Efficient Optimal Kernel Placement for Reliable Visual Tracking

Zhimin Fan, Ming Yang, Ying Wu, Gang Hua, Ting Yu
Department of Electrical & Computer Engineering, Northwestern University
{zfa825,mya671,yingwu,ganghua,tingyu}@ece.northwestern.edu

Abstract

This paper describes a novel approach to optimal kernel placement in kernel-based tracking. If kernels are placed at arbitrary places, kernel-based methods are likely to be trapped in ill-conditioned locations, which prevents the reliable recovery of the motion parameters and jeopardizes the tracking performance. The theoretical analysis presented in this paper indicates that the optimal kernel placement can be evaluated based on a closed-form criterion, and achieved efficiently by a novel gradient-based algorithm. Based on that, new methods for temporal-stable multiple kernel placement and scale-invariant kernel placement are proposed. These new theoretical results and new algorithms greatly advance the study of kernel-based tracking in both theory and practice. Extensive real-time experimental results demonstrate the improved tracking reliability.

1. Introduction

Representing the target by the convolution of its features with a spatially weighted kernel, kernel-based tracking methods, such as mean shift [4, 5] or Newton-style methods [10], are in general computationally efficient due to the gradient-based optimization. This is very attractive for real-time applications, compared with other tracking schemes, such as particle filters [13] or exhaustive template matching.

As a differential approach, the performance of kernel-based methods are greatly influenced by the quality of the searching directions calculated based on the local measurements. A core issue among many improvements on kernel-based methods is to enhance the kernel's ability of acquiring a broad spectrum of measurements, which should be informative enough to determine the best search direction towards the desired mode in the feature space, e.g., the matched target in tracking problems. There are two general attempts towards this goal. One is to tailor the kernel design based on the properties of local measurements, such as the design of kernels with a variable bandwidth [6] or with anisotropic shapes [19]. Another approach is to use multiple kernels [8, 10], because there exists ill-conditioned cases for single kernels, such that the computed searching direction of single kernel is indifferent to certain motion and thus preventing the unique recovery of these motion parameters. As revealed in [10], the culprit of this phenomenon is the rank deficiency in a least square estimation. An in-depth analysis of these "ill-conditioned" cases is given in [8] based on the observability of kernels, leading to the approach of multiple collaborative kernels that guarantees the enhancement of kernel observability and produces more reliable motion estimation.

All these methods assume that unique and stable motion estimation can be obtained as in the well-conditioned cases. In other words, a small perturbation of the placement of the kernel does not change much the motion estimation. Unfortunately, evidence from the practice challenges this assumption. For example, in mean shift tracking, it is often observed that different initializations of the tracker (i.e., delineate the region to track and place the kernel accordingly) may largely influence the performance. If we put the same kernel at one place, the tracker may work well; but when choosing a slightly different place, the tracker may fail unexpectedly. Thus, this raises an interesting and critical question: **is there an optimal placement for the kernels to achieve reliable tracking?** Specifically:

- How can we evaluate the sensitivity of a placement?
- Does there exist a computationally efficient way to find the optimal kernel placement?
- How can we place multiple kernels if a training sequence is available?
- Does there exist a scale-invariant kernel placement?

This paper presents our study in search of the answers to the above intriguing questions in order to achieve more reliable tracking results. Our study starts with a conjecture that subregions of the target may play different roles in tracking, since some subregions of the target may be more reliable for tracking while others may not. We provide a detailed analysis in order to identify those regions, and derive a closed-form criterion for evaluating the sensitivity of kernel placement. To make the optimal kernel placement feasible, we

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

derive a *gradient-based* algorithm to efficiently search for an optimal placement, which greatly reduces the computational cost compared with a brute force way of examining all the possible placement on the image exhaustively. We also propose a method to discover temporal-stable kernels for multiple kernel placement, and study the issue of scale-invariant kernel placement.

2. Kernel-based Tracking

In kernel-based tracking [5, 10], object is represented by a color histogram, $\mathbf{q}=[q_1, q_2, \ldots, q_m]^T \in \mathbb{R}^m$,

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

where $\{\mathbf{x}_i\}_{i=1\ldots n}$ are the pixel locations in the image, $b(\mathbf{x}_i)$ is a binning function that maps the color of $\mathbf{x}_i$ onto a histogram bin u, with $u \in \{1 \ldots m\}$. K is a spatially weighted kernel centered at $\mathbf{c}$ and δ is the Kronecker delta function. A more concise matrix form is written as [10]:

$$\mathbf{q}(\mathbf{c}) = \mathbf{U}^T\mathbf{K}(\mathbf{c}), \tag{2}$$

where

$$\mathbf{U} = \begin{bmatrix} \delta(b(\mathbf{x}_1), u_1) & \ldots & \delta(b(\mathbf{x}_1), u_m) \\ \vdots & \vdots & \vdots \\ \delta(b(\mathbf{x}_n), u_1) & \ldots & \delta(b(\mathbf{x}_n), u_m) \end{bmatrix} \in \mathbb{R}^{n\times m},$$

$$\mathbf{K} = \frac{1}{C}\begin{bmatrix} K(\mathbf{x}_1 - \mathbf{c}) \\ \vdots \\ K(\mathbf{x}_n - \mathbf{c}) \end{bmatrix} \in \mathbb{R}^n.$$

The tracking process is to find the best displacement $\Delta\mathbf{c}$ such that the histogram $\mathbf{p}(\mathbf{c}+\Delta\mathbf{c})$ at the new location best matches the target histogram $\mathbf{q}$, i.e., $\Delta\mathbf{c}^* = \arg\min_{\Delta\mathbf{c}} O(\mathbf{q}, \mathbf{p}(\mathbf{c}+\Delta\mathbf{c}))$, where $O(\cdot,\cdot)$ is the objective function for matching, such as the Bhattacharyya coefficient [5] or the equivalent Matusita metric [10],

$$O(\Delta\mathbf{c}) \triangleq \|\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c}+\Delta\mathbf{c})}\|^2. \tag{3}$$

The ill-conditioned case is discovered when linearizing Eq.(3) w.r.t. $\Delta\mathbf{c}$, we have

$$\mathbf{M}\Delta\mathbf{c} = \sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}, \tag{4}$$

where $\sqrt{\mathbf{q}}, \sqrt{\mathbf{p}(\mathbf{c})} \in \mathbb{R}^m$, $\Delta\mathbf{c} \in \mathbb{R}^r$, $\mathbf{M} \in \mathbb{R}^{m\times r}$,

$$\mathbf{M} = \tfrac{1}{2}\mathtt{diag}(\mathbf{p}(\mathbf{c}))^{-\frac{1}{2}}\mathbf{U}^T\mathbf{J}_\mathbf{K}(\mathbf{c}),$$
$$\mathbf{J}_\mathbf{K}(\mathbf{c}) = \begin{bmatrix} \nabla_c K(\mathbf{x}_1 - \mathbf{c}) \\ \nabla_c K(\mathbf{x}_2 - \mathbf{c}) \\ \vdots \\ \nabla_c K(\mathbf{x}_n - \mathbf{c}) \end{bmatrix}, \tag{5}$$

and $\mathtt{diag}(\mathbf{p})$ represents the matrix with $\mathbf{p}$ on its diagonal. In order to get a stable solution for $\Delta\mathbf{c}$, it is required that $\mathbf{M}$ to be of full rank. Otherwise, $\Delta\mathbf{c}$ cannot be uniquely determined, indicating an ill-conditioned situation.

3. Optimal Single Kernel Placement

The ill-conditioned case greatly deteriorates the tracking performance. In this section, we give more detailed analysis into such a case and propose a criterion to select optimal locations to place kernels, which avoids the ill-conditioning to the largest extent.

3.1. Applying the condition theory

To solve $\mathbf{x}$ from a linear equation

$$\mathbf{A}\mathbf{x} = \mathbf{b}.$$

Besides requiring $\mathbf{A}$ to be invertible, it is also expected that the solution is numerically stable. The analysis of how sensitive the $\mathbf{x}$ is, given changes in $\mathbf{b}$, can be achieved by examining the condition number defined as,

$$\kappa(\mathbf{A}) = \|A\|\|A^{-1}\|.$$

For example, when $2-$norm is used, $\kappa_2(\mathbf{A}) = \|\mathbf{A}\|_2\|\mathbf{A}^{-1}\|_2 = \sigma_1(\mathbf{A})/\sigma_n(\mathbf{A})$, which is the ratio between the largest and the smallest singular value.

For a single kernel, we need to calculate the motion parameter $\Delta\mathbf{c}$ from $\mathbf{M}\Delta\mathbf{c} = \sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}$. The solution is

$$\Delta\mathbf{c} = (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T(\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}).$$

So, $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$ should be considered as a whole entity, which tells how sensitive the $\Delta\mathbf{c}$ is, given small changes in $\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}$.

Since $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$ is not a square matrix, its "condition number" is not well defined. However, considering the essence of this problem, if we take SVD of the $2\times m$ matrix $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$ as $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T = \mathbf{U}\Sigma\mathbf{V}^T$.

We would expect that the 2 singular values in Σ be comparable to each other, such that the $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$ is equally sensible in both directions of its two orthonormal singular vectors. Otherwise, if the two singular values are unbalanced, a fluctuation in $\sqrt{\mathbf{q}} - \sqrt{\mathbf{p}(\mathbf{c})}$ caused by noise will change the solution $\Delta\mathbf{c}$ significantly along the singular vector corresponding to the larger singular value, and negligibly along the singular vector corresponding to the smaller singular value, bringing in undesirable numerical instability, and such a region is generally considered to be a bad placement of the kernel.

Notice that

$$(\mathbf{M}^T\mathbf{M})^{-1} = (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T((\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T)^T = \mathbf{U}\Sigma^2\mathbf{U}^T,$$

and assume σ_1 and σ_2 are two singular values of $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$, it is easy to verify that

$$\kappa_2((\mathbf{M}^T\mathbf{M})^{-1}) = (\sigma_1/\sigma_2)^2.$$

We also have $\kappa_2(\mathbf{M}^T\mathbf{M}) = \kappa_2((\mathbf{M}^T\mathbf{M})^{-1})$. In view of this, the sensitivity evaluation of $(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$ is just equivalent to inspecting the condition number of $(\mathbf{M}^T\mathbf{M})$,

since $\kappa_2(\mathbf{M}^T\mathbf{M})$ monotonically increases/decreases when σ_1/σ_2 increases/decreases.

So, the ***criterion*** for a reliable kernel tracking is: we need to put the kernel to such a place that the condition number of $\mathbf{M}^T\mathbf{M}$ is minimized.

$$\min_{\mathbf{c}} \kappa_2(\mathbf{M}^T\mathbf{M}). \tag{6}$$

3.2. Interpretation of the condition number criterion using the 2-norm

Here, we give an intuitive interpretation of the condition number criterion using the 2-norm, which requires to evaluate the singular values of $\mathbf{M}^T\mathbf{M}$. Actually, evaluating the singular values of $\mathbf{M}$ doesn't affect the analytical result.

In the following, $\mathbf{x}_i$ represents a data point with index i, while $\mathbf{x}_i^j$ denotes point i of color j. For the problem of n points within the kernel range and m color bins. By recalling Eq.(5),

$$\mathbf{M} = \frac{1}{2}\mathtt{diag}(\mathbf{p}(\mathbf{c}))^{-\frac{1}{2}}\mathbf{U}^T\mathbf{J_K}(\mathbf{c}).$$

The i^{th} row of the $n \times 2$ matrix $\mathbf{J_K}$ is $(\mathbf{x}_i - \mathbf{c})g\left(\|\frac{\mathbf{x}_i-\mathbf{c}}{h}\|^2\right)$, with $g(\cdot) = -k'(\cdot)$ and $k'(\cdot)$ being the profile of the kernel $\mathbf{K}$. Then, by left multiplying the $m \times n$ sifting matrix $\mathbf{U}^T$, the resulting $m \times 2$ matrix, denoted as $\mathbf{D} = \mathbf{U}^T\mathbf{J_K}(\mathbf{c})$, has the meaning that the j^{th} row of $\mathbf{D}$ is the sum of $\mathbf{x}_i^j - \mathbf{c}$ weighted by $g\left(\|\frac{\mathbf{x}_i^j-\mathbf{c}}{h}\|^2\right)$ for all pixels $\mathbf{x}_i$ of color j, $i = 1, \ldots, n, j = 1, \ldots, m$.

As for $\mathbf{M} = \frac{1}{2}\mathtt{diag}(\mathbf{p}(\mathbf{c}))^{-\frac{1}{2}}\mathbf{D}$, we can see that each row of $\mathbf{M}$ is just the normalization of the corresponding row in $\mathbf{D}$ by a factor of $2\mathbf{p}(\mathbf{c})^{\frac{1}{2}}$, thus giving a particular constraint on $\Delta\mathbf{c}$,

$$\left[\frac{1}{2\sqrt{\mathbf{p}_j}}\sum_i(\mathbf{x}_i^j - \mathbf{c})g\left(\left\|\frac{\mathbf{x}_i^j-\mathbf{c}}{h}\right\|^2\right)\right]\Delta\mathbf{c} = \sqrt{\mathbf{q}_j} - \sqrt{\mathbf{p}_j}. \tag{7}$$

The intuition of the LHS of this equation, i.e., the j^{th} row of $\mathbf{M}$, is that we sum all the displacement vector $\mathbf{x}_i^j - \mathbf{c}$ of color j, which are weighted by $g\left(\|\frac{\mathbf{x}_i^j-\mathbf{c}}{h}\|^2\right)$, and then the summation is scaled by $\frac{1}{2\sqrt{\mathbf{p}_j}}$. Denote this result as $[d_x^j \quad d_y^j]$, which can be effectively considered as the *center of mass* of all pixels of color j.

Since a good kernel placement is featured by a $\mathbf{M}$ with comparable singular values, this requires that all the rows of $\mathbf{M}$, $[d_x^j \quad d_y^j], j = 1 \ldots, m$ well span the 2D space. The corresponding situation is that all the center of masses of the color components should be distributed evenly around the center $\mathbf{c}$.

3.3. An equivalent condition number

In practice, 2-norm condition number is not straightforward to compute. In this section, we introduce another form of condition number, being equivalent to the 2-norm condition number when the matrix is 2×2 symmetric positive definite. The new condition number offers a great ease of computation and facilitates an efficient searching algorithm for optimal kernel placement, as will be derived as follows.

The Schatten 1-norm [2][11][17] is defined as,

$$\|\mathbf{A}\|_S = \sum \sigma_i,$$

where $\sigma_1, \ldots, \sigma_n$ are the singular values of $\mathbf{A}$. When $\mathbf{A}$ is a symmetric positive definite matrix, we can have,

$$\begin{array}{rcl} \|\mathbf{A}\|_S & = & \sum \sigma_i = \text{trace}(\mathbf{A}), \\ & & \prod \sigma_i = \det(\mathbf{A}). \end{array} \tag{8}$$

Given a 2×2 symmetric positive definite matrix $\mathbf{A}=\{a_{ij}\}$, we can then have a *closed form* expression of S-norm condition number as,

$$\begin{array}{rcl} \kappa_S(\mathbf{A}) & = & \|\mathbf{A}\|_S\|\mathbf{A}^{-1}\|_S = \text{trace}(\mathbf{A})\text{trace}(\mathbf{A}^{-1}) \\ & = & \text{trace}(\mathbf{A})\frac{\text{trace}(\mathbf{A})}{\det(\mathbf{A})} = \frac{(a_{11}+a_{22})^2}{a_{11}a_{22}-a_{12}a_{21}}. \end{array} \tag{9}$$

And equivalently,

$$\kappa_S(\mathbf{A}) = (\sum \sigma_i)^2 / \prod \sigma_i. \tag{10}$$

According to [9], any two condition numbers $\kappa_\alpha(\mathbf{A})$ and $\kappa_\beta(\mathbf{A})$ are equivalent in that constants c_1 and c_2 can be found for which

$$c_1\kappa_\alpha(\mathbf{A}) \le \kappa_\beta(\mathbf{A}) \le c_2\kappa_\alpha(\mathbf{A}).$$

For example, $\frac{1}{n}\kappa_2(\mathbf{A}) \le \kappa_1(\mathbf{A}) \le n\kappa_2(\mathbf{A})$, for $\mathbf{A} \in \mathcal{R}^{n\times n}$. Here, we can show 2 Propositions, the proof of which [1] are omitted due to page limit.

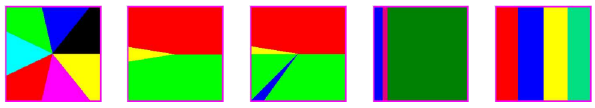
Proposition 1 *$\kappa_2(\mathbf{A}) \le \kappa_S(\mathbf{A}) \le n^2\kappa_2(\mathbf{A})$, if matrix $\mathbf{A}$ is $n \times n$ symmetric positive definite.*

Proposition 2 *κ_S is monotonically* increasing/decreaing *when κ_2* increases/decreases*, if the matrix is 2×2 symmetric positive definite.*

Therefore, since $\mathbf{M}^T\mathbf{M}$ is 2×2 symmetric positive definite, if we find the local/global minimum of its κ_S, we in fact find the local/global minimum of its κ_2. This nice property ensures that the good kernel measured by κ_S is just the good kernel measured by κ_2. Because computing κ_S only involves element-wise calculation, it is much more convenient and efficient than κ_2, which is non-analytical.

So the claim is that, κ_S is an equivalent substitute for κ_2 in all the cases [3][15][16][18] when the 2-norm condition number of a 2×2 covariance matrix needed to be evaluated.

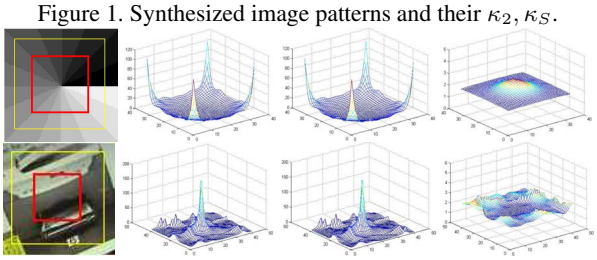
To summarize, given



$\kappa_2 = 1.0$, $\kappa_2 = 8.3$, $\kappa_2 = 11.1$, $\kappa_2 = 10301$, $\kappa_2 = \infty$

$\kappa_S = 4.0$, $\kappa_S = 10.4$, $\kappa_S = 13.2$, $\kappa_S = 10303$, $\kappa_S = \infty$

Figure 1. Synthesized image patterns and their κ_2, κ_S.



(a) (b) (c) (d)

Figure 2. column (a): yellow: image region, red: kernel. column (b), (c), and (d) are κ_2, κ_S and $\kappa_S - \kappa_2$ evaluated in the region of column (a), respectively.

$$\mathbf{M} = \begin{bmatrix} d_x^1 & d_y^1 \\ \vdots & \vdots \\ d_x^m & d_y^m \end{bmatrix},$$

where

$$[d_x^j \quad d_y^j] = \left[\frac{1}{2\sqrt{\mathbf{P}_j}} \sum_{i,b(\mathbf{x}_i)=j} (\mathbf{x}_i^j - \mathbf{c}) g\left(\left\| \frac{\mathbf{x}_i^j - \mathbf{c}}{h} \right\|^2 \right) \right], \quad (11)$$

is the weighted sum of the displacement vectors of all pixels of color j, or called the center of mass of color component j.

We can compute in closed-form

$$\begin{aligned} \kappa_S(\mathbf{M}^T\mathbf{M}) &= \|(\mathbf{M}^T\mathbf{M})\|_S \|(\mathbf{M}^T\mathbf{M})^{-1}\|_S \\ &= \frac{(\sum (d_x^j)^2 + \sum (d_y^j)^2)^2}{\sum (d_x^j)^2 \sum (d_y^j)^2 - (\sum (d_x^j d_y^j))^2}. \end{aligned} \quad (12)$$

The region with a small $\kappa_S(\mathbf{M}^T\mathbf{M})$ is the good choice for kernel placement, on which the stability of tracking will be better than those regions with larger condition numbers. Fig. 1 shows some synthesized image patterns and their κ_2, κ_S. The first one has the lowest condition number, coinciding the interpretation given in Sec 3.2. Fig. 2 shows the κ_2, κ_S and their differences evaluated in two image regions. It is observed that they exhibit the same pattern of ridges and valleys, and their differences are small compared with their own magnitudes.

3.4. Find optimal kernel placement efficiently

In practice, only obtaining the criterion for optimal kernel placement is insufficient, since it is not attractive to exhaustively evaluate this criterion all over the image. In this section, we derive a gradient descent algorithm, which can efficiently find good placement for kernels.

Notice that in Eq.(11) and Eq.(12), the condition number κ_S only involves d_x^j and d_y^j, which are explicitly presented as a function of the kernel position $\mathbf{c}$. So, we can compute the *derivative* of $\kappa_S(\mathbf{M}^T\mathbf{M})$ w.r.t the kernel position, $\mathbf{c}$. Denote

$$\kappa_S(\mathbf{M}^T\mathbf{M}) = \frac{A^2}{B} = \frac{A^2}{DE - F^2},$$

where

$$\begin{aligned} &A = \textstyle\sum_j \|[d_x^j \quad d_y^j]\|^2 = \sum_j (d_x^j)^2 + \sum_j (d_y^j)^2, \\ &D = \textstyle\sum_j (d_x^j)^2, \quad E = \sum_j (d_y^j)^2, \quad F = \sum_j (d_x^j d_y^j), \\ &B = \textstyle\sum_j (d_x^j)^2 \sum_j (d_y^j)^2 - \left(\sum_j (d_x^j d_y^j) \right)^2 = DE - F^2. \end{aligned}$$

Here, the goal is to compute

$$\frac{\partial \frac{A^2}{B}}{\partial \mathbf{c}} = \frac{2AB\frac{\partial A}{\partial \mathbf{c}} - A^2\frac{\partial B}{\partial \mathbf{c}}}{B^2} = \frac{2AB\frac{\partial A}{\partial \mathbf{c}} - A^2(D\frac{\partial E}{\partial \mathbf{c}} + E\frac{\partial D}{\partial \mathbf{c}} - 2F\frac{\partial F}{\partial \mathbf{c}})}{B^2}.$$

So, we need $\frac{\partial A}{\partial \mathbf{c}}$, $\frac{\partial D}{\partial \mathbf{c}}$, $\frac{\partial E}{\partial \mathbf{c}}$, and $\frac{\partial F}{\partial \mathbf{c}}$. In practice, we will express $\frac{\partial D}{\partial \mathbf{c}} = [\frac{\partial D}{\partial c_x} \quad \frac{\partial D}{\partial c_y}]$, $\frac{\partial E}{\partial \mathbf{c}} = [\frac{\partial E}{\partial c_x} \quad \frac{\partial E}{\partial c_y}]$, and $\frac{\partial F}{\partial \mathbf{c}} = [\frac{\partial F}{\partial c_x} \quad \frac{\partial F}{\partial c_y}]$. Details are derived as follows,

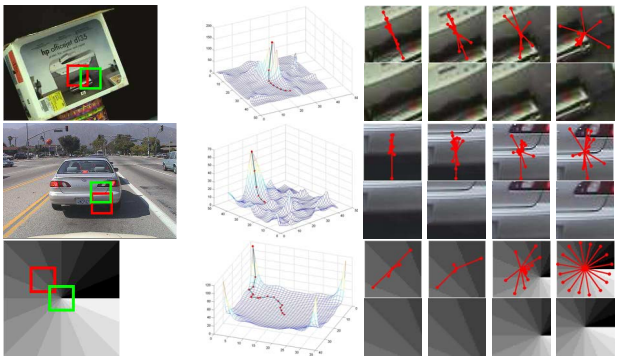
$$\begin{aligned} &\frac{\partial A}{\partial \mathbf{c}} = \sum_j 2[d_x^j \quad d_y^j] \frac{\partial [d_x^j \quad d_y^j]}{\partial \mathbf{c}}, \\ &\frac{\partial D}{\partial c_x} = \sum_j 2d_x^j \frac{\partial d_x^j}{\partial c_x}, \quad \frac{\partial D}{\partial c_y} = \sum_j 2d_x^j \frac{\partial d_x^j}{\partial c_y}, \\ &\frac{\partial E}{\partial c_x} = \sum_j 2d_y^j \frac{\partial d_y^j}{\partial c_x}, \quad \frac{\partial E}{\partial c_y} = \sum_j 2d_y^j \frac{\partial d_y^j}{\partial c_y}, \\ &\frac{\partial F}{\partial c_x} = \sum_j \left[\frac{\partial d_x^j}{\partial c_x} d_y^j + d_x^j \frac{\partial d_y^j}{\partial c_x} \right], \quad \frac{\partial F}{\partial c_y} = \sum_j \left[\frac{\partial d_x^j}{\partial c_y} d_y^j + d_x^j \frac{\partial d_y^j}{\partial c_y} \right]. \end{aligned}$$

where

$$\begin{aligned} &\frac{\partial [d_x^j \quad d_y^j]}{\partial \mathbf{c}} = \frac{1}{2\sqrt{\mathbf{P}_j}} \left[\sum_i (-1) g\left(\left\| \frac{\mathbf{x}_i^j - \mathbf{c}}{h} \right\|^2 \right) + \sum_i (\mathbf{x}_i^j - \mathbf{c})^T \frac{\partial g}{\partial \mathbf{c}} \right], \\ &\frac{\partial d_x^j}{\partial c_x} = \frac{1}{2\sqrt{\mathbf{P}_j}} \left[\sum_i (-1) g\left(\left\| \frac{\mathbf{x}_i^j - \mathbf{c}}{h} \right\|^2 \right) + \sum_i (\mathbf{x}_{ix}^j - c_x) \frac{\partial g}{\partial c_x} \right], \\ &\frac{\partial d_x^j}{\partial c_y} = \frac{1}{2\sqrt{\mathbf{P}_j}} \sum_i (\mathbf{x}_{ix}^j - c_x) \frac{\partial g}{\partial c_y}, \\ &\frac{\partial d_y^j}{\partial c_x} = \frac{1}{2\sqrt{\mathbf{P}_j}} \sum_i (\mathbf{x}_{iy}^j - c_y) \frac{\partial g}{\partial c_x}, \\ &\frac{\partial d_y^j}{\partial c_y} = \frac{1}{2\sqrt{\mathbf{P}_j}} \left[\sum_i (-1) g\left(\left\| \frac{\mathbf{x}_i^j - \mathbf{c}}{h} \right\|^2 \right) + \sum_i (\mathbf{x}_{iy}^j - c_y) \frac{\partial g}{\partial c_y} \right]. \end{aligned}$$

$\frac{\partial g}{\partial \mathbf{c}}$ for Gaussian kernel is omitted here, but is given in [1] When Epanechnikov kernel is used, i.e., $g(\mathbf{c}) = 1$ and $\frac{\partial g}{\partial \mathbf{c}} = 0$, the above equations will be reduced to much simpler forms. Notice that all the above values can be obtained by scanning the pixels in the kernel region only *once*, so the calculation is easy and efficient.

Fig. 3 shows some illustrative examples. In all 3 images in (a), we start from the kernel with red rectangle and end up with the kernel with the green one, led by the gradient-based search. Fig. 3(b) shows the descending κ_S. Fig. 3(c) shows the regions covered by the kernel, which is moving along the direction of the gradient towards the good placement.



(a) (b) (c)

Figure 3. column (a): the red rectangle indicates the start kernel position, the green one is the optimized kernel placement found by the gradient-based searching algorithm. column (b): the corresponding descending value of the κ_S. column (c): the regions covered by the kernel, which is moving along the direction of the gradient towards the good placement. The red line with a spot indicates the center of mass of each color component.

4. Multiple Kernel Placement

The kernel based tracking is no longer confined to single kernels. Multiple kernels have several advantages over single kernel. For, example, multiple kernels can alleviate the singularity and improve the kernel's observability to the motions [8][10]. Multiple kernels are better at handling tracking an object with complex structure, while a holistic representation based on a single kernel is cumbersome. In such a case, distributing the tracking task into several correlated sub-tasks would be viable. Another benefit is the save of the computation since each sub-task only needs a relatively small kernel.

We think good strategies to place multiple kernels are I) each kernel has a reliable tracking performance, i.e., at a good location, and based on which, II) the structure of the multiple kernels should remain stable through the sequence and be simple.

The first strategy can be addressed by the proposed method, that is, the gradient-based searching algorithm can find the good placements near the initialized ones. The second strategy states that those multiple kernels are expected to maintain an invariant structure, thus serving as an consistent description of the object with good fidelity but great simplicity.

If a kernel is good for tracking at the beginning, but is not suitable for tracking due to view or illumination changes afterwards, this kernel is considered unstable and should be pruned away. It is also required that the stable structure be simple, although the object could be complex. By this means, we can adopt the scheme of multiple collaborative kernel tracking [8], which has superior kernel observability than single kernel, to coordinate those representative kernels for an overall reliable tracking performance

However, there is no general answer to the question that what kind of structure of multiple kernels should be chosen, and this is in fact an ongoing research topic in computer vision [7]. For simplicity, we chose triangle, which is easy to manipulate and works well in many sequences.

For each triangle, we build a 2D histogram, recording the 2 internal angles of that triangle. We obtain the statistics of this 2D histogram over a training sequence. The most stable triangle, with all the internal angles exhibiting little variation, is expected to yield a peak in this 2D histogram. So, for each triangle formed by 3 kernels, we measure the entropy of its associated 2D histogram, and choose the one, which has the minimum entropy, to be the most stable triangle kernel structure.

Then, we use this triangle modelling for collaborative kernel tracking [8]. This optimal multiple kernel placement and the mining for stable structure are fully automatic after the initialization of a set of kernels at the very beginning. The initialization can be done either manually or evenly on the image grid.

5. Scale-invariant Kernel Placement

A placement is good for kernel if the condition number evaluated on that region is small. This placement is even better if its associated condition number achieves a local minimum, i.e., it is the only one that should be selected from its neighborhood. However, it is obvious that the condition number changes when the scale of the kernel changes, therefore, the local minima property could also vary w.r.t the scale.

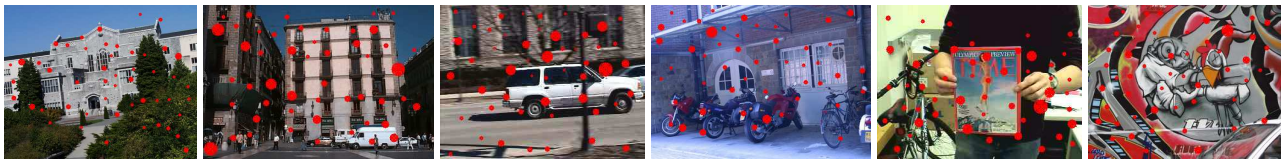
An interesting question is that *if there exists a placement of a kernel, whose condition number is the local minima for all, or for a large number of different, scales?* and how to find them?

Placing a kernel at such a place is invariant to the scale changes, meaning that, it can yield reliable tracking performance when changes occur in the kernel scale, or in the image scale, or in the scales of both image and kernel, since scale changes in the image and the kernel is just a relative term. This is in fact a very nice property.

In order to find such placements, rather than brute force evaluating the condition number through all the scales and neighborhood, the searching algorithm in Sec 3.4 offers considerable efficiency.

We can evenly initialize a set of kernel samples with different scales on the grid. Then, let these kernels converge to their corresponding local minima. When all the kernel samples are converged, a distribution of the places of converged kernels are obtained. The set of local maxima of such a distribution indicate the set of scale-invariant kernel placements.

In experiment, we generalize kernels with 7 different scales, each scale with 400 evenly initialized kernel samples. The places, to which a large amount of kernel sam-



(a) (b) (c) (d) (e) (f).

Figure 4. Scale-invariant kernel placement. Larger circle means higher density of convergent kernel samples.

ples are converged, are shown in Fig. 4. It can be seen that most of these places are featured by a region with diversified color pixels' intensity and spatial distributions.

The property of such placement for featured region selection, pattern recognition will be our main future work.

6. Discussions

6.1. Region selection vs. feature point selection

It may be noted that the form of Eq. (4) is similar to that of feature point matching [17][18], in that they are all in the general form of linear equation, i.e., **Ax=b**. Actually, the property of the matrix $\mathbf{A}$ has been actively studied in the past decades for point matching, such as computing the optical flow [3][15][16][18]. Some work also extends the point matching framework to address other geometrical features such as lines [12][20].

However, our work is different from those work in the following aspects.

1) The content of matrix A, denoted as $\mathbf{A}_{region}$ in our work, and $\mathbf{A}_{point}$ in [17][18].

$$\mathbf{A_{region}} = \mathbf{M} = \begin{bmatrix} d_x^1 & d_y^1 \\ \vdots & \vdots \\ d_x^m & d_y^m \end{bmatrix}, \qquad \mathbf{A_{point}} = \begin{bmatrix} g_x^1 & g_y^1 \\ \vdots & \vdots \\ g_x^n & g_y^n \end{bmatrix}.$$

where $[d_x^j \quad d_y^j]$ is the center of mass of all pixels of color j, for m color bins, $j = 1, ..., m$, and $[g_x^i \quad g_y^i]$ being the image gradient of pixel i w.r.t x and y axes, for n pixels within a small window around the feature point, $i = 1, ..., n$.

The different content determines that their analytical results are different. That is, there is no certain correspondences between optimal feature points and optimal regions for tracking. We think these works have different practical impacts in real applications.

2) The criterion and the analytical method In our work, the criterion is $\kappa_S(\mathbf{M}^T\mathbf{M})$, which has the equivalent effect as analyzing $\kappa_2(\mathbf{M}^T\mathbf{M})$. In [18], the criterion is that the smallest eigenvalue of $\mathbf{A}^T\mathbf{A}$ is larger than a threshold. In [17], $\|(\mathbf{A}^T\mathbf{A})^{-1}\|_S$ is checked, which has been shown to have the same effect as requiring the smallest eigenvalue of $\mathbf{A}^T\mathbf{A}$ to be larger than a threshold. In contrast, our criterion takes into account of both singular values of $\mathbf{M}$, which is more general.

As for the analytical method, we have shown that κ_S is equivalent to κ_2 when considering the 2×2 covariance matrix case, which leads to some nice analytical properties.

3) Gradient descent search We provide a gradient based searching scheme to find the optimal regions within an image efficiently, which avoids exhaustive search. But the selection of feature point has to be exhaustive.

6.2. Interpretation of condition number criterion using the S-norm

We already know the structure of $\mathbf{M} = [\mathbf{v_x} \quad \mathbf{v_y}]$, where $\mathbf{v_x} = [d_x^1, \ldots, d_x^m]^T$, $\mathbf{v_y} = [d_y^1, \ldots, d_y^m]^T$ are the X-coordinates and Y-coordinates of the the centers of color masses, respectively. Then we have

$$\sum_j (d_x^j)^2 = \|\mathbf{v_x}\|^2, \quad \sum_j (d_y^j)^2 = \|\mathbf{v_y}\|^2,$$
$$(\sum_j (d_x^j d_y^j))^2 = \|\mathbf{v_x}^T\mathbf{v_y}\|^2 = \|\mathbf{v_x}\|^2\|\mathbf{v_y}\|^2\cos^2(\theta).$$

where θ is the angle between vectors $\mathbf{v_x}$ and $\mathbf{v_y}$.

Recall Eq.(12),

$$\begin{aligned} \kappa_S(\mathbf{M}^T\mathbf{M}) &= \frac{(\|\mathbf{v_x}\|^2+\|\mathbf{v_y}\|^2)^2}{\|\mathbf{v_x}\|^2\|\mathbf{v_y}\|^2-\|\mathbf{v_x}\|^2\|\mathbf{v_y}\|^2\cos^2(\theta)} \\ &= \frac{\frac{\|\mathbf{v_x}\|^4+\|\mathbf{v_y}\|^4}{\|\mathbf{v_x}\|^2\|\mathbf{v_y}\|^2}+2}{1-\cos^2(\theta)} \geq \frac{4}{1-\cos^2(\theta)} \geq 4. \end{aligned}$$

It is easy to verify that the desirable minimum of the above condition number is achieved when

$$\|\mathbf{v_x}\| = \|\mathbf{v_y}\|, \quad \text{and} \quad cos(\theta) = 0, \text{i.e.,} \mathbf{v_x} \perp \mathbf{v_y},$$

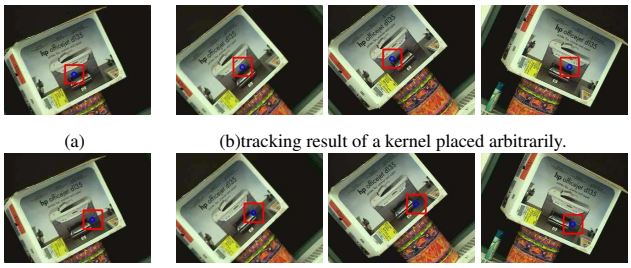
which implies that the roles of X and Y coordinates of those centers of color masses are equivalent and interchangeable. The optimal case would be that these mass centers are located symmetrically around the center of kernel $\mathbf{c}$. This is actually the same as the interpretation we have given in Sec 3.2, but from another point of view. A perfect example is already shown in the left most column of Fig. 1.

7. Experiment

In this section, experiments using real video sequences demonstrate the effectiveness and usefulness of the proposed algorithm for efficient optimal kernel placement.

7.1. Single kernel

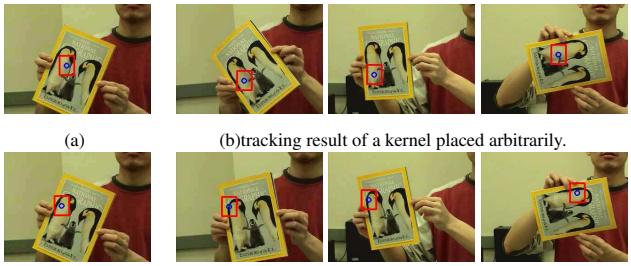
For an object of interest, arbitrarily labelling a region, which meets some special standards, such as owning a high



(a) (b)tracking result of a kernel placed arbitrarily.

(c) (d) tracking result of a kernel with initial location optimized.

Figure 5. Tracking with (bottom row) and without (top row) kernel placement optimization.



(a) (b)tracking result of a kernel placed arbitrarily.

(c) (d) tracking result of a kernel with initial location optimized.

Figure 6. Tracking with (bottom row) and without(top row) kernel placement optimization.

variance, strong edges, or a high entropy, and assigning a kernel on it, may not be optimal. In many cases, unexpected tracking failures make people change the kernel placement through trial and error.

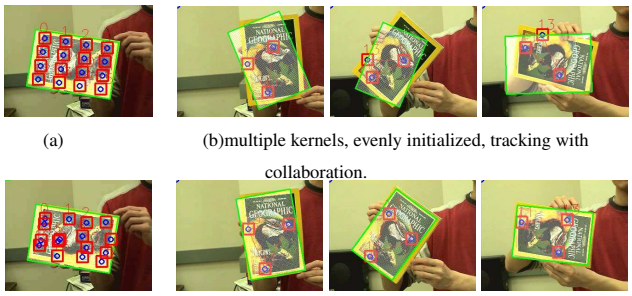
On the contrary, placing kernels by the criterion presented in section 3 and the efficient searching algorithm derived in section 4 can easily bring more reliable tracking performance.

Fig. 5(b) shows the tracking result with a kernel initialized as in Fig. 5(a). The tracking is not stable, since the unidirectional color distribution in the initialized place yields a large condition number. Using the same initialization as 5(a), we apply the *gradient-based algorithm* and find a good kernel placement, with much lower condition number, as shown in 5(c), the corresponding tracking result is shown in 5(d). More reliable performance is obtained.

Fig. 6(a) shows an arbitrarily initialized kernel placement, the corresponding tracking result is in Fig. 6(b), in which drifting is observed. In contrast, the searching algorithm moves the place from 6(a) to a good placement as in 6(c). The tracking performance is instantly improved a lot, as shown in 6(d). Notice that the location in Fig. 6(a) and Fig. 6(c) is very close, and this is indeed difficult for manual initialization, as heedlessly marking a region, to achieve a robust performance. This shows that the searching algorithm effectively help us to discriminate good and bad regions for tracking.

7.2. Multiple collaborative kernels

Good strategies to place multiple kernels are that I) each kernel is at a good placement, II) the structure of the multiple kernels should be stable.



(a) (b)multiple kernels, evenly initialized, tracking with collaboration.

(c) (d) multiple kernels, location optimized, tracking with collaboration.

Figure 7. Multiple kernel tracking, with (bottom row) and without (top row) placement optimization.

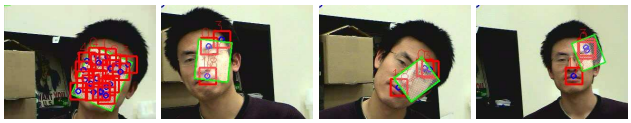
To track a region of interest, we initialize a set of kernels on the grid, and run the searching algorithm to find good placements. By this means, we can have a large coverage of the possible multiple kernel combinations and safely avoid the risk of having bad manually labelled regions.

Then we track these kernels over a training sequence, which can be the first several frames of the video. We choose the most stable triangle formed by 3 kernels.

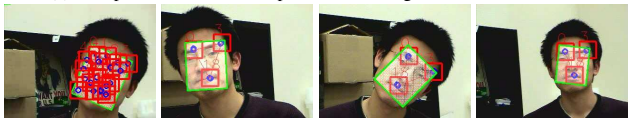
In Fig. 7(a), multiple kernels are evenly initialized on the grid within the region of interest. *Without* the gradient searching algorithm, that is, we just accept and start from the initial kernel locations, mine for the most stable triangle and apply collaborative tracking scheme. The result is shown in Fig. 7(b). Fig. 7(c) shows the good kernel placement found by applying the gradient searching algorithm on the kernel locations in 7(a), the corresponding tracking result is shown in 7(d). In all the figures, the bounding box of the object of interest is reconstructed by conferring the initial localization of the kernels w.r.t. the object. How well can we know the position and orientation of the original object measures the quality of collaborative tracking. It is seen that the performance of kernels, whose locations are optimized, is much better. This is because the kernels, without placement optimization, are more likely to have a large condition number and thus having more exposure to the unstableness in the tracking.

The next two sequences involve object scale changes, in which the optimal multiple collaborative kernel tracking works well.

Fig. 8(a) and Fig. 8(b) show the tracking result from the same initial kernel locations (left most column), after searching from the evenly grid initialization. *Without* collaborative scheme applied, the unsatisfactory result in Fig. 8(a) shows that, although all the kernels are good at the beginning, they still can lost due to various disturbances in tracking, such as, unexpected abrupt motions, disturbance from the object with similar appearance or illumination changes. Since the kernels track independently, they

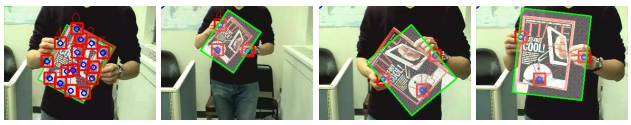


(a) multiple kernels, location optimized, tracking without collaboration.

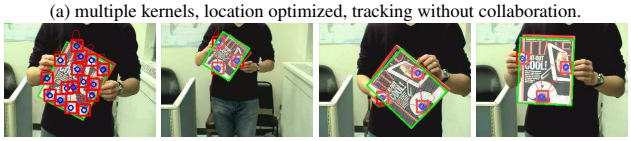


(b) multiple kernels, location optimized, tracking with collaboration.

Figure 8. Multiple kernel tracking, with (bottom row) and without (top row) collaboration.



(a) multiple kernels, location optimized, tracking without collaboration.



(b) multiple kernels, location optimized, tracking with collaboration.

Figure 9. Multiple kernel tracking, with (bottom row) and without (top row) collaboration.

cannot recover themselves, such that the face is lost track. By collaborating the initially good kernels, the result of localizing the face by the 3 kernels is more reliable, as shown in Fig. 8(b).

Fig. 9 has the similar settings as in Fig. 8. The result with collaboration, row (b), again yields much more reliable reconstruction performance of estimating the position, orientation and the scale of the magazine cover.

8. Conclusion

To summarize, in this paper, we present a detailed analysis to the criterion of optimal kernel placement. An equivalent criterion is also derived, which has a closed-form representation and enables a nice gradient-based algorithm to find optimal kernel placement efficiently. Placement of temporal-stable multiple kernels and scale-invariant kernels are also studied. These new theoretical results and new algorithms help us to better understand and implement the kernel-based tracking method.

Acknowledgement

This work was supported in part by National Science Foundation Grants IIS-0347877, IIS-0308222 and Northwestern faculty startup funds.

References

[1] Z. Fan and Y. Wu, "Efficient Optimal Kernel Placement for Reliable Visual Tracking", Technical Report, ECE Dept. Northwestern University, Nov. 2005. 3, 4

[2] J. Benesty and T. Gansler, "A recursive estimation of the condition number in the RLS algorithm", *Proc. IEEE Conference on Acoustics, Speech, and Signal Processing*, 2005, pp. 25-28. 3

[3] M. Brooks, W. Chojnacki, D. Gaeley, and A. Hengel, "What value covariance information in estimating vision paramaters", *Proc. International Conference on Computer Vision*, 2001, pp. 302-308. 3, 6

[4] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002, vol. 24, no. 5, pp. 603-619. 1

[5] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2000, vol. 2, pp. 142-149. 1, 2

[6] D. Comaniciu, V. Ramesh, and P. Meer, "The variable bandwidth mean shift and data-driven scale selection", *Proc. International Conference on Computer Vision*, 2001, vol. 1, pp. 438-445. 1

[7] D. Crandall, P. Felzenszwalb, and D. Huttenlocher, "Spatial priors for part-based recognition using statistical models", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005, vol. 1, pp. 10-17. 5

[8] Z. Fan, Y. Wu, and M. Yang, "Multiple Collaborative Kernel Tracking", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005. 1, 5

[9] G.H. Golub and C.F. Van Loan, "*Matrix computations*", 2nd edition, The John Hopkins University Press, Baltimore, MD, 1989. 3

[10] G. D. Hager, M. Dewan, and C. V. Stewart, "Multiple kernel tracking with SSD", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2004, vol. 1, pp. 790-797. 1, 2, 5

[11] R.A. Horn and C.R. Johnson, "*Topics in matrix analysis*", Cambridge, Mass.: MIT Press, 1991. 3

[12] M. Irani and P. Anandan, "Robust multi-sensor image alignment", *Proc. International Conference on Computer Vision*, 1998, pp. 959-966. 6

[13] M. Isard and A. Blake, "CONDENSATION – conditional density propagation for visual tracking", *Int'l J. Computer Vision*, 1998, vol. 29, pp. 5-28. 1

[14] T. Kadir, A. Zisserman, and M. Brady, "An affine invariant salient region detector", *ECCV*, 2004.

[15] Y. Kanazawa and K. Kanatani, "Do we really have to consider covariance matrices for image features?" *Proc. International Conference on Computer Vision*, 2001, pp. 301-306 3, 6

[16] J.K. Kearney, W.B. Thompson, and D.L. Boley, "Optical flow estimation: an error analysis of gradient-based methods with local optimization", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1990, vol. 9, pp. 229-244. 3, 6

[17] C.S. Kenney, B.S. Manjunath, M. Zuliani, G.A. Hewer, and A.V. Nevel, "A condition number for point matching with application to registration and postregistration error estimation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003, vol. 25, pp. 1437-1454. 3, 6

[18] J. Shi and C. Tomasi, "Good features to track", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593-600. 3, 6

[19] J. Wang, B. Thiesson, Y. Xu, and M. F. Cohen, "Image and video segmentation by anisotropic kernel mean shift", *Proc. European Conference on Computer Vision*, 2004. 1

[20] R. Wildes, D. Horvonen, S. Hsu, R. Kumar, W. Lehman, B. Matei, and W. Zhao, "Video georegistration: algorithm and quantitative evaluation", *Proc. International Conference on Computer Vision*, 2001, pp. 343-350. 6