK objective function. The third row works on the input image $\mathbf{I}$ and is the regular LK objective function measuring the similarity between the template and the image, weighted by $\mathbf{Q}$. This term requires the pixels of the template and image to match each other, but only if both of them are likely to be object pixels. The first row works on the (log) likelihood background image $\mathbf{I_0}$ and requires the motion to match the prior assignment of pixels to foreground and background, as given by the weight function $\mathbf{Q_0}$. Similarly, the second row works on the (log) likelihood foreground image $\mathbf{I_1}$ and requires the motion to match the prior assignment of pixels to foreground and background, as given by the weight function $\mathbf{Q_1}$.

Taking the derivative of L with respect to $\Delta\omega$, setting it to 0 and rearranging into the following vector notation:

$$V_0 = \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{Q_0}(\mathbf{p})\nabla\mathbf{I_0}\frac{dW(\omega)}{d\omega} \tag{20}$$
$$V_1 = \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{Q_1}(\mathbf{p})\nabla\mathbf{I_1}\frac{dW(\omega)}{d\omega}$$
$$V = \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{Q}(\mathbf{p})[\mathbf{T}(\mathbf{p}) - \mathbf{I}(W(\mathbf{p},\omega))]\nabla\mathbf{I}\frac{dW(\omega)}{d\omega}$$
$$M = \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{Q}(\mathbf{p})[\nabla\mathbf{I}\frac{dW(\omega)}{d\omega}]^T[\nabla\mathbf{I}\frac{dW(\omega)}{d\omega}]$$

Leads to the following equation:

$$V_0 + V_1 - 2V + 2M\Delta\omega = 0 \tag{21}$$

And the solution is:

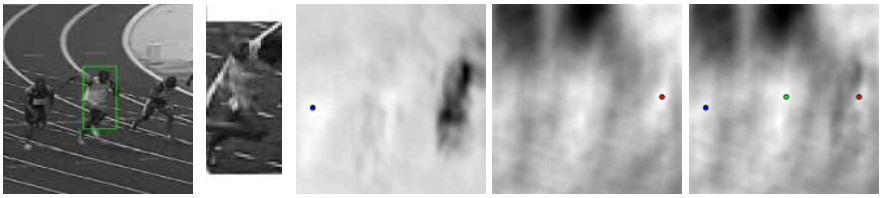
$$\Delta\omega = M^{-1}(V - \frac{V_0 + V_1}{2}) = M^{-1}V - M^{-1}(\frac{V_0 + V_1}{2}) \tag{22}$$

where the vectors V_0, V_1, V_2, V are in R^l (l is the number of parameters of the transformation, i.e., $l = 6$ for 2D affine transformation) and the matrix M is an invertible $l \times l$ matrix.

We obtained a simple extension of the standard LK solution which is $\Delta\omega = M^{-1}V$ in our current notation, by adding a term corresponding to the gradient of the foreground / background (log) likelihood. Like in standard LK this step should be iterated several times until convergence, and repeating the algorithm in multiple scales can enhance the convergence range.

Note that since $\mathbf{Q}(\mathbf{p})$ is inversely proportional to σ (see Eq. 19), so is the vector V and the matrix M, but not V_0, V_1. The term $M^{-1}V$ is invariant to σ, but the newly added term $M^{-1}\left(\frac{V_0+V_1}{2}\right)$ is proportional to σ. Small σ values hence lead to the traditional LK algorithm, and large σ emphasizes the new term. This is reasonable, as large σ corresponds to a weak demand for template matching.

Figure 2 illustrates the main components of ELK. When a new image arrives, we wish to maximize the expected log likelihood (Eq. 17) containing the two terms. In this case, trying to match the template, or the foreground/background images separately leads to wrong answer. Only the combined optimization function tracks the template correctly.



a) Image b) Template c) Template term d) Log-likelihood term e) Combined

Fig. 2. Contribution of new log-likelihood terms to combined optimization function. From left to right: a) Image with final target bounding box overlaid. b) Target template. c) optimization function template matching term (weighted SSD), maximum marked in blue (brighter is better). d) optimization function combined log-likelihood terms, maximum marked in red. e) Combined optimization function including both template and log-likelihood, maximum marked in green. On their own both template matching and log-likelihood terms do not point to the correct target position however the combined loss does point out the correct target position.

3 Experiments

We evaluate ELK tracking performance using two data-sets[1]. The first is a recently published tracking benchmark [26]. comparing 29 tracking algorithms on a challenging set of 50 sequences. The sequences include abrupt motion, object deformations, in/out-of-plane rotations, illumination changes, occlusions, blur and clutter. The second data-set [22], also containing 50 sequences, depicts road scenes captured from 3 backwards facing cameras mounted on a maneuvering vehicle. The data contains vehicle targets

[1] Code and data will be made available at
`http://www.eng.tau.ac.il/~oron/ELK/ELK.html`

undergoing severe view point changes related to turning, overtaking maneuvers and going around traffic circles, some examples are shown in figure 5. Results for 7 tracking algorithms have been reported on this data-set, among which are recently published algorithms which produce state-of-the-art results on the benchmark mentioned above.

We adopt the one-pass success criterion, suggested in the benchmark, which quantifies both centering accuracy and scale. We measure the overlap between predicted and ground truth bounding boxes, i.e. the intersection area of the boxes divided by the union area, for each frame. A success curve is then computed for each sequence by measuring the fraction of frames with $overlap \geq threshold$ for $threshold$ values in $[0, 1]$. The success is then averaged over all sequences producing a final curve showing overall performance of each method at every $threshold$ value. In addition the area-under-curve (AUC) is used as a figure of merit to compare between the different tracking methods.

3.1 Implementation Details

For each frame we run a single EM iteration: the transformation ω is optimized for $P_{old}(H^{\mathbf{T}}, H^{\mathbf{I}})$ computed in the previous frame, followed by an E-step to recompute $P(H^{\mathbf{T}}, H^{\mathbf{I}})$. Taking a region-of-interest (ROI) around the last target position, we use two scales, in the lower scale we search only for a 2D-translation in an exhaustive manner. We then use this as an initial guess for the full resolution level where we search for both location and scale using the Gauss-Newton iterations described in section 2.4. We limit the number of Gauss-Newton iterations to 5 per frame. In addition we always consider zero-order-hold (ZOH). This practice was found to help avoid singular scale errors induced by gradient decent. Choosing between these two states is done using a confidence measure as will be explained later.

The images processed are transformed into YCbCr representation, and photometrically scaled to have standard deviation of 1 in every channel. We use a discriminative classifier in order to obtain pixel foreground/background probabilities. The classifier is trained by boosting random decision stumps [1]. Our feature space consists of pixel YCbCr values in a 8x8 window around each pixel as well a histogram-of-oriented-gradients (HOG) feature of 8 bin histograms built in 4 spatial cells of size 2×2. The margins provided by the classifier are transformed into the range $[0, 1]$ using a sigmoid.

In order to cope with target deformations and appearance changes we regularly update both our target model and our foreground/background model every K frames (in our experiments $K = 5$). This is done only when tracking confidence is high meaning we are not occluded or drifting. We use two measures to establish tracking confidence. The first is the weighted mean-square-error (MSE), between the current target image and the predicted target location, normalized by mean weight value (punishing for overall low foreground likelihood). As a threshold for this measure we use twice its median value in a sliding temporal window. The second confidence measure is demanding that the median of the weight map exceeds a threshold (in our experiments we use threshold$= 0.75$, meaning we require at least 50% of pixels to have foreground likelihood greater than 0.75). The template and foreground/background model are updated only if both measures indicate high confidence. When updating the template we consider the current appearance, the previous appearance, or the initial appearance, taking the one producing the minimal normalized MSE. When updating the

foreground/background model the image foreground/background weights are used as weights for classifier training.

We note that using a standard PC our non-optimized Matlab implementation runs at $\sim 1 fps$, processing rate may vary according to target size.

3.2 Results

For the benchmark data-set [26], ELK produces results comparable to state-of-the-art methods as presented in figure 3. It is ranked in 3rd place for overall performance on this benchmark data-set (among 30 tracking methods evaluated), with AUC of 0.454 (following Struck 0.474 and SCM with 0.499). See table 2 for a full list of tracking methods appearing in all figures. Performance of simple LK tracking (without our extensions) are not presented since the simple LK tracker produces very poor results achieving an AUC score of 0.05. Table 1 presents AUC and ELK rank for different sequence attributes in the benchmark data-set. We observe that ELK ranks first or second for sequences exhibiting out-of-plane rotations or deformation. It also produces decent results for sequences with fast motion scale variations, occlusions and in-plane rotations ranking 4th in all categories. The lowest rank (7) is obtained for sequences with illumination variation. This is not surprising as both template appearance and object / background models suffer from abrupt illumination variations affecting all terms in the optimized.

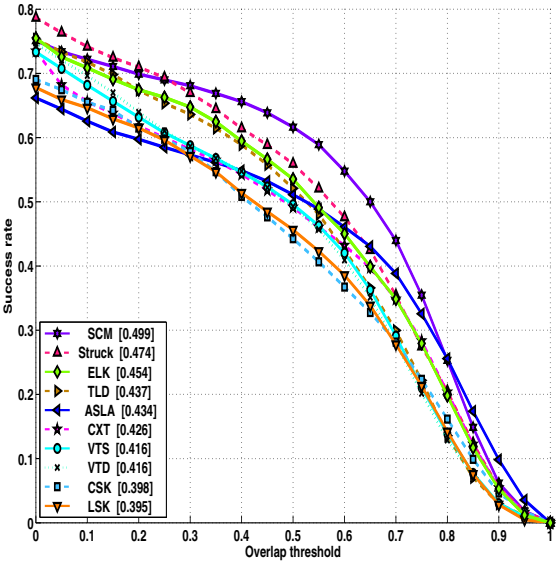


Fig. 3. Success plot for the benchmark data-set [26], showing top 10 methods (out of 30): ELK (in Green) is ranked 3rd in overall performance demonstrating results comparable to state-of-the-art methods (best viewed in color)

Table 1. ELK success and rank for different sequence attributes in the benchmark [26] data-set

Attribute	Number of Seq.	AUC	Rank
In-plane rotation	31	0.430	4
Out-of-plane rotation	39	0.462	2
Deformation	19	0.479	1
Scale variation	28	0.423	4
Occlusion	29	0.409	4
Illumination variation	25	0.390	7
Motion blur	12	0.336	5
Fast motion	17	0.387	4

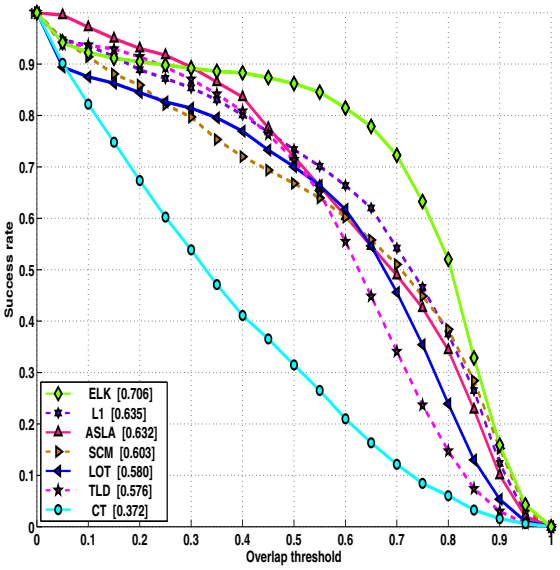


Fig. 4. Success plot for the vehicle data-set [22]: ELK (in Green) is ranked 1st in overall performance, with a large margin, among 8 tracking methods evaluated on this data (best viewed in color).

On the vehicles data set of [22], where template matching playes a more significant role, ELK outperforms all the other methods tested with a significant margin. The success plots are presented in figure 4. As can be seen in figure 5, this data set contains challenging scenarios with respect to viewpoint, scale change, and illumination. However, the fact that vehicles are rigid provide more opportunities for template matching, and makes ELK the clear winner. For this data simple LK achieved AUC of only 0.35.

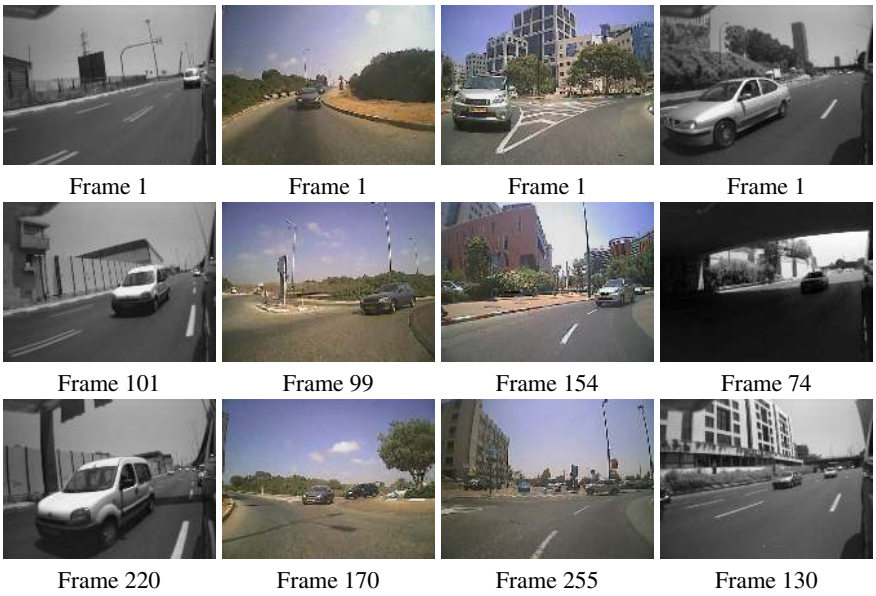


Fig. 5. Sample frames from the vehicle data-set [22] depicting vehicles undergoing severe view point changes. Each column shows frames taken from the same sequence.

Table 2. List of tracking methods appearing in result figures

Method	Paper
ASLA[14]	Visual Tracking via Adaptive Structural Local Sparse Appearance Model
CSK[13]	Exploiting the Circulant Structure of Tracking-by-Detection with Kernels
CT[29]	Real-time Compressive Tracking
CXT[10]	Context Tracker: Exploring Supporters and Distracters in Unconstrained Environments.
ELK	Extended Lucas Kanade Tracking - **Proposed method**
L1[5]	Real Time Robust L1 Tracker Using Accelerated Proximal Gradient Approach
LOT[23]	Locally Orderless Tracking
LSK[18]	Robust Tracking using Local Sparse Appearance Model and K-Selection
SCM[30]	Robust Object Tracking via Sparsity-based Collaborative Model
Struck[12]	Struck: Structured Output Tracking with Kernels.
TLD[15]	Tracking-Learning-Detection
VTD[16]	Visual Tracking Decomposition
VTS[17]	Tracking by Sampling Trackers

4 Conclusions

ELK is a novel tracking algorithm combining template matching with pixel object / background segregation. This special combination allows ELK to be more resistive to drift as it can perform template matching while disregarding template background pixels. Additionally the new log-likelihood terms introduced into the optimization, can direct the algorithm when deformation, that cannot be accounted for by the template, occur. This allows the algorithm to maintain reliable tracking in the presence of severe deformations until the model is updated.

ELK was demonstrated to produce results comparable to state-of-the-art methods on a recently published tracking data-set ranking 3rd among 30 tracking methods. In addition, on a second challenging data-set, of vehicles undergoing severe view point changes, ELK came in first outperforming 7 other tracking methods.

ELKs performance can be further improved through better occlusion reasoning and explicit handling of illumination variations which is currently a weak spot for the algorithm.

References

1. Appel, R., Fuchs, T., Dollar, P., Perona, P.: Quickly boosting decision trees a pruning underachieving features early. In: ICML (2013)
2. Avidan, S.: Support vector tracking. PAMI (2004)
3. Avidan, S.: Ensemble tracking. In: CVPR (2005)
4. Babenko, B., Yang, M.H., Belongie, S.: Visual tracking with online multiple instance learning. In: CVPR (2009)
5. Bao, C., Wu, Y., Ling, H., Ji, H.: Real time robust l1 tracker using accelerated proximal gradient approach. In: CVPR (2012)
6. Comaniciu, D., Ramesh, V., Meer, P.: Real-time tracking of non-rigid objects using mean shift. In: CVPR (2000)
7. Cootes, T., Edwards, G., Taylor, C.: Active appearance models. TPAMI (2001)
8. DeGroot, M.: Optimal Statistical Decisions. McGraw-Hill, New York (1970)
9. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society 39(1), 1–38 (1977)
10. Dinh, T.B., Vo, N., Medioni., G.: Context tracker: Exploring supporters and distracters in unconstrained environments. In: CVPR (2011)
11. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via online boosting. In: BMVC (2006)
12. Hare, S., Saffari, A., Torr., P.H.S.: Struck: Structured output tracking with kernels. In: ICCV (2011)
13. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012)
14. Jia, X., Lu, H., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: CVPR (2012)
15. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. TPAMI (2010)
16. Kwon, J., Lee, K.M.: Visual tracking decomposition. In: CVPR (2010)
17. Kwon, J., Lee, K.M.: Tracking by sampling trackers. In: ICCV (2011)
18. Liu, B., Huang, J., Yang, L., Kulikowsk, C.: Robust tracking using local sparse appearance model and k-selection. In: CVPR (2011)
19. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proccedings of Imageing Understanding Workshop (1981)
20. Matthews, I., Baker, S.: Lucas-kanade 20 years on: A unifying framework. IJCV (2004)
21. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. TPAMI (2004)
22. Oron, S., Bar-Hillel, A., Avidan, S.: Real time tracking-with-detection. Submitted to Machine Vision and Applications (2014)
23. Oron, S., Hillel, A.B., Levi, D., Avidan, S.: Locally orderless tracking. In: CVPR (2012)

24. Ross, D., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. IJCV (2007)
25. Stauffer, C., Grimson, E.: Learning patterns of activity using real-time tracking. PAMI (2000)
26. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: CVPR (2013)
27. Xiang, Y., Song, C., Mottaghi, R., Savarese, S.: Monocular multiview object tracking with 3d aspect parts. In: European Conference on Computer Vision, ECCV (2014)
28. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM. Comp. Survey 38(4) (2006)
29. Zhang, K., Zhang, L., Yang, M.-H.: Real-time compressive tracking. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part III. LNCS, vol. 7574, pp. 864–877. Springer, Heidelberg (2012)
30. Zhong, W., Lu, H., Yang, M.H.: Robust object tracking via sparsity-based collaborative model. In: CVPR (2012)