Multiple Kernel Tracking with SSD

Gregory D. Hager and Maneesh Dewan
Johns Hopkins University
Baltimore, MD 21218
{hager,maneesh}@cs.jhu.edu

Charles V. Stewart
Renssalaer Polytechnic Institute
Troy, NY 12180
stewart@cs.rpi.edu

Abstract

Kernel-based objective functions optimized using the mean shift algorithm have been demonstrated as an effective means of tracking in video sequences. The resulting algorithms combine the robustness and invariance properties afforded by traditional density-based measures of image similarity, while connecting these techniques to continuous optimization algorithms.

This paper demonstrates a connection between kernel-based algorithms and more traditional template tracking methods. There is a well known equivalence between the kernel-based objective function and an SSD-like measure on kernel-modulated histograms. It is shown that under suitable conditions, the SSD-like measure can be optimized using Newton-style iterations. This method of optimization is more efficient (requires fewer steps to converge) than mean shift and makes fewer assumptions on the form of the underlying kernel structure. In addition, the methods naturally extend to objective functions optimizing more elaborate parametric motion models based on multiple spatially distributed kernels. We demonstrate multi-kernel methods on a variety of examples ranging from tracking of unstructured objects in image sequences to stereo tracking of structured objects to compute full 3D spatial location.

1 Introduction

Kernel-based methods for computer vision have received significant attention since they were first introduced several years ago [12]. Recently, these methods have been introduced to solve visual tracking problems involving location [3] and location and scale [8]. These techniques track a target region that is described as a spatially-weighted intensity histogram. An objective function that compares target and candidate kernel densities is formulated using the Bhattacharyya measure, and tracking is achieved by optimizing this objective function using the mean shift algorithm. Experimental results have shown the promise of kernel-based tracking methods in a wide range of contexts.

Intuitively, kernel-based descriptions of tracking regions are attractive because they combine summary descriptions of both intensity values and spatial positions in a way that avoids the need for complex modeling of object shape, appearance or motion. Conceptually, they forge a link between statistical measures of similarity (which historically required brute force optimization) with powerful methods from continuous optimization.

The underlying assumption in kernel-based tracking is that a statistical summary in terms of a kernel-weighted feature histogram is sufficient to determine location, and is sufficiently insensitive to other motions to be robust. This raises a question as to what motions can and cannot be recovered using kernel-based methods. For example, rotational motion cannot be estimated with current kernel-based methods because only rotationally-symmetric kernels have been used. On the other hand, most kernels are not scale-invariant, requiring some apparatus to deal with scaling of the target.

In order to gain an understanding of the performance and performance limitations of current kernel-based methods, we use the equivalent SSD form of the original Bhattacharyya metric. We then derive a Newton-style minimization procedure on this measure. The structure of this optimization makes explicit the limitations in kernel-density tracking. These limitations arise both from the structure of the kernel alone and from interactions between the kernel and the image spatial structure. We also show empirically that the Newton-style formulation is a more efficient optimization than mean shift which is fundamentally a gradient descent algorithm.

This analysis provides the basis for considering the design of an expanded set of kernel-based trackers using one or more modified kernels. Intuitively, multiple kernels increase the measurement space and, in doing so, increase the sensitivity of a kernel-based tracking algorithm. The SSD measure we develop extends naturally to multiple kernels, as does the associated optimization procedure. We show designs for kernels that attend to particular types of image motions, including scale and rotation, and to particular types of image intensity structures, including generic region properties such as symmetry. These are demonstrated on several

motion sequences. Comparative results with the mean shift algorithm are also provided.

2 Mean Shift Tracking of Location

Here, we review the basic concepts of kernel-based tracking using the terminology and notation similar to [3, 11].

Consider a target chosen for tracking as a defined region of pixel locations $\{\mathbf{x}_i\}_{i=1\ldots n}$ in an image I with time index t. For each pixel location, a feature vector $\mathbf{f} \in \mathscr{F}$ characterizes the appearance within some neighborhood of that pixel location. Let $\mathscr{U} = 1\ldots m$ represent a finite number of feature "bins," and let $b : \mathscr{F} \to \mathscr{U}$ denote the "binning" function on features. For simplicity, the expression $b(\mathbf{x},t)$ will represent the bin of the feature value of location $\mathbf{x}$ in an image with time index t. Let $K : \Re^2 \to \Re^+$ denote a kernel which weights image locations.

With these definitions, a kernel-weighted empirical distribution, or histogram, $\mathbf{q} = (q_1, q_2, ..., q_m)^t$ of a target region can be computed as:

$$q_u = C\sum_{i=1}^{n} K(\mathbf{x}_i - \mathbf{c})\delta(b(\mathbf{x}_i,t),u) \tag{2.1}$$

$$C = \frac{1}{\sum_{i=1}^{n} K(\mathbf{x}_i - \mathbf{c})} \tag{2.2}$$

where δ is the Kronecker delta function and $\mathbf{c}$ is the kernel center location. Note that the definition of C implies that $\sum_{u=1}^{m} q_u = 1$. Unless otherwise noted, we subsequently consider only kernels that have been suitably normalized so that $C = 1$.

Equation (2.1) can be written more compactly by defining, for each feature value u, a corresponding *sifting vector* $\mathbf{u}$ as $\mathbf{u}_i = \mathbf{u}_i(t) = \delta(b(\mathbf{x}_i,t),u)$. We can combine these sifting vectors into an n by m *sifting matrix* $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m]$. Similarly, we can define a vector version of the kernel function $\mathbf{K}$ by $\mathbf{K}_i(\mathbf{c}) = K(\mathbf{x}_i, \mathbf{c})$. With this, we can now rewrite (2.1) in a more concise form:

$$\mathbf{q} = \mathbf{U}^t\mathbf{K}(\mathbf{c}) \tag{2.3}$$

Suppose we are now given a candidate region centered about $\mathbf{c}$ in a subsequent image acquired at time t'. The corresponding empirical feature distribution would be

$$\mathbf{p}(\mathbf{c}) = \mathbf{p}(\mathbf{c},t') = \mathbf{U}^t(t')\mathbf{K}(\mathbf{c}) \tag{2.4}$$

The location tracking problem can now be stated as follows: given a model distribution, $\mathbf{q}$, and a candidate distribution, $\mathbf{p}(\mathbf{c})$, choose a location $\mathbf{c}^*$ that maximizes the similarity between the target distribution and the model distribution.

In [3], the location estimation problem is solved by optimizing the sample estimate of the Bhattacharyya coefficient:

$$\hat{\rho}(\mathbf{c}) \equiv \hat{\rho}(\mathbf{p}(\mathbf{c}),\mathbf{q}) = \sum_{u=1} \sqrt{p_u(\mathbf{c})q_u}, \tag{2.5}$$

In the notation developed above, this expression can be written

$$\sum_{u=1}^{m} \sqrt{p(\mathbf{c})q} = \sum_{u=1}^{m} \sqrt{p(\mathbf{c})}\sqrt{q} = \sqrt{\mathbf{p}(\mathbf{c})} \cdot \sqrt{\mathbf{q}}$$

where the square root operator is taken to apply componentwise to the vector argument.

At this point, the following additional assumptions are required: 1) $K(\mathbf{x} - \mathbf{c}) = k(\|\mathbf{x} - \mathbf{c}\|^2)$; 2) k is non-negative[1] and non-increasing; 3) k is piecewise differentiable [12]. Under these assumptions, the mean shift algorithm is then derived in two steps. The first is to expand the above expression in a Taylor series about $\mathbf{p}$. Doing so and rearranging terms yields the following expression to be maximized:

$$O(\mathbf{c}) = \sum_{i=1}^{n} w_i K(\mathbf{x}_i - \mathbf{c}) \tag{2.6}$$

$$w_i = \sum_{u=1}^{m} \frac{\sqrt{q_u}}{\sqrt{p_u(\mathbf{c})}}\delta(b(\mathbf{x}_i,t),u) \tag{2.7}$$

In the vector notation developed above, this becomes

$$\mathbf{w} = \mathbf{U}\left(\frac{\sqrt{\mathbf{q}}}{\sqrt{\mathbf{p}(\mathbf{c})}}\right) \tag{2.8}$$

$$O(\mathbf{c}) = \mathbf{w}^t\mathbf{K}(\mathbf{c}) \tag{2.9}$$

where / is again taken to apply componentwise to the associated vectors.

This optimization is then solved by computing the gradient and setting it equal to zero. The final solution can be written in the form of a weighted mean:

$$\mathbf{c}^* - \mathbf{c} = \triangle\mathbf{c} = \frac{\sum_{i=1}^{n}(\mathbf{x}_i - \mathbf{c})w_i g(\|\mathbf{x}_i - \mathbf{c}\|^2)}{\sum_{i=1}^{n} w_i g(\|\mathbf{x}_i - \mathbf{c}\|^2)} \tag{2.10}$$

where $g(x) = -k'(x)$.

It can be shown that the series of "mean shifts" computed using this rule is seeking the mode of the kernel-weighted distribution of w_i (which can be thought of a similarity measure) as a function of kernel location. It is useful to note that other authors [12] consider g as the kernel function, in which case the notion of this being a "mean shift" is more direct and obvious. In this case, it can also be shown that the optimization is a form of local gradient ascent on a convolution surface defined by K applied to the weights w_i [12].

[1] Collins [8] has recently developed a generalization of the mean shift algorithm that does not require non-negativity.

3 An Alternative Optimization

Consider the sum of squared differences (SSD) objective function also known as the Matusita metric [6]

$$O(\mathbf{c}) = \|\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}\|^2. \tag{3.1}$$

It is well known that the Matusita metric and the Bhattacharyya coefficient are related [6, 5] by

$$O(\mathbf{c}) = 2 - 2\hat{\rho}(\mathbf{c}) \tag{3.2}$$

As a result, the minima of (3.1) coincide with the maxima of the Bhattacharyya coefficient (2.5), and hence we can equivalently work with (3.1), which we will refer to subsequently as the SSD error.

We derive a Newton-style iterative procedure to solve this optimization by expanding the expression for $\sqrt{\mathbf{p}(\mathbf{c})}$ in a Taylor series and dropping higher order terms:

$$\sqrt{\mathbf{p}(\mathbf{c}+\triangle\mathbf{c})} = \sqrt{\mathbf{p}(\mathbf{c})} + \frac{1}{2}\mathrm{d}(\mathbf{p}(\mathbf{c}))^{-\frac{1}{2}}\mathbf{U}^t\mathbf{J}_\mathbf{K}(\mathbf{c})\triangle\mathbf{c} \tag{3.3}$$

where $\mathbf{J}_\mathbf{K}$ is the n by 2 matrix of the form

$$\mathbf{J}_\mathbf{K} = \left[\frac{\partial\mathbf{K}}{\partial\mathbf{c}_1}, \frac{\partial\mathbf{K}}{\partial\mathbf{c}_2}\right] = \begin{bmatrix} \nabla_\mathbf{c}K(\mathbf{x}_1-\mathbf{c}) \\ \nabla_\mathbf{c}K(\mathbf{x}_2-\mathbf{c}) \\ \vdots \\ \nabla_\mathbf{c}K(\mathbf{x}_n-\mathbf{c}) \end{bmatrix}$$

and $\mathrm{d}(\mathbf{p})$ denotes the matrix with $\mathbf{p}$ on its diagonal.

Thus the optimization equation (3.1) can now be written in terms of a correction $\triangle\mathbf{c}$ as

$$O(\triangle\mathbf{c}) = \|\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})} - \frac{1}{2}\mathrm{d}(\mathbf{p})^{-\frac{1}{2}}\mathbf{U}^t\mathbf{J}_\mathbf{K}\triangle\mathbf{c}\|^2 \tag{3.4}$$

The minimum of this objective function is then the solution of the linear system

$$\mathbf{J}_\mathbf{K}^t\mathbf{U}\mathrm{d}(\mathbf{p})^{-1}\mathbf{U}^t\mathbf{J}_\mathbf{K}\triangle\mathbf{c} = 2\mathbf{J}_\mathbf{K}^t\mathbf{U}\mathrm{d}(\mathbf{p})^{-\frac{1}{2}}\left(\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}\right) \tag{3.5}$$

Hence, the solution to this optimization will exist provided that $\mathbf{J}_\mathbf{U} = \mathrm{d}(\mathbf{p})^{-\frac{1}{2}}\mathbf{U}^t\mathbf{J}_\mathbf{K}$ is of column rank 2.

At this point, it is interesting to compare the mean shift correction method with SSD. Comparing (3.5) and (2.8), note that the term $\mathbf{U}\mathrm{d}(\mathbf{p})^{-\frac{1}{2}}(\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})})$ corresponds to the weighting vector $\mathbf{w}$. Further, if the kernel satisfies the assumptions required by the mean shift procedure, then $\nabla_\mathbf{c}K(\mathbf{x}_i-\mathbf{c}) = g(\|\mathbf{x}_i-\mathbf{c}\|^2)(\mathbf{x}-\mathbf{c})^t$ and thus setting the right hand side of (3.5) to zero and solving for $\mathbf{c}$ *yields a modified mean shift operator.* However, in (3.5) the linear system solution attempts to jump directly to the minimum in a single step. Figure 1 shows a simple example of a return map illustrating this behavior. As with all gradient ascent algorithms, mean shift tends to under perform Newton style iteration.

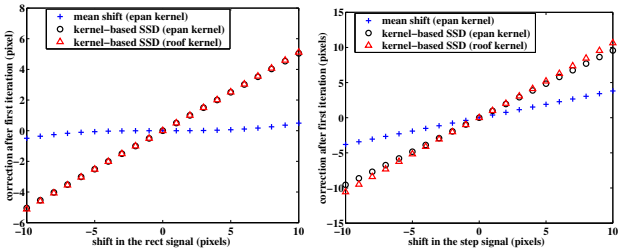

Figure 1: Return map comparison between mean shift and kernel-based SSD approach. The left plot compares the return map for shift when the target is centered on "box" signal; the right plot the performance when the target is centered on a 1D step function (an "edge"). For kernel-based SSD approach, both epan and triangular kernels have been used. It can be seen that SSD has nearly perfect 1-step performance, whereas mean shift much slower to return.

3.1 More Complex Motion Models

The basic idea of kernel-based estimation can now be easily extended to recover richer models of target motion following the line of development generally used for template tracking [4, 1, 9]. Let $f(\mathbf{x},\mu)$ represent a parametric deformation model of a target region characterized by the parameters $\mu = (\mu_1,\mu_2,\ldots,\mu_r)$. The function f is assumed to be differentiable w.r.t to both μ and $\mathbf{x}$. The definition of the kernel function can be extended to include f by defining

$$K(\mathbf{x},\mathbf{c},\mathbf{c}_\mathbf{f},\mu) = C_\mu K(f(\mathbf{x}-\mathbf{c}_\mathbf{f},\mu)-\mathbf{c}) \tag{3.6}$$

$$C_\mu = \frac{1}{\sum_i K(f(\mathbf{x}-\mathbf{c}_\mathbf{f},\mu)-\mathbf{c})} \tag{3.7}$$

Note we have been forced to reintroduce C_μ as non-rigid mappings f may change the effective area under the kernel and thus will require renormalization. The reason for differentiating between the location of the kernel and the location that the deformation acts about will become apparent when we introduce multiple kernels in the next section. For now, we will identify $\mathbf{c} = \mathbf{c}_\mathbf{f}$ and simply write $K(\mathbf{x}-\mathbf{c},\mu)$. With this, the definition of a corresponding vector form $\mathbf{K}(\mathbf{c},\mu)$ exactly parallels the development above, and thus we can now define a kernel-modulated histogram as:

$$\mathbf{q}(\mathbf{c},\mu) = \mathbf{U}^t\mathbf{K}(\mathbf{c},\mu) \tag{3.8}$$

Again, following the same steps of Taylor series expansion, we see the histogram Jacobian includes a kernel Jacobian that is now an n by $2+r$ matrix of the general form:

$$\mathbf{J}_\mathbf{K} = \left[\frac{\partial\mathbf{K}}{\partial\mathbf{c}_1}, \frac{\partial\mathbf{K}}{\partial\mathbf{c}_2}, \frac{\partial\mathbf{K}}{\partial\mu_1}, \frac{\partial\mathbf{K}}{\partial\mu_2}, \ldots, \frac{\partial\mathbf{K}}{\partial\mu_r}\right]$$

For a concrete example, consider the problem of determining the appropriate scaling of the kernel as discussed in

[8]. The three parameters of interest are the two translation parameters and one scaling parameter. It is easy to show that for kernels with $C = 1$, the kernel jacobian has the general form

$$\mathbf{J_K} = [\mathbf{K}_x, \mathbf{K}_y, \mathbf{x}.*\mathbf{K}_x + \mathbf{y}.*\mathbf{K}_y - (\overline{\mathbf{x}.*\mathbf{K}_x + \mathbf{y}.*\mathbf{K}_y})\mathbf{K}]$$

where $\mathbf{K}_x = \frac{\partial \mathbf{K}}{\partial \mathbf{c}_1}$, $\mathbf{K}_y = \frac{\partial \mathbf{K}}{\partial \mathbf{c}_2}$, $\mathbf{x}$ and $\mathbf{y}$ are vectors comprising the locations $\mathbf{x}_i - \mathbf{c}$, and $\overline{(\cdot)}$ denotes the sum of the elements of the vector argument. The subtraction of the summed derivatives is a direct consequence of the requirement that the kernel remain normalized after scaling.

Given a predicted set of model parameters $\mathbf{c}$ and μ, a correction term is computed as the solution of the linear system

$$\mathbf{e} = \sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c}, \mu)} \quad (3.9)$$

$$\mathbf{J_K^t U d(p)^{-1} U^t J_K} \begin{bmatrix} \triangle \mathbf{c} \\ \triangle \mu \end{bmatrix} = \mathbf{J_K^t U d(p)^{-\frac{1}{2}} e} \quad (3.10)$$

We defer at this point to [1] for an in-depth discussion of the general form and structure of f, and, in particular, conditions under which it is possible to optimize these computations.

3.2 The Limits of Single Kernels

At this point, it is interesting to consider some of the properties of kernel-based tracking using a single kernel. First, note that, as with the mean shift algorithm, histogram bins with zero values must be ignored or otherwise regularized [7] in order for the inverse of $\mathrm{d}(\mathbf{p})$ to be well defined. Furthermore, the rank of $\mathbf{J_U} = \mathrm{d}(\mathbf{p})^{-\frac{1}{2}}\mathbf{U}^t\mathbf{J_K}$ can in any case be no larger than $min(r+2, m)$ [10]. In fact, note that the mth value of the histogram is a function of the previous $m-1$ (due to the constraint that the histogram sums to 1), and it can be easily shown that this results in a rank reduction on $\mathbf{U}^t\mathbf{J_K}$. Note any zero values in the histogram will serve to further lower m.

From this it follows that at least three different features values are necessary to track two degrees of translational freedom, no matter how the feature values are distributed. Moreover, this form of rank deficiency *is an inherent limitation on any kernel-based objective function.* That is, this rank deficiency indicates that, at *any* point in the solution space, there are target motions that lie in the kernel of $\mathbf{J_U}$ and hence are not detectable by a change in the kernel-weighted histogram. But this in turn implies that even at an optimum, there must be always be local motions that leave the value of a histogram-based objective function unchanged. The only case where this is not true is when the histogram values themselves are at an extremum at the optimum.

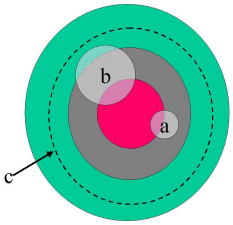

Figure 2: A tracking on a bulls-eye figure.

To briefly illustrate these ideas, consider the image shown in Figure 2. The kernel at a is ambiguous both because it only spans two colors, and because it lies along a spatial boundary. The kernel at b, although spanning three colors is still ambiguous due to the spatial boundary. The kernel at c has been centered on the bullseye and, based on the fact that the histogram is at an extremum here, does have a unique optimum, but this is an unstable equilibrium. It is interesting to note that, in general, there is no single, circularly symmetric kernel that can stably track a unique location on this target.

It is possible to have rank deficiency with an arbitrarily large number of feature values, given the appropriate interaction between the image and the kernel structure. In general, the invariance of a kernel to a given motion is analytically equivalent to examining the equation

$$0 = \mathbf{U}^t \frac{\partial \mathbf{K}}{\partial \mu_i}$$

for parameter μ_i of a given composition of kernel K and deformation f. If the chosen column vector of partial derivatives of the kernel lies in the null space of the columns of $\mathbf{U}$, then that deformation is unrecoverable and $\mathbf{J_U}$ will be rank deficient. This is the kernel-tracking equivalent of the *aperture problem.* Of course, other rank deficiencies can result due to linear combinations of local motions that lie in the (column) null space of $\mathbf{U}$.

4 Multi-Kernel Tracking

As discussed in the previous section, a single kernel, no matter what its structure, is ultimately limited by two factors: 1) dimensionality of the histogram (which in turn may be a function of available image structure), and 2) the interaction between its derivative structure and the spatial structure of the image as it is exposed by the histogram. Thus, the obvious direction to pursue is to somehow increase the dimensionality of the measurement space, and to simultaneously architect the derivative structure of the kernel to be sensitive to desired directions of motion.

The recent paper by Collins [8] is in fact a step in this direction, where two kernels (one for location and one for scale) are employed. We follow a similar approach by placing *multiple* kernels on the image at locations (and ultimately profiles) that provide independent information on motion. Consider the following kernel

Definition 4.1. A *roof kernel* of length l, span s, center $\mathbf{c}$ and normal vector $\mathbf{n}$ is defined as

$$K_r(\mathbf{x};\mathbf{c},\mathbf{n}) = \frac{4}{(l*s^2)}\max(s/2 - |(\mathbf{x}-\mathbf{c})\cdot\mathbf{n}|, 0).$$

Intuitively, a roof kernel is simply an extrusion of a triangular kernel.

The choice of the roof kernel above is motivated by the following observation. In 1-D, consider a triangular kernel centered at a location where the signal changes histogram bins (and thus forms a step edge). In 1-D, the form $\mathbf{U}^t\mathbf{J}_\mathbf{K}(\mathbf{x})$ computes the scalar convolution of g (which is now itself a step edge) with the underlying binned image at $\mathbf{x}$. It follows that the triangular kernel, in the absence of noise, is the optimal detector for the step edge in the binned signal [2]. Furthermore, the response is a linear function of translation within the span of the kernel. Although this discussion is for the one-dimensional case, it is easy to see that a similar argument can be made in for a two-dimensional roof kernel with orientation $\mathbf{n}$ where $\mathbf{n}$ is points along the local "spatial gradient" of the binned image[2].

Intuitively, two roof kernels oriented in orthogonal directions provide independent information on x and y motion. Algebraically, the use of two kernels doubles the dimension of the histogram, and therefore increases the potential rank of the system. To formalize and generalize these ideas, we proceed as follows:

Suppose we adjoin several kernels into an $n \times r$ kernel matrix $\mathbf{K}(\mathbf{c_f},\mu) = [\mathbf{K}_1(\mathbf{c}_1,\mathbf{c_f},\mu), \mathbf{K}_2(\mathbf{c}_2,\mathbf{c_f},\mu), \ldots, \mathbf{K}_r(\mathbf{c}_r,\mathbf{c_f},\mu)]$. Following the development above, we can now define matrices that contain the corresponding model and target histograms as $\mathbf{Q} = \mathbf{U}(t_0)^t\mathbf{K}(0,0)$ and $\mathbf{P}(\mathbf{c}_f,\mu,t) = \mathbf{U}(t)^t\mathbf{K}(\mathbf{c}_f,\mu)$. Note that we now distinguish between kernel location ($\mathbf{c}$) and the center of the deformation ($\mathbf{c_f}$); the latter is now taken to be the origin of the entire collection of kernels. The optimization is then

$$O(\mathbf{c}_f,\mu) = \|\mathbf{Q} - \mathbf{P}(\mathbf{c}_f\mu,t)\|^2 \tag{4.1}$$

where $\|\cdot\|$ now denotes the Frobenius norm.

At this point, it is useful to recall that the Frobenius norm on a matrix is equivalent to the norm of the single column vector constructed by concatenating the columns of the matrix. Let $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ denote the "stacked" versions of $\mathbf{P}$ and $\mathbf{Q}$. By recasting the problem in this form, and performing the same derivation of the Newton iterations, we have:

$$\mathbf{e} = \sqrt{\bar{\mathbf{p}}(\mu,t)} - \sqrt{\bar{\mathbf{q}}} \tag{4.2}$$

$$\mathbf{J}_\mathbf{U}(\mu) = \mathrm{d}(\bar{\mathbf{p}})^{-\frac{1}{2}} \begin{bmatrix} \mathbf{U}^t\mathbf{J}_\mathbf{K}(\mathbf{c}_1,\mathbf{c_f},\mu) \\ \vdots \\ \mathbf{U}^t\mathbf{J}_\mathbf{K}(\mathbf{c}_r,\mathbf{c_f},\mu) \end{bmatrix} \tag{4.3}$$

$$\mathbf{J}_\mathbf{U}^t\mathbf{J}_\mathbf{U} \begin{bmatrix} \triangle\mathbf{c}_f \\ \triangle\mu \end{bmatrix} = \mathbf{J}_\mathbf{U}^t\mathbf{e} \tag{4.4}$$

It should be clear that the algebraic structure of the stacked system will be no worse than that of any single kernel, and in general will be much better. Of course, multiple kernels is not a panacea for improving tracking quality. For example, applying the same kernel at the same location does not improve the rank structure of the system. Similarly, kernels placed or oriented appropriately may not yield independent information. Thus, care and analysis of kernel properties is essential in constructing multi-kernel trackers.

4.1 Multiple Kernel Constructions

The previous section has already motivated one possible multi-kernel construction: the combination of two ramp kernels in orthogonal directions to provide location information. In particular, returning to the simple bulls-eye example of Figure 2, it is easy to see that the two kernels now provide two independent measures of motion, and thus it is possible to track many locations on the target.

Once we take the step of allowing multiple kernels, there are a number additional possibilities for defining tracking structures. Here we detail a few.

Symmetry Seeking: As currently defined, the goal of kernel-based tracking is to match a fixed kernel-weighted histogram to a time and parameter varying one. An alternative is to match two time and location varying histograms to each other. This is particularly useful in cases where the image itself exhibits some type of symmetry, in which case the trackers will be forced into a symmetric configuration. For example, two kernels placed on the opposing sides of the bulls-eye figure in Figure 2 will automatically array themselves symmetrically about the center.

Fixed Reference: Rather than selecting a sample histogram at a location, an alternative is to choose a fixed reference or some or all values of the kernel-weighted histogram. For example, a uniformly colored foreground object might be forced to have a histogram which is exactly 50% the foreground color (but leaving the remaining distribution unspec-

[2] It is important to note that these are changes in the *bin index* which reflect the underlying spatial gradient only to the extent the binning function is uniform and monotonic.

ified). In this case the kernel would naturally center itself on the occluding contour.

Translation and Rotation: Kernels that are responsive to rotation must have changing value along circular lines about the origin. This is true of the roof kernel for a small range of motion. However, it is not hard to design kernels that emphasize this attribute. For example, the kernel that is defined in polar coordinates as $K(\theta, r) = \sin^2(\theta), r < r_{max}$ with suitable normalization is such a kernel.

5 Demonstrations

5.1 Tracking Location

In this section, we compare the tracking results of the kernel-based SSD tracking with the mean shift tracker for translation motion. The kernel-based SSD approach is implemented both using single kernels and multiple kernels. Thus, we compare results from three implementations, namely mean shift, single kernel-based SSD and the multiple kernel-based SSD. The kernel used in the single kernel-based and the mean shift approach is the Epanechnikov kernel used by [3] for mean shift tracking. For the multi-kernel approach, we use the triangular kernel discussed in Section 4.

The histogram binning for comparison is based on the HSV space. The bins in the model distribution containing less than 10% of full value were ignored in the calculations to enhance stability. The bins in the multi-kernel approach consists of a bin for dark pixels ($< 20\%$ of full scale brightness), a bin for unsaturated pixels ($< 20\%$ full saturation) and ten equally spaced bins for hue values. All model distributions are computed once on the initializing frame and held fixed for the entire sequence.

As indicated by the return map figure in Section 3.2, the convergence for mean shift is much slower compared to the kernel-based approach, so we compare the tracking results for all the three implementations with fixed number of iterations per frame. The Figure 3, Figure 4 and Figure 5 show the tracking results when only 2 iterations per frame were run and every third frame in the sequence was used. It is quite evident from Figure 3 that the mean shift tracker performs quite poorly with HSV based histogram binning. On the other hand, the kernel-based SSD trackers perform quite well. If the number of iterations are increased and fewer frames are dropped, the mean shift tracker gives better results. We found that the mean shift tracker performs better with histogram binning based on RGB space. Figure 4 shows the comparison of the mean shift and kernel-based SSD tracking with histogram binning based on both HSV and RGB space. It can be seen that although the mean shift tracker performs better, the kernel-based SSD trackers still outperform the mean shift tracker. Figure 5 shows the selected template region in the first frame and five frames from the 450 frame tracked sequence. The kernel-based SSD trackers do a better job of tracking than the mean shift trackers with fixed number of iterations. The fast convergence of kernel-based SSD tracking and slow convergence of the mean shift tracking is thus quite evident.

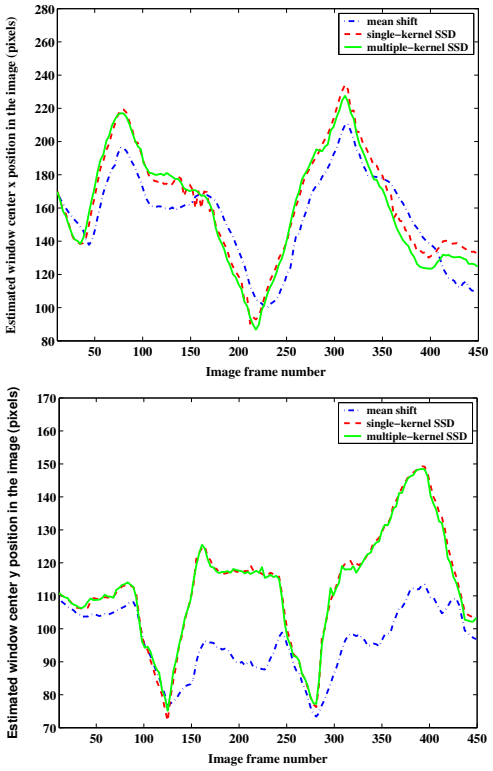

Figure 3: The center position of the tracked region for mean shift, single-kernel SSD and the multiple-kernel SSD with histogram binning based on HSV space (2 iterations per frame).

5.2 Tracking Similarities

As discussed earlier in the Section 4, the power of the multiple-kernel SSD tracking lies in tracking similarities. Different kernels for each similarity namely x translation, y translation, scale etc can be used to compute each of these similarities. In Figure 6, we show results for tracking an image sequence with primarily translation and scale. Two orthogonal triangular kernels are used for x and y translation and a limited conical kernel is used for scaling. Six frames from a 90 frame sequence are shown. It can be easily seen the scaling changes by more than a factor of 2.

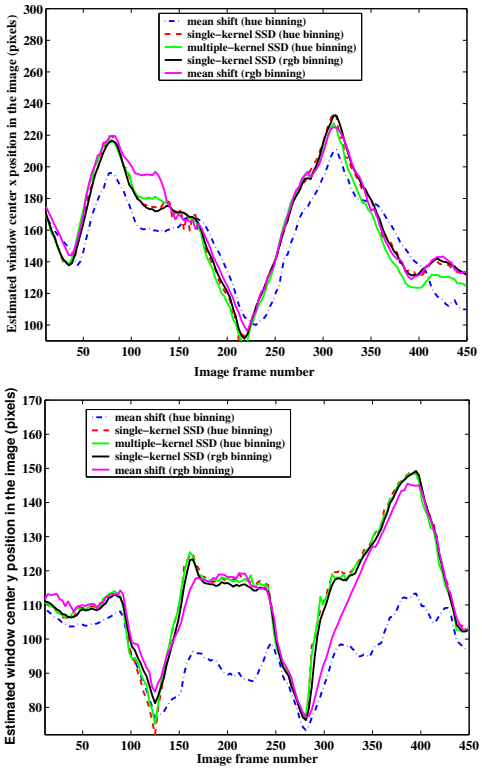

Figure 4: The center position of the tracked region for mean shift, single-kernel SSD and the multiple-kernel SSD with histogram binning based on both HSV and RGB space (2 iterations per frame).

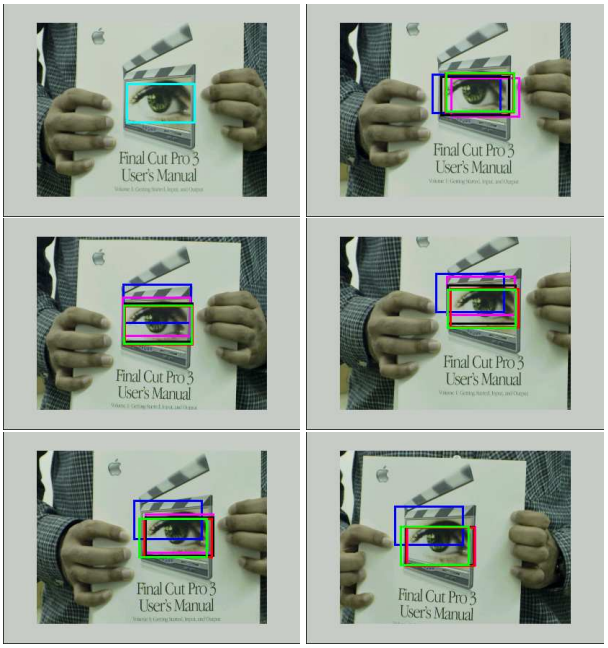

Figure 5: Tracking comparison for mean shift, single-kernel SSD and multiple-kernel SSD approaches with 2 iterations per frame. The first frame shows the selected template region. In the following frames, the tracked region from these different tracking approaches are shown. The tracked region in each frame for mean shift, single-kernel SSD and multiple-kernel SSD with histogram binning based on HSV space are shown in blue, red and green rectangle respectively. The tracked region with histogram binning based on RGB space for mean shift and single-kernel SSD are shown in magenta and black respectively.

5.3 A Structured Tracking Example

Figure 7 shows four frames from a 500 frame stereo sequence showing a light yellow wand moving above a highly textured background.

A tracker for this wand was designed using the following elements. The histogram space for this problem is identical to the HSV binning described previously. Two symmetrically arrayed roof trackers were keyed to the foreground color of the wand. The results of these trackers are shown by the red and blue (upper and middle) plusses on the wand. One roof kernel orthogonal to the axis of symmetry was keyed to the foreground color of the wand, with the percentage of foreground set to be 50%. For small changes in angle, the roof kernel trackers on a straight edge are rotation invariant, so location is first tracked using the five roof kernels, and then the orientation of the wand is computed after the tracking cycle by fitting a line though the centerline between the symmetry pairs. This is the blue (or central) line on the tool). Although in principle the estimation of rotation can be direction combined with translation as detailed above, the local invariance of the translation stage to rotation facilitates this simpler solution. Note the entire solution for location and rotation is also scale invariant.

This sequence is the output of one camera of a stereo pair. A similar tracking algorithm for a second pair is used to track the wand, and the resulting locations and kernel orientations are used to construct the five degree of freedom pose of the wand in space.

6 Conclusions

We believe the use of kernel-weighted histograms as described in this paper provide a number of unifying insights into the general form and structure of visual tracking problems. Historically, two "extremes" in the spectrum of tracking algorithms are blob tracking and template tracking. The former is unstructured, approximate, is insensitive to the details of target geometry, and is typically solved using either brute force search, or statistical techniques (e.g. centroids)

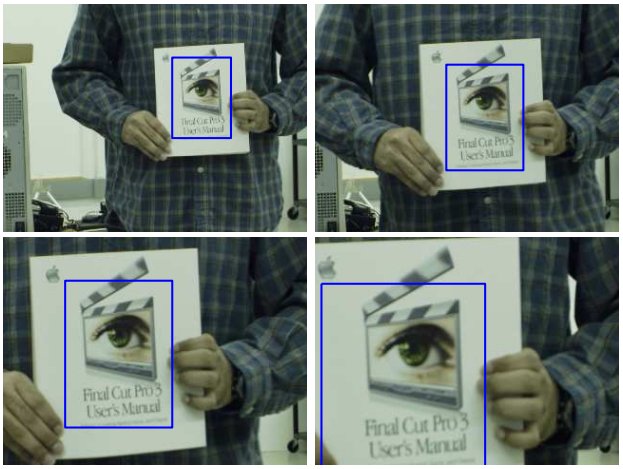

Figure 6: Similarity tracking with the multi kernel SSD approach. The first frame shows the selected template region. The subsequent frames show the tracked region with a blue rectangle

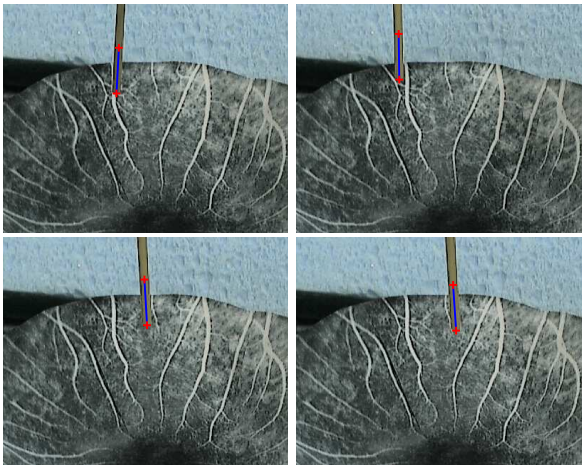

Figure 7: Tracking a colored wand against a complex background. This is the left image of a stereo pair.

on segmented images. Template tracking, on the other hand, emphasizes the spatial structure of the target, can recover a rich class of motions, and is typically solved using continuous optimization.

Kernel-based methods forge a link between these two extremes. Not only do similar optimization techniques apply, but there is a fundamental relationship between the structure of a collection of histograms for detecting and estimating target motion, and the spatial structure of the target itself. At one extreme, a simple circularly symmetric kernel places few constraints on target motion, but in the limit, as more and more kernels are added, the spatial structure of the target becomes more and more constrained.

Our current work is continuing to develop this connection in a more formal sense, and to apply these insights to challenging tracking problems. For example, it is not yet clear how to properly adapt the histogram structure over time to adapt to changing illumination, changing target appearance, or occlusion. Likewise, there is, as of yet, no well-developed theory of how to best make use of multiple features simultaneously. Finally, in addition to tracking, the same SSD-based approach can be applied to other registration problems. By doing so, we hope to gain the robustness associated with integral measures such as histograms, while making use of well-developed optimization results in this area.

Acknowledgements The authors would like to thank Xiangtian Dai for the helpful discussions. This material is based upon work supported by the National Science Foundation under Grant Nos. IIS-0099770 and ERC-9731478.

References

[1] S. Baker and I. Matthews. Equivalence and Efficiency of Image Alignment Algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Patt. Recognition*, 2001.

[2] John Canny. A Computational Approach to Edge Detection. *IEEE Trans. Patt. Anal. Mach. Intelligence*, pages 679–98, November 1986.

[3] Comaniciu D., Ramesh V., and Meer P. Kernel-Based Object Tracking. *PAMI*, 25(5):564–575, 2003.

[4] G. D. Hager and P. N. Belhumeur. Efficient Region Tracking of with Parametric Models of Illumination and Geometry. *IEEE Trans. Patt. Anal. Mach. Intelligence*, 20(10):1025–1039, October 1998.

[5] T. Kailath. The Divergence and Bhattacharyya Distance Measures in Signal Selection. *IEEE Transactions on Communication Technology*, 15(1):52–60, 1967.

[6] K. Matusita. Decision Rules Based on Distance for Problems of Fit, Two Samples and Estimation. *Annals of Mathematical Statistics*, 26(4):631–640, 1955.

[7] A. Orlitsky, N. P. Santhanam, and J. Zhang. Always Good Turing: Asymptotically Optimal Probability Estimation. *Science*, 302:427–431, 2003.

[8] Collins R. Mean-shift Blob Tracking through Scale Space. *CVPR*, 2:234–240, 2003.

[9] J. Shi and C. Tomasi. Good Features to Track. In *Proc. IEEE Conf. Comp. Vision and Patt. Recog.*, pages 593–600. IEEE Computer Society Press, 1994.

[10] G. Strang. *Linear Algebra and its Applications*. Academic Press, Inc., 1980. Second edition.

[11] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman and Hall, 1995. First edition.

[12] Cheng Y. Mean Shift, Mode Seeking, and Clustering. *PAMI*, 17(8):790–799, 1995.