Real-Time Tracking Using Trust-Region Methods

Tyng-Luh Liu, Hwann-Tzong Chen

Abstract— **Optimization methods based on iterative schemes can be divided into two classes: line-search methods and trust-region methods. While line-search techniques are commonly found in various vision applications, not much attention is paid to trust-region ones. Motivated by the fact that line-search methods can be considered as special cases of trust-region methods, we propose to establish a trust-region framework for real-time tracking. Our approach is characterized by three key contributions. First, since a trust-region tracking system is more effective, it often yields better performances than the outcomes of other trackers that rely on iterative optimization to perform tracking, e.g., a line-search based mean-shift tracker. Second, we have formulated a representation model that uses two coupled weighting schemes derived from the covariance ellipse to integrate an object's color probability distribution and edge density information. As a result, the system can address rotation and non-uniform scaling in a continuous space, rather than working on some presumably possible discrete values of rotation angle and scale. Third, the framework is very flexible in that a variety of distance functions can be adapted easily. Experimental results and comparative studies are provided to demonstrate the efficiency of the proposed method.**

Keywords— **Tracking, vision, iterative optimization, trust-region methods.**

I. Introduction

A key component of a successful tracking system is its ability to *search efficiently* for the target. Focusing on this goal, we propose a new approach for tracking using *trust-region methods* [6]. Previous uses of trust-region have been in areas other than real-time tracking, e.g., [12], [13]. While the applications are different, the efficiency of trust-region methods as an optimization tool has been demonstrated. Recently, Chen and Liu [4] have applied trust-region to tracking, and Sminchisescu and Triggs [16] have used them for 3-D body tracking.

A. Our Approach

We view a tracking process as a sequence of iterative optimization problems: For each image frame the task is to find an optimal solution that best describes the status of a target object. It requires an effective method to solve the underlying optimization problem appropriately. Most of the iterative optimization techniques used in tracking as well as other vision research are *line-search* in that the iterates are restricted to some iteration-dependent directions, e.g., the gradient. We instead use trust-region methods for their efficiency and reliability.

In addition, motivated by [5], we formulate a flexible object representation. It integrates both color and edge information via two coupled weighting schemes derived from a *covariance ellipse* model. Unlike other previous related works [2], [5], where the values of scale are limited to few pre-determined ones, the representation allows a system to perform optimization over a *continuous* space to yield better performance.

T.-L. Liu and H.-T. Chen are with the Institute of Information Science (IIS), Academia Sinica, Taipei, Taiwan. E-mail: {liutyng, pras}@iis.sinica.edu.tw. H.-T. Chen is also a Ph.D. student at the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.

B. Previous Work

Methods based on Bayesian framework have been playing an important role in tracking, e.g., [11], [15], [18]. Among them, the Condensation algorithm [11], introduced by Isard and Blake to track contours in clutter via *factored sampling*, is perhaps the most well-known one. Its main idea is to pinpoint the inappropriateness of the Gaussian state density assumption for tracking in clutter while multiple competing observations exist.

It is also possible to track objects of more complicated shapes using a learning approach [1], [7], [8], [9], [17]. Different from Condensation, Freedman and Brandstein [7], [8] consider the contour-tracking problem without assuming any dynamical model. They establish, via learning, a *subset tracker* to perform tracking through minimization. Exemplar-based methods [9], [17] require an off-line learning phase to generate object representations from examples, and then use distance measures to perform template matching.

If the objects to be tracked are non-rigid, it is convenient to represent them with probability distributions. A straightforward way to derive a distribution model is through *histogram analysis* [2], [3], [4], [5]. Birchfield [2] has proposed an algorithm to track a person's head by modeling it as a vertical ellipse with a fixed aspect ratio. In [3], Bradski presents a CAMSHIFT (continuously adaptive mean shift) system for use in a perceptual user interface to track faces. Comaniciu et al. [5] have used the mean shift to track non-rigid objects. They model objects by color distributions, and then measure the similarity between the target and candidate distributions using a Bhattacharyya coefficient. Note that the mean-shift technique is indeed a line search, and later we will discuss the comparisons between the mean-shift and our approach.

II. Trust-Region Methods

Iterative algorithms for optimization can be divided into two classes: line-search and trust-region. For a line-search one, the iterates are determined along some specific directions, e.g., *steepest descent* locates its iterates by considering the gradient directions. A trust-region method, however, derives its iterates by solving the corresponding optimization problem in a bounded region iteratively. So, there are more options to select the iterates. In fact, line-search methods can be considered special cases of trust-

region methods [6].

The concept of trust-region methods can be better understood by considering a typical unconstrained minimization problem,

$$\min_{\mathbf{x}\in\mathbf{V}} f(\mathbf{x}), \tag{1}$$

where $\mathbf{V}$ is a vector space, and f is some objective function to be minimized.

Essentially, there are three elements to any trust-region method: (i) *trust-region radius*, to determine the size of a trust region; (ii) *trust-region subproblem*, to approximate a minimizer in the region; and (iii) *trust-region fidelity*, to evaluate the accuracy of an approximating solution.

To illustrate, suppose an initial guess $\mathbf{x}_0$ and an initial trust-region radius $\triangle_0 > 0$ are given, and let η_1 and η_2 be some constants satisfying $0 < \eta_1 \leq \eta_2 < 1$. For each iteration $k \geq 0$, we first define, for the vector space $\mathbf{V}$, an iteration-dependent norm $\|\cdot\|_k$ and an iteration-dependent inner product $\langle\cdot\,,\cdot\rangle_k$ by

$$\|\mathbf{s}\|_k^2 = \langle \mathbf{s}, \mathbf{s}\rangle_k \stackrel{\text{def}}{=} \langle \mathbf{s}, M_k\mathbf{s}\rangle, \qquad \text{for any } \mathbf{s}\in\mathbf{V},$$

where $\langle\cdot\,,\cdot\rangle$ is the inner product, and M_k is an iteration-dependent matrix. (We will discuss how to determine M_k later.) Then at iteration k, with iterate $\mathbf{x}_k$ and trust-region radius $\triangle_k$, the following three steps are performed within the trust region $\mathcal{B}_k = \{\mathbf{x}\in\mathbf{V} \mid \|\mathbf{x}-\mathbf{x}_k\|_k \leq \triangle_k\}$.

1. Trust-region subproblem: We first construct a model m_k to approximate f in $\mathcal{B}_k$. In our system, a quadratic model is used for the approximation, i.e.,

$$m_k(\mathbf{x}_k+\mathbf{s}) = m_k(\mathbf{x}_k) + \langle g_k, \mathbf{s}\rangle + \frac{1}{2}\langle \mathbf{s}, H_k\mathbf{s}\rangle, \tag{2}$$

where $m_k(\mathbf{x}_k) = f(\mathbf{x}_k)$, $g_k = \nabla_{\mathbf{x}} f(\mathbf{x}_k)$, and H_k is the Hessian of f at $\mathbf{x}_k$. When $H_k \neq 0$, m_k is said to be a second-order model. A trust-region subproblem is then to compute an $\mathbf{s}_k$, where $\|\mathbf{s}_k\|_k \leq \triangle_k$, such that the model m_k is "sufficiently reduced," that is,

$$\mathbf{s}_k = \underset{\|\mathbf{s}\|_k\leq\triangle_k}{argmin}\ \psi_k(\mathbf{s}) \stackrel{\text{def}}{=} \langle g_k, \mathbf{s}\rangle + \frac{1}{2}\langle \mathbf{s}, H_k\mathbf{s}\rangle. \tag{3}$$

2. Trust-region fidelity: After solving a subproblem, the trial point $\mathbf{x}_k + \mathbf{s}_k$ will be tested to see if it is a good candidate for the next iterate. This is evaluated explicitly by

$$r_k = \frac{f(\mathbf{x}_k) - f(\mathbf{x}_k+\mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k+\mathbf{s}_k)}.$$

If $r_k \geq \eta_1$, then the trial point is accepted, i.e., $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$. Otherwise, $\mathbf{x}_{k+1} = \mathbf{x}_k$. Since η_1 is a small positive number, the above rule favors a trial point only when the value of the objective function f is also reduced. When m_k approximates f well and yields a large r_k, the trust-region radius will be expanded for the next iteration. On the other hand, if r_k is smaller than η_1 or r_k is negative, it suggests that the objective function f is not well approximated by the model m_k within the current trust region $\mathcal{B}_k$. Therefore, the iterate remains unchanged, and the trust-region radius will be shrunk to derive more appropriate model and subproblem for the next iteration.

3. Trust-region radius: More specifically, the trust-region radius can be updated as follows.

$$\triangle_{k+1} = \begin{cases} \max\{\alpha_1\|\mathbf{s}_k\|_k, \triangle_k\} & \text{if } r_k \geq \eta_2, \\ \triangle_k & \text{if } r_k \in [\eta_1, \eta_2), \\ \alpha_2\|\mathbf{s}_k\|_k & \text{if } r_k < \eta_1, \end{cases}$$

where, following [6, p.782], we have $\eta_1 = 0.05$, $\eta_2 = 0.9$, and $\alpha_1 = 2.5$, $\alpha_2 = 0.25$. The iterative optimization process for (1) will be repeated until the sequence of iterates $\{\mathbf{x}_k\}$ converges.

A. Trust-Region Scaled Norm

An objective function $f(\mathbf{x})$ may have variables whose typical values are of different orders of magnitude. For example, in real-time tracking, the values of spatial variables are often much larger than the vales of scale variables. Without re-scaling the variables properly, the contributions from variables of small values tend to be dominated by those from variables of large values. It can then lead to unexpected optimization results. To deal with such issues, the re-scaling will be done for each iteration k through a *nonsingular* matrix S_k to ensure every trust-region subproblem is solved in a reasonably scaled space. In particular, we have used nonsingular diagonal matrices S_ks, where the diagonal entries correspond to typical values of the respective variables. It follows that the new variables, say $\tilde{\mathbf{x}}$, in the scaled space are derived by $\tilde{\mathbf{x}} = S_k^{-1}\mathbf{x}$. As a result, $\tilde{\mathbf{x}}$ will be of comparable scales after the re-scaling. Moreover, as is proved in [6], it is not necessary to reformulate a trust-region subproblem using the new variables since *re-scaling the variables is equivalent to using an iteration-dependent scaled norm* defined by $\|\mathbf{s}\|_k^2 =< \mathbf{s}, M_k\mathbf{s} >=< \mathbf{s}, S_k^{-T}S_k^{-1}\mathbf{s} >$, where $M_k = S_k^{-T}S_k^{-1}$ is an iteration-dependent matrix.

B. Trust-Region vs. Line-Search

In our approach, we have used a quadratic model m_k for the implementation. If, instead, a linear model is used, then the RHS of (2) is reduced to the first two terms. This implies that a trust-region method with a linear model approximation is *almost* like gradient descent, but it often achieves better performances owing to its ability to adjust trust regions adaptively throughout the iterations. This is why line-search methods can be considered special cases of trust-region.

Both trust-region and line-search are guaranteed to converge to a local minimum. However, not all local minima are of interest for real application. Typical line-search, e.g., steepest descent or even trust-region with a linear model approximation may often converge to a local minimum that is *inferior* to a nearby one. In Fig. 1, we construct an objective function with three local minima, x_1, x_2, and x_3. Among them, x_1 is clearly the global minimum. We test the three schemes, using 1000 different initial positions, x_0s, sampled uniformly from $[56.96, 66.95]$. Though the

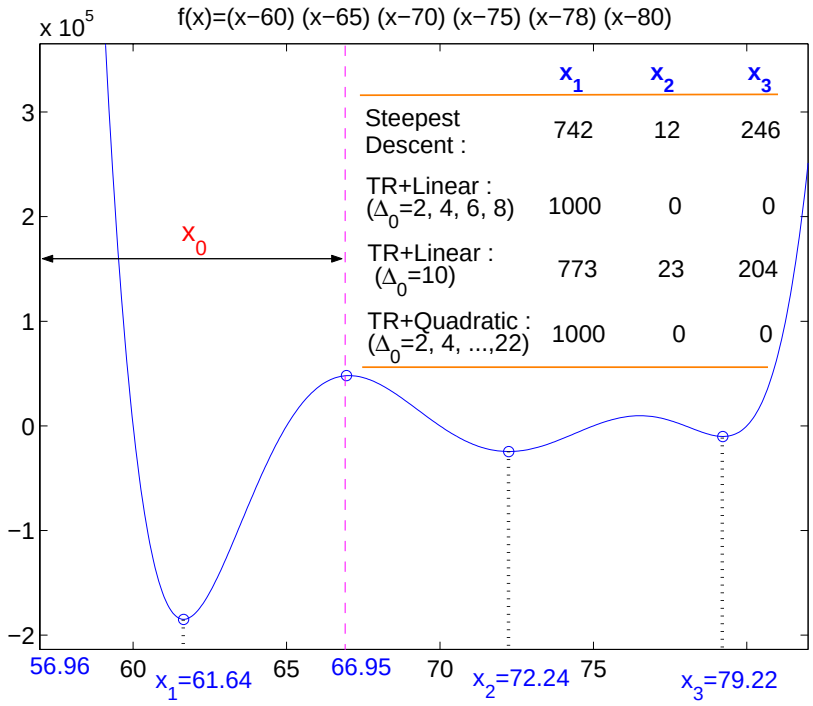


Fig. 1. Optimizing with Steepest Descent, TR+linear model, and TR+quadratic model. Out of 1000 runs, with initial positions x_0s, sampled uniformly from $[56.96, 66.95]$, we record in each entry the number of times that a method converges to a local minimum.

x_0s are indeed close to the global minimum x_1, steepest descent fails to converge to x_1 258 times. Trust-region methods are tested with different Δ_0, and are more successful in converging to x_1. In passing, note that the mean-shift technique in [5] is a more conservative line-search. Instead of taking largest/steepest steps along gradients, it usually progresses by small steps, computed from the information within *fixed-size* windows. Such an approach tends to converge to a nearby local minimum regardless of its significance. Thus, both a more sophisticated model approximation and a mechanism to iteratively adjust the regions of interest are needed to reduce the chance of converging to a local minimum not of interest.

III. Representations and Objective Functions

Motivated by the work of Comaniciu et al. [5], we also use probability distributions to represent targets. But unlike [5], where the analysis relies on kernel properties, we simply treat the color distribution as a *weighted color histogram* to account for the possible non-rigidity of objects.

A. Representation Models for Tracking

Tracking objects by distribution is efficient but not necessarily sufficient. Suppose the scale of a target object of monotone color is enlarged. Then, it is not guaranteed that the appropriate scale will always be recovered since, in this case, any sub-portion of the object has a similar distribution to that of the object. Other tracking cues are needed to elevate the performance, e.g., [14]. In our system, the representation model consists of two elements: the first is to characterize the RGB color distribution, and the second is to estimate the edge density near the object boundary. Since the edge density is contributed mainly from samples near the boundary, and more prone to be affected by the background, we choose color distribution as the primary cue for tracking.

A.1 Color Distribution

For computing the weighed color histogram, the RGB color space is first divided into n bins, and a bin assignment function b is defined uniquely by each pixel $\mathbf{x}_i$'s RGB value as $b : \mathbf{x}_i \mapsto \{1, \ldots, n\}$. We then formulate a color weighting scheme based on the *bivariate normal distribution*, defined by $\phi(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi|\boldsymbol{\Sigma}|^{1/2}} e^{-(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})/2}$, where $\mathbf{x} = (x_1, x_2)^T$, $\boldsymbol{\mu} = (\mu_1, \mu_2)^T$ is the mean vector, and $\boldsymbol{\Sigma}$ is the covariance matrix.

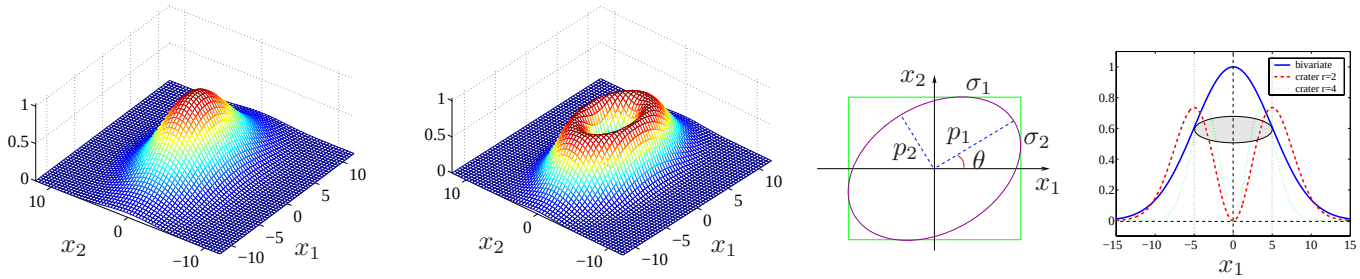
Let the correlation coefficient $\rho = \sigma_{12}/\sigma_1\sigma_2$. Then, when $|\rho| < 1$, the bivariate normal distribution can be rewritten as

$$\phi(\mathbf{x}; \boldsymbol{\zeta}) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left\{-\frac{\varepsilon(\mathbf{x}; \boldsymbol{\zeta})}{2}\right\}, \qquad (4)$$

where, to simplify the notations, we have $\boldsymbol{\sigma} = (\sigma_1, \sigma_2)^T$, $\boldsymbol{\zeta} = (\boldsymbol{\mu}, \boldsymbol{\sigma}, \rho) = (\mu_1, \mu_2, \sigma_1, \sigma_2, \rho)$, and

$$\varepsilon(\mathbf{x}; \boldsymbol{\zeta}) = \frac{1}{1-\rho^2}\left\{\frac{(x_1-\mu_1)^2}{\sigma_1^2} - 2\rho\frac{(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2}\right\}.$$

From (4), it implies that lines of constant ϕ correspond to constant exponents, i.e., $\varepsilon(\mathbf{x}; \boldsymbol{\zeta}) = constant$ represents an



(a) Color weights (b) Edge weights (c) Covariance ellipse (d) 1-D plot

Fig. 2. (a) Bivariate normal for color weights. (b) Crater function for edge weights. (c) A covariance ellipse can be represented either by (p_1, p_2, θ) or by $(\sigma_1, \sigma_2, \rho)$, where p_1, p_2 are lengths of the principal semi-diameters, and θ is the angle between the p_1 semi-diameter and the x_1 axis. (d) Peaks of a crater function occur near the loci of the coupled covariance ellipse.

ellipse centering at $\boldsymbol{\mu}$. We focus only on the *covariance ellipses*, which satisfy $\varepsilon(\mathbf{x};\boldsymbol{\zeta}) = 1$ (denoted as $\varepsilon_1(\boldsymbol{\zeta})$), to construct the color weighting scheme.

Now, let I^0 be the first image frame and $\boldsymbol{\zeta}^0 = (\boldsymbol{\mu}^0, \boldsymbol{\sigma}^0, \rho^0)$. Then, a target object initially centering at $\boldsymbol{\mu}^0$ can be associated with $A_1(\boldsymbol{\zeta}^0) = \{\mathbf{x} \mid \varepsilon(\mathbf{x};\boldsymbol{\zeta}^0) \leq 1\}$, the area enclosed by $\varepsilon_1(\boldsymbol{\zeta}^0)$. We define the target's color distribution within $A_1(\boldsymbol{\zeta}^0)$, denoted as $p(u;\boldsymbol{\zeta}^0)$, by

$$\begin{aligned} p(u;\boldsymbol{\zeta}^0) &= \frac{1}{C_p} \sum_{\mathbf{x}_i \in A_1(\boldsymbol{\zeta}^0)} w_c(\mathbf{x}_i;\boldsymbol{\zeta}^0)\delta(b(\mathbf{x}_i) - u) \\ w_c(\mathbf{x}_i;\boldsymbol{\zeta}^0) &= \exp\left\{-\frac{\varepsilon(\mathbf{x}_i;\boldsymbol{\zeta}^0)}{2}\right\}, \end{aligned} \tag{5}$$

where δ is the Kronecker delta function, and w_c is the derived color weighting function. That $p(u;\boldsymbol{\zeta}^0)$ is a probability implies $C_p = \sum_{\mathbf{x}_i \in A_1(\boldsymbol{\zeta}^0)} w_c(\mathbf{x}_i;\boldsymbol{\zeta}^0)$. For convenience, the notation $p(u;\boldsymbol{\zeta}^0)$ will be abbreviated into $p(u)$ since $\boldsymbol{\zeta}^0$ only describes the target's initial state. Analogously, during tracking, the color distribution of some $A_1(\boldsymbol{\zeta})$, denoted as $q(u;\boldsymbol{\zeta})$, is

$$q(u;\boldsymbol{\zeta}) = \frac{1}{C_q} \sum_{\mathbf{x}_i \in A_1(\boldsymbol{\zeta})} w_c(\mathbf{x}_i;\boldsymbol{\zeta})\delta(b(\mathbf{x}_i) - u),$$

where C_q is the total weight such that $\sum_{u=1}^{n} q(u;\boldsymbol{\zeta}) = 1$.

A.2 Edge Density

For every w_c in (5), we can use a *crater function* to define a coupled edge-point weighting function w_e. Specifically, we have $w_e(\mathbf{x}_i;\boldsymbol{\zeta}) = \gamma\,\varepsilon(\mathbf{x}_i;\boldsymbol{\zeta}) \exp\left\{-\frac{\gamma}{2}\varepsilon(\mathbf{x}_i;\boldsymbol{\zeta})\right\}$, where γ is the parameter to adjust the shape of a crater function and the size of a crater's *opening*. It can be verified by a straightforward calculation that for $\gamma = 2$, the peaks of the level surface of a crater function occur at the loci of the associated covariance ellipse as shown in Fig. 2d. In practice, we find better tracking performance can be achieved by using a slightly larger γ, say $\gamma = 4$, in that significant values of edge weight are within the covariance ellipse.

Finally, we adopt the notation $e(\boldsymbol{\zeta})$ to represent the edge density within and near the boundary of a covariance elliptic region $A_1(\boldsymbol{\zeta})$. The *scale-invariant* definition of $e(\boldsymbol{\zeta})$ is as follows.

$$e(\boldsymbol{\zeta}) = \frac{1}{\sigma_1\sigma_2} \sum_{\mathbf{x}_i \in A_1(\boldsymbol{\zeta})} w_e(\mathbf{x}_i;\boldsymbol{\zeta})E(\mathbf{x}_i), \tag{6}$$

where $E(\mathbf{x}_i)$ is a binary edge map, derived from a high-pass 5×5 Laplacian filter in [10].

B. Objective Functions for Tracking

In view of the object representation model just described, a tracking process for an arbitrary target can be characterized then by evolution dynamics of a covariance ellipse, $\varepsilon_1(\boldsymbol{\zeta}^t)$, where we simply denote the process as $\boldsymbol{\zeta}^0 \rightarrow \boldsymbol{\zeta}^1 \rightarrow \boldsymbol{\zeta}^2 \rightarrow \cdots$. To decide an *optimal* $\boldsymbol{\zeta}^t$ for each frame I^t, we still need to formulate an appropriate objective function to complete the framework.

Since there are two rather distinct features included in the representation model, the resulting objective functional must address these two factors justifiably. First, to measure the similarity between two color distributions, we consider the *Kullback-Leibler distance*, i.e.,

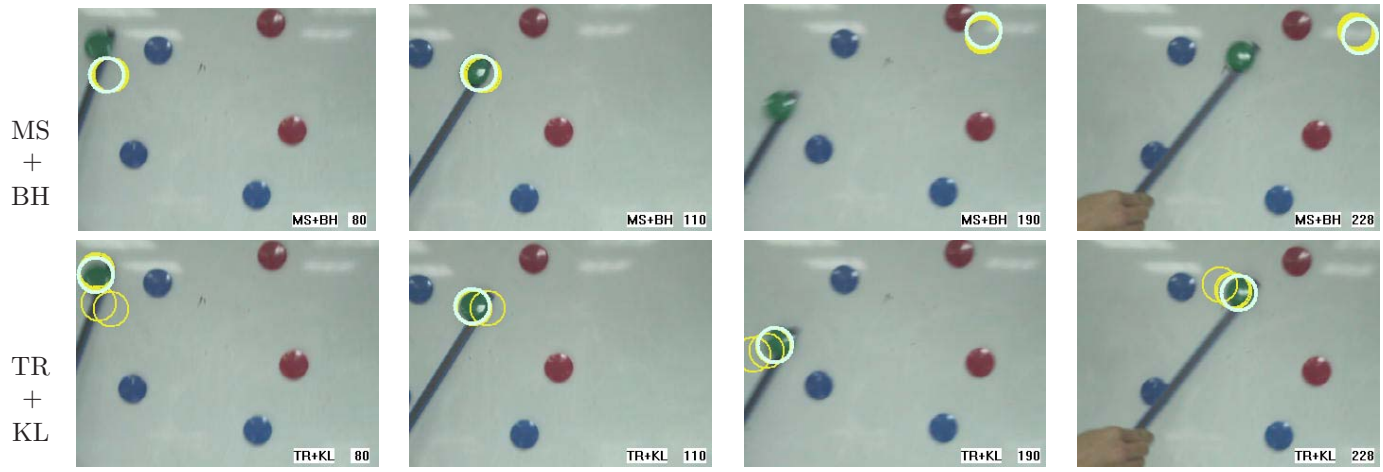
$$f_c(\boldsymbol{\zeta}) = \sum_{u=1}^{n} p(u) \log \frac{p(u)}{q(u;\boldsymbol{\zeta})}, \tag{7}$$

where $p(u)$ is the target (true) color distribution and $q(u;\boldsymbol{\zeta})$ is the one for the covariance ellipse $\varepsilon_1(\boldsymbol{\zeta})$. Second, to estimate whether the boundary edge density of a candidate covariance ellipse is comparable to the one of target's, we embed the edge density ratio, denoted as $h(\boldsymbol{\zeta}) = e(\boldsymbol{\zeta})/e(\boldsymbol{\zeta}^0)$, into a *sigmoid function* to derive the following,

$$\begin{aligned} f_e(\boldsymbol{\zeta}) &= 1 - \frac{1}{1 + \exp\{-\alpha(h(\boldsymbol{\zeta}) - \beta)\}} \\ &= \frac{1}{1 + \exp\{\alpha(h(\boldsymbol{\zeta}) - \beta)\}}, \end{aligned} \tag{8}$$

where α and β are parameters for setting up the initial sigmoid function. (We use $\alpha = 5$ and $\beta = 1$ for all the experiments.) Finally, with the definitions in (7) and (8), the underlying optimization problem for each image frame I^t can be formally written as

$$\boldsymbol{\zeta}^t = \underset{\boldsymbol{\zeta} \in \Omega^t}{argmin}\, f(\boldsymbol{\zeta}) = f_c + \lambda\, f_e, \tag{9}$$



(a) *Magnet* #080 (b) *Magnet* #110 (c) *Magnet* #190 (d) *Magnet* #228

Fig. 3. TR: Trust-Region, MS: Mean-Shift, BH:Bhattacharyya, and KL: Kullback-Leibler. In each image, the final convergent circle is plotted in white and the intermediate ones in yellow.

where λ is a parameter to weigh the relative importance of the two terms, and Ω^t is the space consisting of all possible $\boldsymbol{\zeta}$'s for any combinations of translation, scale, and orientation.

IV. Experimental Results and Discussion

We demonstrate the efficiency of our method by (i) making comparisons with a mean-shift tracker [5], and (ii) carrying out a variety of experiments of different scenarios.

A. Trust-Region vs. Mean-Shift

In [5], the color distribution is used as the only cue for tracking, and the Bhattacharyya coefficient, defined by $\sum_{u=1}^{n}\sqrt{p(u)q(u;\mathbf{x})}$, is chosen to be the objective function to be maximized. Since a mean-shift vector is simply to approximate the gradient of an objective function, thus for the sake of comparison, we implement a trust-region tracker with a linear model approximation, and use the exact color representation model described in [5] for all comparisons. This implies we are dealing with two trackers: trust-region (TR) and mean-shift (MS), and two objective functions: Kullback-Leibler distance (KL) and Bhattacharyya coefficient (BH). Totally there are four possible combinations: MS+BH, TR+BH, MS+KL, and TR+KL.

In Fig. 3, we show some of the results obtained by using MS+BH and TR+KL, respectively. The main advantage of experimenting with such a sequence is that the resulting level surfaces are mostly smooth but with sporadic local extrema. Thus, it is easier to pinpoint the causes of different outcomes. We also examine the values of objective functions explicitly. To do so, we randomly generate 500 initial positions for an arbitrary image frame from the *Magnet sequence*, then perform optimizations from each position using MS+BH and TR+BH, respectively. The same process is repeated for MS+KL and TR+KL. We then count the number of occurrences of converging to a better objective function value by trust-region. This quantitative analysis is also performed for the other three sequences shown in Fig. 5. Our results, in Fig. 4, indicate that no matter which objective function is used, BH or KL, the probability that a trust-region tracker is more effective is about 90%, by converging to better values 3675 times out of 4000 tests. Note that the efficiency can be further improved by using a quadratic model approximation.

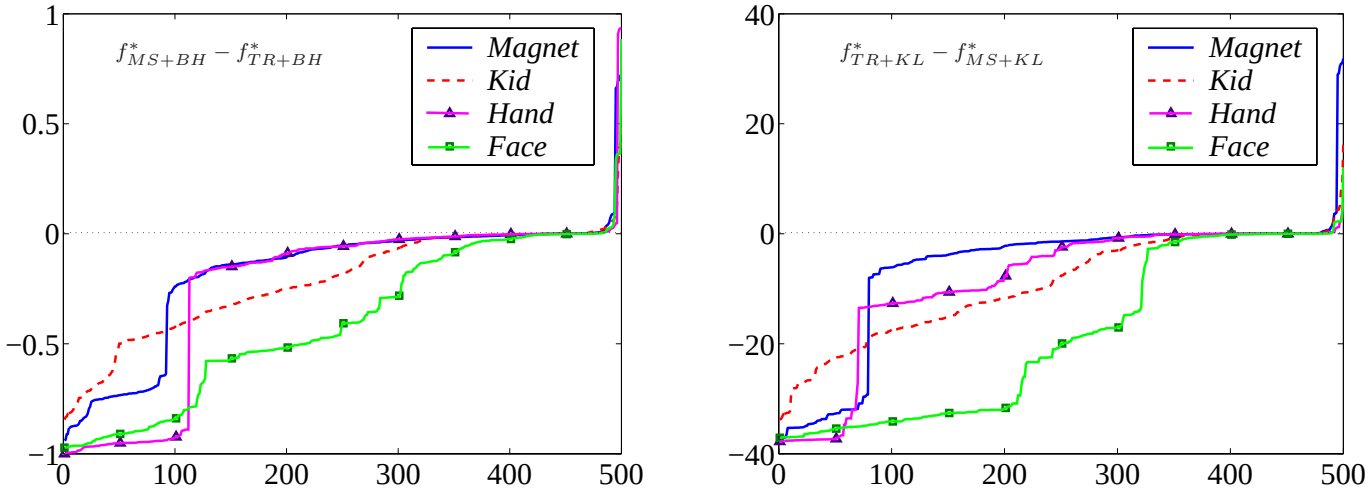
B. Tracking by TR+KL+Edge

We turn now our attention to experimenting with the complete algorithm, i.e., using trust-region with a quadratic model to preform tracking via optimizing with (9). In all our experiments, the RGB space is divided into $16 \times 16 \times 16 = 4096$ bins. Other parameters used include: $\lambda = 0.2$, initial trust-region radius $\Delta_0 = 4$, and typical values of the diagonal of S_k are $(10, 10, 1, 1, 0.1)$. The experiments are carried out on a Pentium-4 2.4GHz PC.

The first sequence is to show the tracker's ability to pursue a fast moving object (a kid jumping around), accounting for the 2-D translation factor only. Note that, in Fig. 5a - 5d, the intermediate iterates/ellipses are plotted in green to illustrate the underlying optimization process. In the second experiment, we demonstrate, in Fig. 5e - 5h, the effectiveness of optimizing over a 5-dimensional continuous space to capture various changes in the object's scale, shape, and orientation. We emphasize that if a system has a status variable $\boldsymbol{\zeta}$ limited to just some pre-determined discrete values of scale and orientation, it generally could not deliver a comparable performance. The third test is performed using a pan/tilt/zoom camera where the target person in the scene moves back and forth to bring about rapid and substantial changes in the size of the face appeared in *Face sequence*. While most tracking-by-distribution systems cannot handle such difficulties, our method addresses the issues of scales robustly as shown in Fig. 5i - 5l.

C. Complexity Analysis

Since the algorithm is iterative, and it typically takes just few iterations to converge, it suffices to analyze the



(a) TR+Linear+BH vs. MS+BH (b) TR+Linear+KL vs. MS+KL

Fig. 4. Sorted differences in the objective function values derived by TR+Linear and MS.

time complexity for one iteration. For frame t, let m be the number of pixels within $A_1(\boldsymbol{\zeta}^t)$ and d be the dimensionality of $\boldsymbol{\zeta}$. (In our formulation, $d = 5$.) We first need to compute the color histogram $q(u; \boldsymbol{\zeta}^t)$, the edge density $e(\boldsymbol{\zeta}^t)$ in (6), the gradient g_k, and the Hessian matrix H_k, which it takes $O(m)$, $O(m)$, $O(dm)$, and $O(d^2m)$ time, respectively. Next, it takes $O(n)$ time to evaluate f_c by summing up $p(u) \log \frac{p(u)}{q(u)}$, and $O(1)$ time for f_e. (Recall that n is the number of color bins.) Finally, for solving a trust-region subproblem, since the number of iterations is assumed to be less than a fixed number, the time complexity only depends on dimensionality d. In particular, to find a minimizer of the subproblem, we have to compute ψ_k and $\|\mathbf{s}_k\|_k$ in (3), or find the intersection on the region boundary. The first requires $O(d^2)$ time, and the latter two take $O(d)$ computation time. When H_k is non-convex, we need extra $O(d^2)$ time to find the other possible iterate. Therefore, the time complexity for one iteration of the complete TR tracking algorithm is $O(d^2m + n)$.

D. Discussion

Our approach focuses mainly on two important issues: *optimization* and *representation*. Specifically, we have discussed three choices for optimization: line-search, trust-region with a linear-model approximation, and trust-region with a quadratic-model approximation. While the three all have the desired property to converge to a local minimum, we investigate the *quality* of a solution. To emphasize, we note that a line-search method may fail to converge to a better, nearby extremum due to a crude approximation to the local shape of an objective function. We then compare our method with a well-known mean-shift tracker to demonstrate the advantages of being able to find the iterates in a region and to adjust the size of the region adaptively. Nonetheless, it is difficult to evaluate quantitatively the performances of two different tracking methods because when testing with a video sequence, they often start at different initial positions for each intermediate image frame. Thus, we instead do the quantitative analysis for one arbitrary image frame with randomly generated starting positions. This is equivalent to solving an iterative optimization problem using the two methods, respectively, for each initial value. Such modifications make it possible to analyze the results explicitly, and to further verify that a trust-region implementation for tracking is often more reliable and effective than a line-search one.

Other efforts have been made to design a good representation model. We have formulated a covariance-ellipse representation to integrate color and edge density information. It enables the system to perform optimization over a continuous space to yield more accurate results. Our future work includes extending the framework for multiple-object tracking, and exploiting other possible applications in computer vision using trust-region methods.

Acknowledgments

This work was supported by NSC grants 90-2213-E-001-016 and 91-2213-E-001-023, and in part by the Institute of Information Science, Academia Sinica of Taiwan. H.-T. would like to thank the Foundation for the Advancement of Outstanding Scholarship for a student travel grant.

References

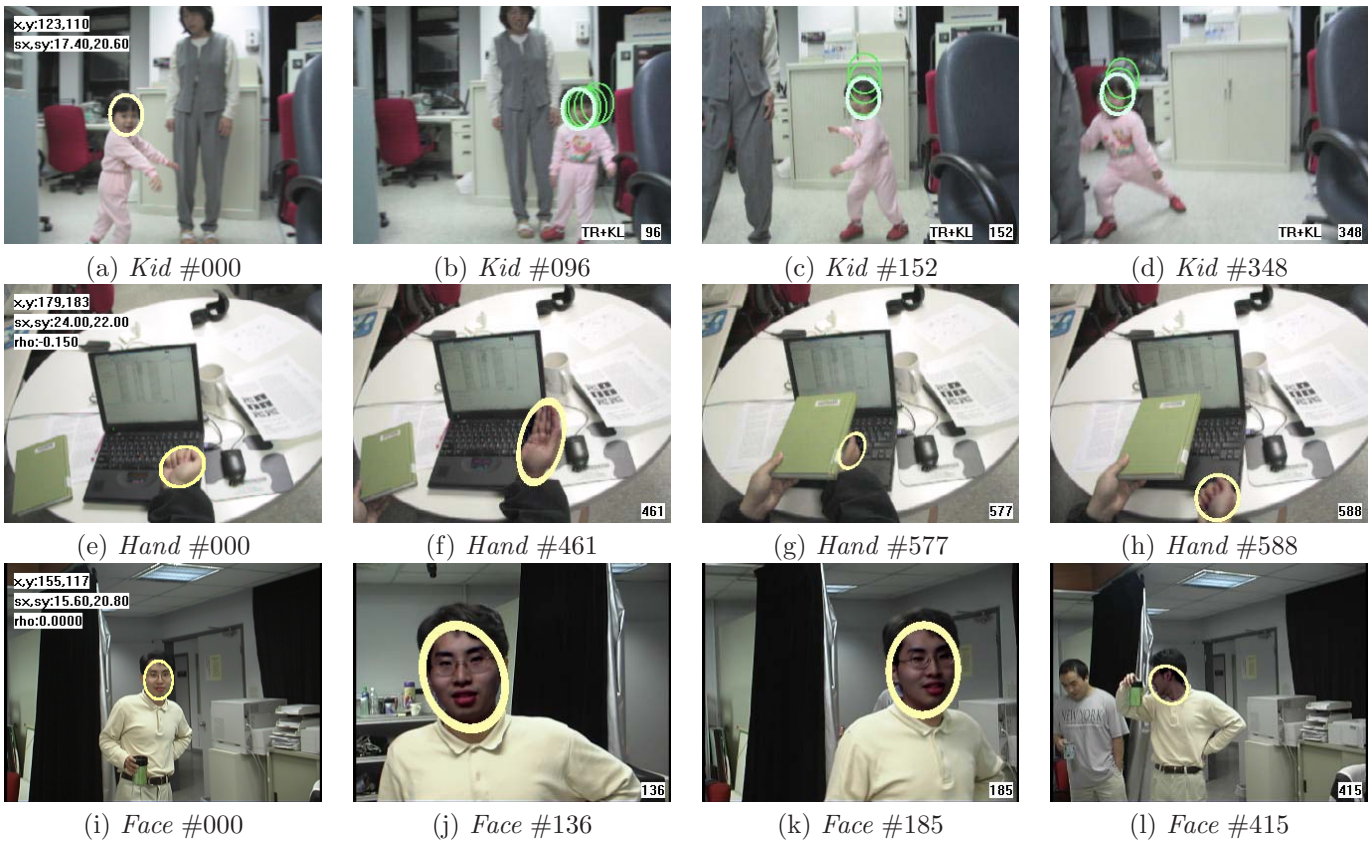
[1] S. Avidan, "Support Vector Tracking," *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 184–191, Kauai, Hawaii, 2001.

[2] S.T. Birchfield, "Elliptical Head Tracking Using Intensity Gradients and Color Histograms," *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 232–237, Santa Barbara, CA, 1998.

[3] G.R. Bradski, "Computer Vision Face Tracking for Use in a Perceptual User Interface," *Intel Technology Journal*, 1998.

[4] H.T. Chen and T.L. Liu, "Trust-Region Methods for Real-Time Tracking," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 717–722, Vancouver, Canada, 2001.

[5] D. Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 142–149, Hilton Head Island, South Carolina, 2000.



(a) *Kid* #000 (b) *Kid* #096 (c) *Kid* #152 (d) *Kid* #348

(e) *Hand* #000 (f) *Hand* #461 (g) *Hand* #577 (h) *Hand* #588

(i) *Face* #000 (j) *Face* #136 (k) *Face* #185 (l) *Face* #415

Fig. 5. (a)-(d) *Kid sequence*: track a target with rapid motion (frame rate: $>$ 200fps). (e)-(h) *Hand sequence*: track a target with substantial changes in size, shape, and orientation (frame rate: 35fps). (i)-(l) *Face sequence*: the strength of a trust-region tracker is even more appreciable where a wide range of scales of target face are tracked properly (frame rate: 20fps).

[6] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Trust-Region Methods*, SIAM, 2000.
[7] D. Freedman and M.S. Brandstein, "Contour Tracking in Clutter: A Subset Approach," *Int'l J. Computer Vision*, vol. 38, no. 2, pp. 173–186, July 2000.
[8] D. Freedman and M.S. Brandstein, "Provably Fast Algorithms for Contour Tracking," *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 139–144, Hilton Head Island, South Carolina, 2000.
[9] D. Gavrila and V. Philomin, "Real-Time Object Detection for Smart Vehicles," *Proc. Seventh IEEE Int'l Conf. Computer Vision*, pp. 87–93, Corfu, Greece, 1999.
[10] Intel Corporation, *Intel Image Processing Library Reference Manual*, 2000, Document Number 663791-005.
[11] M. Isard and A. Blake, "Contour Tracking by Stochastic Propagation of Conditional Density," *Proc. Fourth European Conf. Computer Vision*, vol. 1, pp. 343–356, Cambridge, England, 1996.
[12] M. Jagersand, O. Fuentes, and Nelson R. C., "Experimental Evaluation of Uncalibrated Visual Servoing for Precision Manipulation," *Proc. 1997 Int'l Conf. Robotics and Automation*, Albuquerque, NM, 1997.
[13] T.Q. Phong, R. Horaud, A. Yassine, and P.D. Tao, "Object Pose from 2-D to 3-D Point and Line Correspondences," *Int'l J. Computer Vision*, vol. 15, no. 3, pp. 225–243, July 1995.
[14] C. Rasmussen and G.D. Hager, "Probabilistic Data Association Methods for Tracking Complex Visual Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 560–576, June 2001.
[15] H. Sidenbladh and M.J. Black, "Learning Image Statistics for Bayesian Tracking," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 709–716, Vancouver, Canada, 2001.
[16] C. Sminchisescu and B. Triggs, "Covariance Scaled Sampling for Monocular 3D Body Tracking," *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 447–454, Kauai, Hawaii, 2001.
[17] K. Toyama and A. Blake, "Probabilistic Tracking in a Metric Space," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 50–57, Vancouver, Canada, 2001.
[18] Y. Wu and T.S. Huang, "A Co-inference Approach to Robust Visual Tracking," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 26–33, Vancouver, Canada, 2001.