Linear Scale and Rotation Invariant Matching

Hao Jiang, *Member*, *IEEE*, Stella X. Yu, *Member*, *IEEE*, and David R. Martin

Abstract—Matching visual patterns that appear scaled, rotated, and deformed with respect to each other is a challenging problem. We propose a linear formulation that simultaneously matches feature points and estimates global geometrical transformation in a constrained linear space. The linear scheme enables search space reduction based on the lower convex hull property so that the problem size is largely decoupled from the original hard combinatorial problem. Our method therefore can be used to solve large scale problems that involve a very large number of candidate feature points. Without using prepruning in the search, this method is more robust in dealing with weak features and clutter. We apply the proposed method to action detection and image matching. Our results on a variety of images and videos demonstrate that our method is accurate, efficient, and robust.

Index Terms—Scale and rotation invariant matching, deformable matching, linear programming, action detection, shape matching, object matching.

1 Introduction

Finding the point-to-point correspondence of related visual patterns is a fundamental problem in computer vision. Many applications such as stereo, motion estimation, object detection, and action detection benefit from accurate and fast matching algorithms. Visual matching is also a challenging problem because the imagery of an object may have large variations, e.g., geometrical transformations, deformations, and appearance changes in different viewing conditions. In this paper, we propose an efficient linear solution to the visual matching problem which yields reliable results even when there is large rotation, scaling, translation, deformation, and clutter.

Different methods have been proposed to solve matching problems. The Hough transform [36], [1], [2] and RANSAC [35], [3], [34] are robust methods that have been widely used in object matching. The Hough transform requires a careful selection of its parameters (e.g., bin size) and breaks down in the presence of strong clutter. RANSAC is more resistant to clutter: It generates matching hypotheses for a small number of anchor points and then evaluates the hypotheses using all the model points. RANSAC becomes increasingly slow with weak features or strong clutter. To reduce the complexity, prepruning methods based on heuristics or General Hough Transform have been used [3] to remove unpromising local matches. However, when local feature matching becomes increasingly ambiguous, the prepruning process is less and less effective. Besides, the Hough transform and RANSAC find the overall geometrical transformation, but do not directly give point-to-point deformable object matching.

- *H. Jiang and S.X. Yu are with the Computer Science Department, Boston College, 140 Commonwealth Avenue, Chestnut Hill, MA 02467. E-mail: {hjiang, syu}@cs.bc.edu.*
- *D.R. Martin is with Google Inc., Mountain View, CA 94043. E-mail: david.r.martin@gmail.com.*

Manuscript received 18 Aug. 2009; revised 12 Feb. 2010; accepted 5 Oct. 2010; published online 29 Nov. 2010.
Recommended for acceptance by J. Kosecka.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-08-0547.
Digital Object Identifier no. 10.1109/TPAMI.2010.212.

Another class of methods, graph matching, directly optimizes the point-to-point correspondence. By optimizing an energy function that contains a unary data term and pairwise or higher order smoothness terms, graph matching enforces the matching to be consistent. Graph matching is NP-hard in general. For special cases where the graphs have no loops or the target candidates have linear orders, exact polynomial-time algorithms such as dynamic programming (DP) [4] and max flow [5] can be used. For general graph matching in which the graph template contains cycles, an exact solution is often too slow to be feasible for large scale problems.

Various approximate methods for graph matching have been developed. Iterative Conditional Modes (ICM) [7] is a local optimization method that gets easily trapped in local optima. Back tracking [6] with heuristics for search tree pruning has been proposed to explore the search space globally. This method has been successfully applied to rigid object detection. Graph cuts [8] and belief propagation (BP) [10] are recent global search methods that have been successfully applied to a range of matching problems, including stereo [9], [11], motion estimation [17], object pose estimation [13], tracking [12], and recognition [14]. BP often has a linear to quadratic complexity with respect to the number of target candidates and becomes slow when searching over large ranges in target images. Specialized message passing methods [41] have been proposed to improve the efficiency of BP. However, for large scale problems that contain a very large number of target candidates, the algorithm's complexity is still quite high.

Mathematical programming is another approach to graph matching. Soft-assign [15] with its extension [16] is one of the few methods that handle large object deformations. It employs an iterative routine that alternates between point matching and global transformation estimation. Combined with shape context, this scheme has been used in shape matching [20]. Recently, global search methods have received a lot of interest. Concave programming [32] is proposed to match point sets. Spectral graph methods [33] approximate the discrete optimization using convex relaxation and find point correspondence by solving eigenvector

problems. Another convex programming [42] method, semidefinite programming, has also been used to relax a graph matching problem so that feature matching can be globally optimized [37], [38].

The high complexity of general mathematical programming for large scale problems limits its usage in practical applications, while linear programming, the simplest form of mathematical programming, is found to be a powerful method for solving visual matching problems both reliably and efficiently. Jiang et al. [19] propose an efficient linear programming method that does not take into account the scaling and rotation in feature point matching. Berg et al. propose an integer quadratic formulation and a linear relaxation [18] for scale-invariant object matching. Torresani et al. [39] propose a dual decomposition scheme to relax the integer quadratic program. Komodakis and Paragios [40] propose another dual decomposition method for linear relaxation. Glocker et al. [22] use primal-dual linear programming in medical image registration. The linear formulation is based on [25] and this method only computes local deformations. Taylor et al. [23] propose an interior point method for image registration. A novel method is proposed to take advantage of the special structure of this formulation to handle large scale problems. To further improve the efficiency, a multiple scale method is applied. Komodakis and Tziritas [24] derive a graph-cuts-like optimization methods for metric labeling based on primal-dual linear programming. This method improves the result on motion estimation, stereo, and image denoising. Shekhovtsov et al. [26] propose a linear method that models local affine constraints. Message passing on trees (TRW-S) is used to solve the linear program efficiently. This method is applied to 2D local deformable matching. Most current linear methods for image matching focus on metric labeling and applications such as stereo and motion estimation. They are not easily extended to matching problems that involve large rotations and scale changes.

Even though intensively studied, optimizing scale and rotation invariant point-to-point visual matching is still a hard problem, especially for weak features and problems with a large number of candidate points. In this paper, we study how to explicitly incorporate scale and rotation inference in a linear programming framework and how it can be efficiently solved. We propose a novel linear formulation of scale and rotation invariant matching. Using the lower convex hull property, we can effectively solve the linear problem on a small number of lower convex-hull vertex variables. Our method thus has a complexity rather independent of the number of target candidates, making it suitable for very large scale problems. The result can be further improved by successively shrinking trust regions. Our extensive experimentation demonstrates that the proposed linear solution is accurate, fast, and robust, and it works well with both scale and rotation invariant features such as SIFT [3] and noninvariant features such as shape context [20], shape flow [30], and simple image patches. The proposed method has been successfully applied to action detection [30] and object matching [31].

The arrangement of the rest of the paper is as follows: In Section 2, we formulate scale and rotation invariant matching into linear programs. We propose how to simplify the linear programs for efficient solution, and we study a successive scheme to refine the linear approximation for accurate results. In Section 3, we benchmark the proposed method using synthetic data and compare it with other state of the art methods. In Section 4, we illustrate how the proposed method can be used in action detection [30] and object matching [31]. We conclude the paper in Section 5.

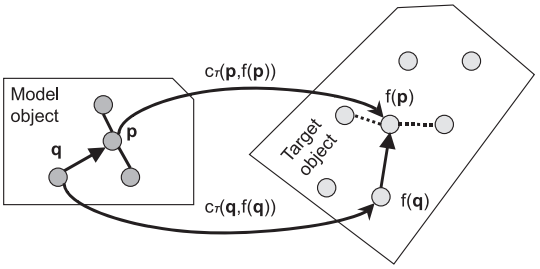


Fig. 1. Our matching criterion minimizes feature matching cost and spatial matching cost. The feature matching cost is determined by the difference of the feature vectors associated with each model point $\mathbf{p}$ and its match $\mathbf{f}(\mathbf{p})$. The matching cost may depend on the object global transformation T. The spatial matching cost is determined by the difference between the vectors $(\mathbf{p},\mathbf{q})$ and $(\mathbf{f}(\mathbf{p}),\mathbf{f}(\mathbf{q}))$, which is invariant to unknown global scaling and rotation.

2 Scale and Rotation Invariant Matching

Given two sets of points, each point is associated with a single feature vector or a sequence of feature vectors for different global transformations; we would like to find the point-to-point correspondence of the model set with the target set that is a globally translated, scaled, rotated, and locally deformed version of those model points embedded in irrelevant clutter points (Fig. 1). In the following, we show how scale and rotation invariant matching can be formulated and further simplified into linear programs.

2.1 Criterion

Our goal is to find, for every model point, a corresponding target point so that they share similar local features and pairwise spatial connections. Formally, let $\mathcal{M}$ be the set of model points and $\mathcal{N}$ the set of all pairs of neighboring model points. Let $\mathbf{f}(\mathbf{p})$ be the target point matched to model point $\mathbf{p}$. The objective function minimizes both feature and spatial matching costs:

$$\min_{\mathbf{f},T}\left\{\sum_{\mathbf{p}\in\mathcal{M}} c_T(\mathbf{p},\mathbf{f}(\mathbf{p})) + \lambda \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} g_T(\mathbf{p},\mathbf{q},\mathbf{f}(\mathbf{p}),\mathbf{f}(\mathbf{q}))\right\}.$$

Here, c_T is the feature matching cost, which is small if the model point $\mathbf{p}$ and target point $\mathbf{f}(\mathbf{p})$ have small feature difference under the global transformation T; g_T is the spatial matching cost, which is small if the spatial connection $(\mathbf{p},\mathbf{q})$ is similar to $(\mathbf{f}(\mathbf{p}),\mathbf{f}(\mathbf{q}))$ under the global transformation T; and λ controls the relative weight of the two terms. Since we directly match points from a template to targets, such a formulation is able to deal with arbitrary translations.

We can write the scale and rotation invariant objective function as

$$\min_{\mathbf{f},s,R}\left\{\sum_{\mathbf{p}\in\mathcal{M}} c_{s,R}(\mathbf{p},\mathbf{f}(\mathbf{p}))+\lambda\sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}}\|R\cdot(\mathbf{p}-\mathbf{q})-s\cdot(\mathbf{f}(\mathbf{p})-\mathbf{f}(\mathbf{q}))\|\right\}, \tag{1}$$

where s and R are unknown scaling factor and rotation matrix, respectively. In this paper, we are most interested in $c(.)$ that is scale or rotation invariant, for which the proposed method achieves the most efficiency. Apart from scale and rotation invariance, this formulation is also invariant to translations. As shown later, even for non-invariant matching costs the proposed method can still be used for efficient optimization. In contrast to other ways of constructing a scale and rotation invariant matching objective function, e.g., by constraining the pairwise lengths between model point pairs and target pairs, our formulation can be converted to a simpler linear formulation and solved efficiently.

The nonlinear optimization problem in (1) involves both discrete and continuous variables. For real applications, there are a large number of model points and target points. Exhaustive search is not an option. Our idea is to convert the problem into a small set of convex programs which can be efficiently solved. For the rest of the paper, we assume points are in 2D and their spatial matching cost is measured with the L_1 norm, although our approach can be extended to higher dimensions, as well as the L_2 norm.

To facilitate the discussion, we start from a formulation in which the local feature matching cost is scale and rotation invariant, i.e., $c_{s,R}(\mathbf{p},\mathbf{f}(\mathbf{p}))$ can be written as $c(\mathbf{p},\mathbf{f}(\mathbf{p}))$, which is only dependent on the model point and target candidate point locations. In the following sections, we first write the optimization in matrix formulation; then, we study how to linearize it so that it can be relaxed into linear programs; we further show how the linear formulation can be extended to accommodate general local cost functions; finally, we study how to efficiently solve the linear programs using the variable reduction method.

2.2 Matrix Formulation

It helps to clarify the formulation by writing the optimization in a succinct matrix form. We introduce an assignment matrix X to express the matching from model points to target points. Let 1_n denote a column vector of n 1s, $'$ matrix transpose, tr the trace of a matrix, and $|\cdot|$ the summation of absolute values of all the elements in a matrix. In matrix form, the optimization becomes

$$\begin{aligned}&\min\ \varepsilon(X,s,R)=\mathrm{tr}(C'X)+\lambda|EMR-sEXT|\\&\text{subject to } X1_{n_t}=1_{n_m}, X\in\{0,1\}^{n_m\times n_t}\\&\qquad s>0, R'R=I,\end{aligned} \tag{2}$$

where n_m denotes the number of model points $|\mathcal{M}|$ and n_t the number of target points. Here, to simplify the discussion, we assume model to target point matching costs to be invariant. The invariance indicates that the overall matching costs are only related to the positions of model and target points. We will show how this condition can be relaxed in the later sections.

There are three unknown variables:

$X=n_m\times n_t$ binary assignment matrix. Each row of X contains exactly one 1: $X(i,j)=1$ indicates that model point i is matched to target point j.
$s=$ global scaling factor.
$R=2\times 2$ global coordinate rotation matrix. It is in fact the transpose of the R in (1).

and four known matrices:

$M=n_m\times 2$ model point coordinate matrix.
$T=n_t\times 2$ target point coordinate matrix.
$C=n_m\times n_t$ feature matching cost matrix. $C(i,j)$ is the feature matching cost between model point i and target point j which is computed using $c(.)$ in (1) with the assumption that it is scale and rotation invariant.
$E=n_e\times n_m$ edge-node incidence matrix for the model graph, where $n_e=|\mathcal{N}|$. Each row describes an edge with exactly two non-zero numbers: 1 and -1, and their signs can be switched. For example, $E(e,i)=1$, $E(e,j)=-1$ indicate edge e connects nodes i and j in the model.

We make four additional comments regarding the formulation.

1. In our model, the constraint on X does not force model points to match unique target points. Matching multiple model points to a single target point is in fact necessary if the target object shrinks. The unit sum constraint on rows of X also implies that each model point matches a target point. This assumption is shown to be not restrictive. As will be shown later, the occluded model points can still match correct target points based on the structure constraint.
2. XT, which will be used to compute the target locations, is involved in spatial regularization. It represents valid target locations even when X is relaxed into floating-point numbers, in which case the target point locations are linear combinations of target candidate points.
3. We can take reflection into account by dropping $s>0$: A negative s simply means that the spatial connection in the image could be a mirror reflection of that in the model (after rotation), scaled by factor $|s|$. Our method can be extended accordingly.
4. It is essential to separate scale and rotation in the spatial matching cost. If we combine scale s and rotation R into one similarity transform $S=R/s$, i.e.,

$$\min\ \varepsilon(X,S)=\mathrm{tr}(C'X)+\lambda|EMS-EXT|,$$

we would introduce a strong bias favoring small scales. Even though the formulation is seemingly simpler, it is wrong and results in matching a small pattern in the image.

2.3 Linearization

Instead of directly solving the hard mixed integer nonlinear program in (2), we convert it into linear problems which can be efficiently solved. There are three obstacles in linearizing (2): 1) the L_1 norm in the spatial matching term, 2) the

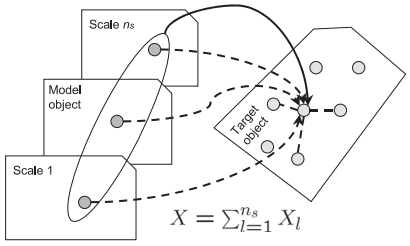


Fig. 2. Scale linearization. The assignment matrix X is expanded with another dimension, the scale, and form $\{X_l\}$; the scaled assignment term sX in the spatial matching cost becomes linear.

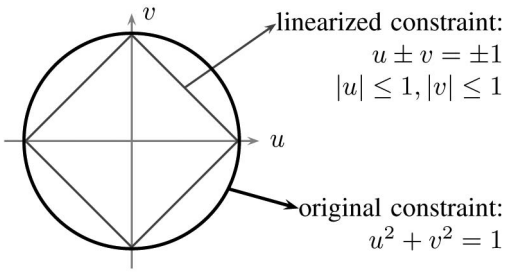


Fig. 3. Rotation linearization. The orthonormal constraint on the rotation matrix R is approximated by four line segments.

nonlinearity introduced by the multiplication of the integer variable X and the continuous variable s, and 3) the quadratic constraint on the rotation matrix R.

2.3.1 L_1 Norm Linearization

We introduce two $n_e \times 2$ nonnegative auxiliary matrices, Y and Z, to turn the L_1 norm optimization into a linear objective with linear constraints. It is well known that

$$\begin{aligned} \min|x| \Leftrightarrow \min \quad & y + z \\ \text{subject to} \quad & y - z = x \\ & y \geq 0, z \geq 0. \end{aligned}$$

Applying to every element of $EMR - sEXT$, we have

$$\begin{aligned} &\min \varepsilon(X, s, R, Y, Z) = \text{tr}(C'X) + \lambda 1'_{n_e}(Y + Z)1_2 \\ &\text{subject to } Y - Z = EMR - sEXT \\ &\qquad Y \geq 0, Z \geq 0. \end{aligned} \tag{3}$$

Intuitively, for each element pair in Y and Z, at most one of them is nonzero as ε is minimized. Otherwise, we can always subtract the values with the smaller one of each pair, zeroing out at least one of the values; the solution remains feasible and achieves lower energy, which contradicts the assumption that the ε has achieved the minimum. Based on this property, the sum of all the elements in Y and Z, when the objective is minimized, must equal $|EMR - sEXT|$.

2.3.2 Scale Term Linearization

The quadratic term $sEXT$ in (3) has to be linearized. The idea is that we quantize the scale into discrete levels and we further introduce matching variables X_l at multiple scales so that

$$X = \sum_l X_l.$$

Ideally, X_l, $l = 1..n_s$, should contain a single 1 at the correct scale corresponding to the true target point; other elements in X_l, $l = 1..n_s$, should all be zero. Matrices X_l, $l = 1..n_s$, therefore augment the assignment matrix X with a new dimension, the scale. It is helpful to imagine the stack of X_l, $l = 1..n_s$, as a 3D matrix which indicates the assignment at a specific location and a specific scale. If we collapse X_l, $l = 1..n_s$, along the dimension of scale, we obtain X. We can now transform the multiplication of s and X in (3) into a linear function of X_l subject to linear constraints among s, X_l, and X. Illustrated in Fig. 2, s is quantized into n_s discrete values, $0 < s_1 < \cdots < s_{n_s}$. Fig. 2 shows how the matching from a model point to a target point is augmented with this new scale variable. This can also be viewed as matching multiple templates in different scales to the target; we therefore have to decide which scale needs to be used. The nonlinear term sX can now be represented as a linear term:

$$sX = \sum_{l=1}^{n_s} s_l X_l,$$

and therefore

$$sEXT = \sum_{l=1}^{n_s} s_l E X_l T.$$

Recall that since $X1_{n_t} = 1_{n_m}$, we also have the constraint

$$\sum_{l=1}^{n_s} s_l X_l 1_{n_t} = s1_{n_m},$$

which forces each model point to select the same scale in matching. As we relax the binary constraints on X_l to any value within $[0, 1]$, s is no longer restricted to a discrete level s_l, but can be any real number within $[s_1, s_{n_s}]$.

2.3.3 Rotation Term Linearization

We reparameterize the rotation matrix R in terms of its elements u, v, and approximate the orthonormal constraint $u^2 + v^2 = 1$, i.e., a circle in the uv plane, with four line segments (Fig. 3):

$$\begin{aligned} R'R = I \quad \approx \quad & R = \begin{bmatrix} u & -v \\ v & u \end{bmatrix} \\ & u \pm v = \pm 1, \quad |u| \leq 1, \ |v| \leq 1. \end{aligned}$$

Even though more line segments may refine the approximation, they also introduce more linear programs and increase the complexity. We found four line segments present sufficient approximation. The linear approximation in fact introduces different scales for different rotation angles. However, since we optimize the scale and rotation simultaneously, the distortion does not pose a problem.

2.4 The Linear Program

Overcoming the three obstacles, we reach a complete linearization of the original optimization problem in (2):

$$
\begin{aligned}
\text{LP}: \quad & \min \varepsilon(X, s, u, v, Y, Z, X_1, \ldots, X_{n_s}) \\
& = \operatorname{tr}(C'X) + \lambda 1'_{n_e}(Y+Z)1_2, \\
\text{subject to } & Y - Z = EM\begin{bmatrix} u & -v \\ v & u \end{bmatrix} - \sum_{l=1}^{n_s} s_l E X_l T \\
& Y, Z \geq 0, \quad u \pm v = \pm 1, \quad |u| \leq 1, \quad |v| \leq 1 \\
& X = \sum_{l=1}^{n_s} X_l, \quad X_l \geq 0, \forall l \\
& \sum_{l=1}^{n_s} s_l X_l 1_{n_t} = s 1_{n_m} \\
& X 1_{n_t} = 1_{n_m}, X \geq 0.
\end{aligned} \tag{4}
$$

If we constrain X and X_l to be binary matrices, this linear mixed integer program is equivalent to the original problem. We relax X into continuous domain so that the optimization can be efficiently solved. The optimal target point coordinates are computed by $T^* = XT$, after the optimal X is found for the LP.

2.5 Extensions to More General Cases

In the above formulation, we assume that the matching cost matrix C is invariant to both scale and rotation. We can extend the formulation to more general cases.

Sometimes, it is convenient to use matching costs that are rotation invariant but are not scale invariant. We denote the matching cost matrices at scale 1 to scale n_s as $C_1, C_2, \ldots, C_{n_s}$ and the linear program can be rewritten as

$$
\min\left\{\sum_l \operatorname{tr}(C_l' X_l) + \lambda 1'_{n_e}(Y+Z)1_2\right\}, \tag{5}
$$

with the same set of constraints.

We can further extend the linear formulation to the case in which the matching costs are neither scale nor rotation invariant. Similar to the scale linearization, we quantize the rotation angles into discrete levels $\theta_1, \theta_2, \ldots, \theta_{n_\theta}$. Let $X_{l,k}$ indicate the assignment at scale l and rotation k. X_l and $X_{l,k}$ are related by

$$
X_l = \sum_k X_{l,k}.
$$

We further introduce a matrix X_{θ_k} and let

$$
X_{\theta_k} = \sum_l X_{l,k} \text{ and } X = \sum_k X_{\theta_k}.
$$

We require that each model point should choose the same rotation angle in matching

$$
\sum_{k=1}^{n_\theta} \theta_k X_{\theta_k} 1_{n_t} = \theta 1_{n_m}.
$$

With constraints in (4), we optimize the objective function

$$
\min\left\{\sum_{l,k} \operatorname{tr}(C'_{l,k} X_{l,k}) + \lambda 1'_{n_e}(Y+Z)1_2\right\}, \tag{6}
$$

where $C_{l,k}$ is the matching cost at scale l and rotation k.

2.6 Lower Convex Hull Property

The LPs in (4), (5), or (6) are much easier to solve than the original mixed integer program, but a direct solution would still be slow because the number of variables is proportional to $n_m \times n_t \times n_s$. Fortunately, we do not need to solve the large LPs. In the linearized formulation, not all of the target points are effective. By removing the ineffective target points, we can simplify the linear programs without changing the optimal solution.

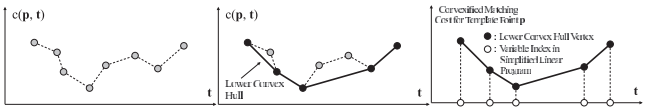


Fig. 4. Convexification using lower convex hull. The actual cost surface of the integer program is nonconvex (a). Relaxing the variables in X yields a linear program, the solution for which lies on the lower convex hull (b) of the original cost surface. Any variables corresponding to points above this surface are redundant in the linear program, so they may be pruned (c).

Property 1. *If the local matching cost is scale and rotation invariant, for each model point* $\mathbf{p}$*, the effective target points* $\mathbf{t}$ *and their associated feature matching costs* $c(\mathbf{p}, \mathbf{t})$ *correspond to the vertices of the lower convex hull of the point cloud* $\{(\mathbf{t}, c(\mathbf{p}, \mathbf{t})) : \forall \mathbf{t}\}$*. Here,* $c(\mathbf{p}, \mathbf{t})$ *corresponds an element of* C *to the model point* $\mathbf{p}$ *and target candidate point* $\mathbf{t}$*. We define the target points "effective" if they are used in the linear program construction.*

Recall that the LP in (4) uses a linear combination of target point costs to approximate the original matching cost and the target location is a linear combination of the target candidate points, i.e., XT. If we expand the matrix formulation, for each model point $\mathbf{p}$ we have $\sum_{\mathbf{t}} \mathbf{t} \cdot x_{\mathbf{p},\mathbf{t}} = \mathbf{t}_{\mathbf{p}}$ and $\sum_{\mathbf{t}} x_{\mathbf{p},\mathbf{t}} = 1$, in which $x_{\mathbf{p},\mathbf{t}}$ is an element of X corresponding to model point $\mathbf{p}$ and target candidate point $\mathbf{t}$. For each model point, at each fixed scale, rotation and matching target location, the minimum objective function is achieved by minimizing the data term of the objective function in (2), i.e., the linear combination of all the matching costs for the model point. For model point $\mathbf{p}$, optimizing the objective $\sum_{\mathbf{t}} c(\mathbf{p}, \mathbf{t}) x_{\mathbf{p},\mathbf{t}}$ can therefore be achieved by adjusting the weight $x_{\mathbf{p},\mathbf{t}}$ under the constraint $\sum_{\mathbf{t}} \mathbf{t} \cdot x_{\mathbf{p},\mathbf{t}} = \mathbf{t}_{\mathbf{p}}$ and $\sum_{\mathbf{t}} x_{\mathbf{p},\mathbf{t}} = 1$. The optimum point must be a point on the lower convex hull of the matching cost surface $c(\mathbf{p}, \mathbf{t})$ over $\mathbf{t}$. Shown in Fig. 4, a lower convex hull is the tightest envelope that supports the 3D point cloud $(\mathbf{t}, c(\mathbf{p}, \mathbf{t}))$: At each location the cost achieves the lowest possible value by a suitable linear combination of the point clouds.

The vertices on the lower convex hulls are all we need to represent cost surfaces. For each model point, the candidate points corresponding to the lower convex hull vertices are effective target candidate points. The elements of X and X_l that do not correspond to lower convex hull vertices are zero, and only the vertex variables in X and X_l participate in the linear program pivoting. As shown before, for fixed scale s, rotation angle θ, and target location of each model point or equivalently fixed XT, we can choose X so that the lowest objective function achieves on the lower convex hull of each matching cost surface, i.e., nonzero elements in X must correspond to the vertices of a facet on each lower convex hull (the facet may degenerate into a single vertex or an edge). Therefore, all of the X variables that do not correspond to the lower convex hull vertices are zero;

otherwise, we can reduce the objective function by choosing X corresponding to a lower convex hull facet below these nonzero vertices. Note that as we assume XT is fixed, the smoothness term is fixed too and therefore can be ignored in the reasoning. As we assume arbitrary scale, rotation angle, and target location, our conclusion is general. Given X, we can always find X_l that are nonzero only for the same set of lower convex hull vertices as those for X to satisfy the constraints $sX = \sum_l s_l X_l$ and $\sum_{l=1}^{n_s} s_l X_l 1_{n_t} = s1_{n_m}$. This can be achieved by solving a simple linear system. For each model point, the linear program therefore approximates the original optimization problem by replacing the matching cost surface with its lower convex hull.

The above property means that we can select the effective target candidates for each model point and discard the rest so that the linear program can be solved efficiently. Since the shape of the lower convex hull is not directly related to the number of target points but only to their overall structure, the number of effective target candidates and the size of the linear program are greatly decoupled from the number of the target points.

Property 2. *If the local matching costs are scale dependent, we only need to keep those target points* $\mathbf{t}$*, scales* s*, and their associated feature matching costs* $c(\mathbf{p}, \mathbf{t}, s)$ *that correspond to the vertices of the lower convex hull of the point cloud* $\{(s\mathbf{t}, s, c_{\mathbf{p},\mathbf{t},s}) : \forall \mathbf{t}\}$.

When the cost surface is scale variant, we need to compute a 4D lower convex hull of the point cloud $(sx, sy, s, c(x_m, y_m, x, y, s))$, where (x_m, y_m) is a model point and (x, y) is a target candidate point. Imagine a 3D volume with coordinate (x, y, s), in which each point at (sx, sy, s) has a value $c(x_m, y_m, x, y, s)$. The 4D lower convex hull of the point cloud is a volume in which each point's value is the minimum of all the linear combinations of the 4D point costs (the values of the 4th coordinate). Similarly to 3D cases, only vertices at locations (x, y) and scales s on the lower convex hulls need to be used in the optimization. With this property, the size of the LP is roughly decoupled from the levels of the discrete scales.

Property 3. *If the local matching costs are both scale and rotation dependent, we only need to keep those target points* $\mathbf{t}$*, scales* s*, angles* θ*, and their associated feature matching costs* $c(\mathbf{p}, \mathbf{t}, s, \theta)$ *that correspond to the vertices of the lower convex hull of the point cloud* $\{(s\mathbf{t}, s, \theta, c(\mathbf{p}, \mathbf{t}, s, \theta)) : \forall \mathbf{t}\}$.

Comparing to solving the original intractable nonlinear and nonconvex optimization, the linear relaxation can be solved efficiently. Using the simplex method [21], its average complexity is roughly proportional to the logarithm of the number of target candidate points. It is also more efficient than the quadratic formulation based on pairwise model point assignments [33], which has the average complexity proportional to the square of the number of target points. By using the lower convex hull trick, our formulation makes the numbers of variables and constraints largely decoupled from the number of target candidates. The complexity of the proposed method is therefore nearly independent of the number of target points. Fig. 5 shows an example of how the number of lower convex hull vertices

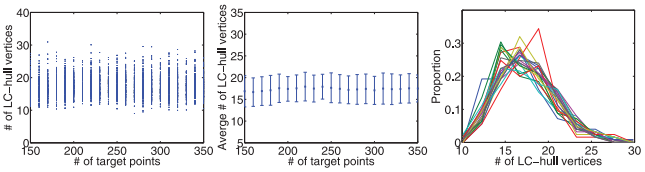


Fig. 5. The number of lower convex hull vertices relative to the number of target candidate points in random point cloud matching. The template contains 100 random points; the template is randomly scaled in $[0.5, 2]$, rotated in $[0, 2\pi]$, perturbed in 0-5 pixels pointwise and embedded in noise points to form the target image. Shape context is used as the matching feature. In each measurement, we compute the average number of lower convex hull vertices for all of the model points. We repeat 1,000 trials for each noise level setting. (a) The scatter plot. (b) The average number of lower convex hull vertices almost keeps constant as the number of target candidate points increases. (c) The histograms of the number of lower convex hull vertices have similar shapes as the number of target points scales.

corresponds to the number of target candidate points in random point cloud matching. The number of effective target points is largely decoupled from the number of target candidate points. In our experiments, the number of effective target points is about 20 for 200 target points in shape context matching; the number is around 50 for about 2,000 target points in SIFT feature point matching and there are around 60 effective target points for about 9,000 edge points in local image patch matching.

We solve the LP using a modified simplex method with GNU GLPK. Typically, using scale and rotation invariant matching costs, for matching 100 model points and thousands of target points, the LP converges in less than 1 second on a 2.8 GHz PC. If the matching cost is not scale and rotation invariant, the complexity of precomputing the local cost matching may dominate the complexity of the method because the local matching costs are needed to be computed in different rotations and scales. Parallel computing methods can be used to relieve the problem. The optimization can still be accelerated by using the lower convex hull trick to reduce the size of the LP.

2.7 Successive Refinement

The above solution provides a linear approximation to the original combinatorial problem. It may yield a result that is roughly correct but not accurate. A successive refinement method is used to solve this problem. Instead of solving the linear relaxation once, we iteratively solve a sequence of linear programs, each of which linearizes the combinatorial problem in smaller and smaller trust regions. For each model point, a trust region is a rectangular region in the target images. We refine the locations and the sizes of the trust regions based on the linear solution in each LP iteration. The trust regions in an iteration are rectangular areas centered on the previous target point estimations. As the trust regions become smaller, the result gradually improves. Even though the trust region for the scale may also be refined, for our applications we simply use its largest trust region in each iteration. The scale and rotation invariant matching algorithm with successive refinement is summarized as follows:

1. Compute the feature matching cost matrix C, C_l, or $C_{l,k}$ between model features and target features.

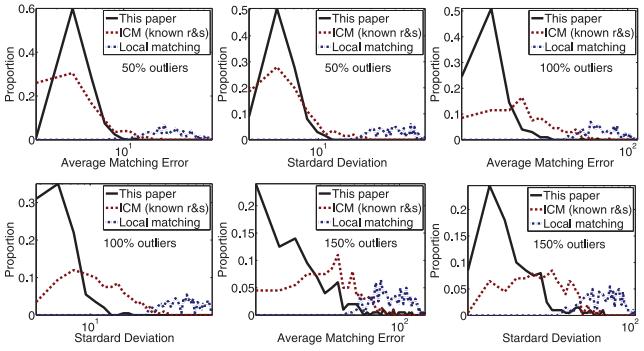


Fig. 6. Our matching error for *fish* is smaller than ICM and local matching in terms of both the mean and the standard deviation, for the three levels of clutter.

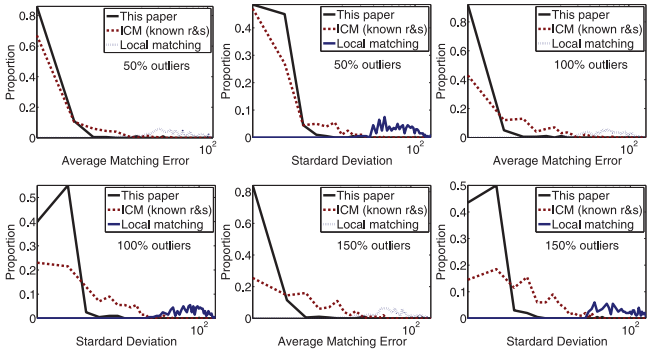


Fig. 7. Our matching error for *random point cloud* is smaller than ICM or local matching in terms of both the mean and the standard deviation, for the three levels of clutter.

2. Initialize trust region for each model point to be the entire target image.
3. Compute the lower convex hull vertices of matching costs for each model point within its trust region.
4. Solve four linear programs in (4), (5), or (6), one for each uv line.
5. Update the trust regions. If they are small enough, find the linear program that has the lowest matching cost ε and output the matching result; otherwise, go to 3.

We simplify the algorithm in the implementation. Instead of solving four linear programs in each iteration, we only refine the one that yields the lowest cost in the first iteration. The simplified algorithm is still accurate and robust.

3 Benchmarking Using Ground Truth Data

Synthetic point set matching is used to benchmark our algorithm. We use shape context as the feature. For each target point, we compute shape context at seven different scales ranging from 0.5 to 2 times of the template size, and at multiple angles by shifting the shape context along the angular axis. The feature matching cost is the minimal χ^2 distance between the model feature and all the scaled and rotated versions of the target feature (7). We first compare the performance of the proposed method with a few low complexity methods, including local matching and the greedy method ICM. We would like to confirm that the proposed method has a big advantage over these simple methods. For synthetic data, we give ICM the advantage of knowing the right scale s and rotation R, i.e., the same energy function is used, but s and R are fixed to the correct values. In local matching, the target point for each model point is simply selected to be the one with the smallest local matching cost.

Two synthetic point models are used: One is the fish in Fig. 18 and the other is a random point cloud. The local deformation is smooth for the fish and restricted to a random shift of 0 to 10 pixels for the random point cloud. In the experiment, the scale varies within $[0.5, 2]$ and the angle of rotation is in $[0°, 360°)$. We consider three clutter levels, where the number of random noise points is 50 percent, 100 percent, and 150 percent of the number of model points. The matching performance is quantified by the normalized histograms of the error means and standard deviations over all the trials for a specific noise and deformation setting. There are therefore six test cases total. For an ideal matching method, the normalized histograms for both the average error and the standard deviation should have a unit peak at zero and vanish everywhere else. In reality, the histograms that have high values in low error ranges and a short tail in high error ranges indicate good performance of a matching algorithm. Figs. 6 and 7 show the normalized histograms computed over 200 random trials for each test case. Our LP solution consistently gives smaller average errors than ICM with known scale and rotation. The standard deviation histograms also show that the proposed method has the most consistent performance in matching. The local matching results are always the worst. With increasing clutter, our method still has a high chance of finding the right scale, rotation, and point-to-point correspondences. The proposed method is also efficient. It has a similar complexity to these greedy search methods.

We further compare the proposed method with RANSAC. RANSAC prunes the matches by comparing the ratio of the best local match cost to the second best match cost with a threshold. If the ratio is smaller than the threshold, the model point and its best match target point are kept as a candidate matching pair. In verification, RANSAC randomly picks up three point pairs surviving in the local matching and generates a similarity transform from the template to the target. After globally transforming the template points, we find the nearest target point for each template point and compute the overall feature difference. We repeat this procedure for a large number of times and keep the best matching result. Since the RANSAC matching is not deformable, we test the overall success rate—the percentage of trials with average matching errors less than a threshold. The fish data and random point data are used in the testing. With proper parameter settings, RANSAC is quite reliable in the low to median noise level tests. In the high level noise tests, RANSAC breaks down. Fig. 8 shows the error histogram of the proposed method and RANSAC for the fish and random point test both with 200 percent noise points. Table 1 shows that the overall success rates for the proposed method and RANSAC. We set the threshold for a correct match to 50, which roughly complies with the visual inspection result. RANSAC uses a threshold 0.9 in preprunning and generates 5,000 random hypotheses. As shown in Fig. 8, the proposed method has better performance than

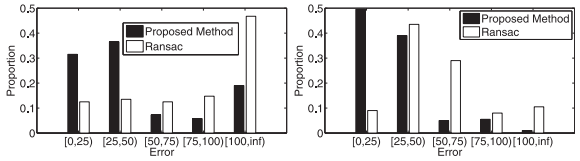


Fig. 8. Comparison with RANSAC Test I on fish (left) and random points (right).

TABLE 1
Comparison with RANSAC, Test I

	Fish	Random point
Proposed method	68%	88%
RANSAC	26%	52%

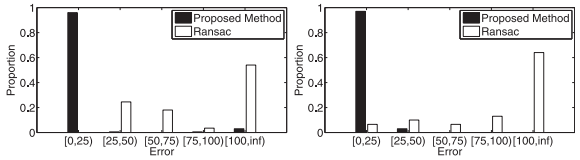


Fig. 9. Comparison with RANSAC Test II on fish (left) and random points (right).

RANSAC. RANSAC is also slower than the proposed method because it has to validate a large number of random hypotheses.

The failure of RANSAC is due to the heuristics for pruning the matching pairs: It always tries to find a nearest neighbor for a template feature. When the noise level increases, the nearest neighbor is less and less reliable; RANSAC has to sample a large number of candidate matches and becomes increasingly inefficient. When the noise level becomes so high that no triple of correct matching pairs can be found, RANSAC fails no matter how many validation trials it takes. The proposed method does not have this limitation because it uses all the candidates in matching. To justify the robustness of the proposed method against the corruption of the best matching candidates, we repeat the fish and random point matching experiments and purposely replace the best matching cost for each template point with a large value. We compare the result of the proposed method with RANSAC. In the fish and random point tests, we add 50 and 100 clutter points, respectively. The results are shown in Fig. 9 and Table 2. The proposed method is little affected by the disturbance and has much better results than RANSAC.

We further compare the proposed method with a recent spectral graph matching method [33] and the widely used iterative thin plate spline (TPS) method [20]. The spectral graph method [33] is shown to be one of the most robust methods for graph matching. TPS [20] has been successfully applied to shape matching. These two methods represent the state of art for image matching and are good candidates for matching performance comparison. To be consistent with our previous experiments, we use shape and rotation invariant shape context as local features and use the fish and random point data set in the test. The graph matching method uses 3rd-order tensor and 3-point local groups. Besides the shape context matching cost, the graph method also uses inner angle differences of the local point groups as a regularization term. We adjust the weight coefficients to

TABLE 2
Comparison with RANSAC, Test II

	Fish	Random point
Proposed method	96.5%	100%
RANSAC	24.5%	16.5%

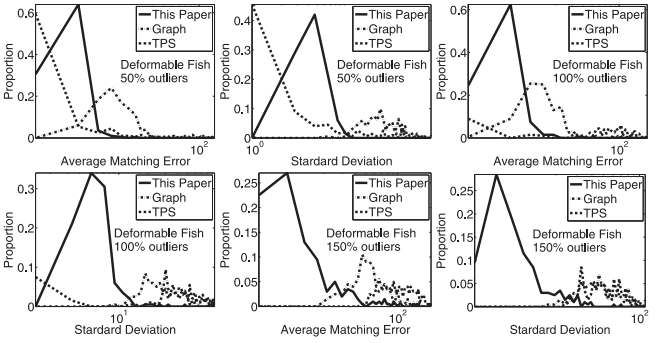


Fig. 10. Comparison with graph matching [33] and TPS using shape context [20] on the fish test data.

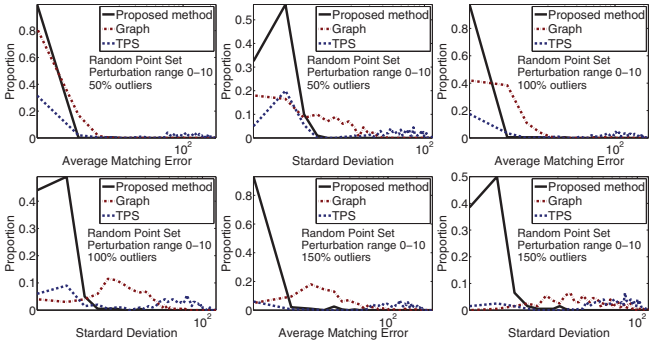


Fig. 11. Comparison with graph matching [33] and TPS using shape context [20] on the random point data.

balance the two terms so that the spectral graph method achieves the best performance. TPS [20] is an iterative matching method that alternatively finds the point correspondences and the global deformation. We use the demo codes for [33] and [20] in the experiment. The comparison is on the fish and random point data set on three different noise levels: 50 percent, 100 percent, and 150 percent. Apart from the noise points, each point of the random noise pattern is also randomly perturbed by 0-10 pixels to simulate local deformation. There are six test cases, for each of which we conduct 200 trials. The fish data set comparison result is shown in Fig. 10 and the random point comparison result is in Fig. 11. The average error histograms and standard deviation histograms show that the proposed method performs better in all the test cases and degrades more slowly as the noise level increases from 50 to 150 percent than the graph matching and TPS methods. The average errors in Table 3 further confirm the observation. Even though graph matching and TPS work well in the low level noise cases, they break down as clutter increases. The graph matching method deteriorates quickly when the noise level goes up to 150 percent and TPS breaks down when the noise exceeds 50 percent. In terms of the complexity, the proposed method is also many orders faster than graph matching or TPS, especially for matching a large set of target points.

TABLE 3
Average Errors in Matching

	Fish (50% clutter)	Fish (100% clutter)	Fish (150% clutter)	Point (50% clutter)	Point (100% clutter)	Point (150% clutter)
Proposed method	3.8026	7.3242	20.6566	1.9562	2.9529	4.3628
Graph matching [33]	13.5004	20.9156	64.7240	2.8631	9.7601	37.5275
TPS [20]	22.6079	89.4639	114.2155	83.0433	99.1534	130.8878

4 Applications of Invariant Matching

Our matching scheme is quite general and can be used in various applications, including action detection and image matching.

4.1 Action Detection

Action detection can be treated as a scale invariant matching problem [30]. We match the movement of an object in videos to models and determine whether a specific action occurs.

We use flow lines, the 3D trajectory of features points in the space-time volume, as local features. Flow lines can be easily computed using a greedy method. We use iterative conditional modes (ICM) to estimate the sparse point motion between adjacent frames. ICM is applied to edge pixels that surpass a Canny detector threshold; neighborhood relations are defined by the Delaunay triangulation of these edge pixels. The resultant sparse motion field is then interpolated across Delaunay cells to produce a dense motion field. The frame-by-frame motion fields produced by this procedure are then simply concatenated to form 3D flow lines in the space-time video volume. There are no constraints to prevent flow lines from intersecting. This method is designed to produce flow lines that are good enough on average to generate a coherent flow field since our robust matching procedure tolerates flow line errors. Although the method is greedy, one can see from Fig. 12 that the result is actually quite clean. More accurate but computationally more expensive methods such as [29] could be used to improve the flow line extraction.

To detect specific actions in videos, we match an exemplar clip to videos at each time instant using the proposed linear matching method. The template shape flow consists of a set of flow lines. These flow lines originate from randomly selected edge points on the object. Pairwise neighbor relationships between flow lines are given by the Delaunay triangulation of the flow line start points. Thus, the neighbor graph has cycles. All of the flow lines in the template have the same temporal extent, but vary in length due to motion.

The target search domain is a space-time video volume with the same temporal extent as the template action. The search is performed over a randomly selected subset of flow lines anchored by edge points in the first frame of the volume. The goal is to find a consistent assignment of flow lines in the template to flow lines in the target. Matched flow lines should be similar, and the spatial arrangement in the target should match that of the template. We use normalized flow lines in which each flow line is normalized by the longest x and y span. Simple euclidean distance is used in computing the feature distance. The local matching cost is roughly scale invariant.

An example of matching shape flow features is shown in Fig. 13. Figs. 13a and 13b show the top-down view (projected along the time axis) of a template shape flow for a person waving both arms. Note that the shape flow consists of a collection of flow lines (Fig. 13a) related by a neighborhood graph (Fig. 13b) given by the Delaunay triangulation of the flow line start points. Fig. 13c shows the top-down view of the flow lines in the target video volume, which are sampled from edge pixels in the first frame. Note that the target is a different person at a different scale, and that, individually, the target flow lines differ significantly from those in the template.

Our proposed linear matching method's success relies on the combination of matching the loopy relation graph and doing a robust global search. We can see the result of removing one or the other of these elements. Fig. 13e shows the result of our method if the loopy graph is replaced by a chain. In this case, without cycles, the matching may be done efficiently using DP. Fig. 13f shows the result of using the fully loopy graph, but using ICM instead of our proposed

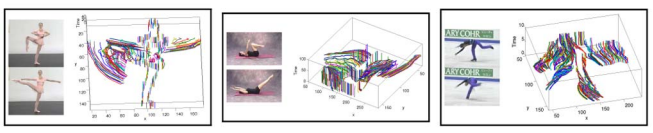


Fig. 12. Example shape flows and first/last frames for a variety of actions. Even though individual flow lines are noisy, the shape flow represents the holistic shape *and* movement of the object reliably. We show how to efficiently search for a template shape flow in a target video using linear invariant matching.

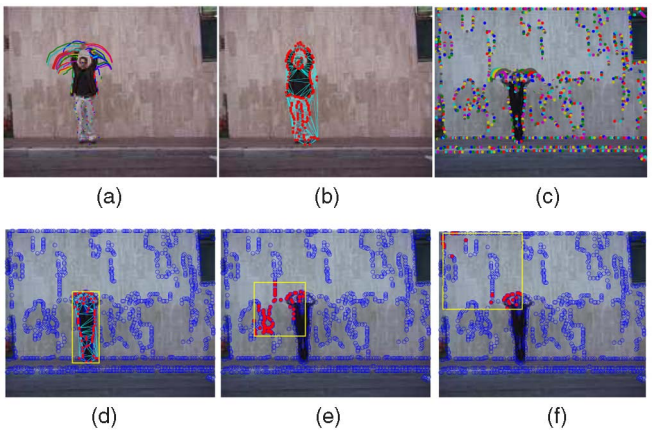


Fig. 13. Matching shape flows. A template shape flow (a) and its neighbor relation graph (b) are matched to a target video (c). Note that stationary points in (a) and (c) produce flow lines orthogonal to the page, which are depicted as dots. The matching result using the proposed method of this paper (d) is superior to the result achieved either by DP (e) or iterative conditional modes (f). In (d)-(f), the blue dots denote the start points of target candidate flow lines.

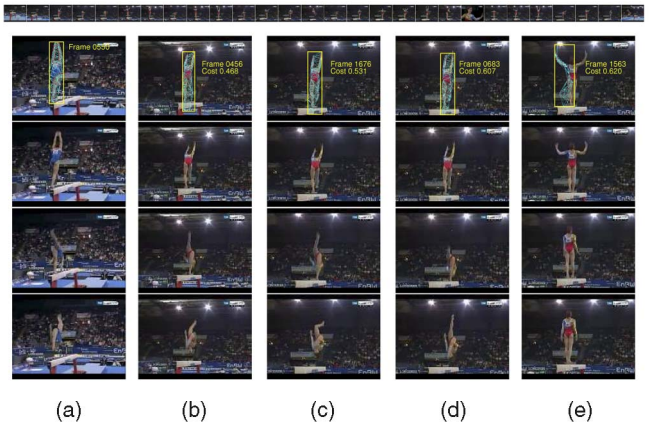


Fig. 14. Detection of a complex 15-frame action in the 2,091-frame *gymnastics* video. The action is the first half of a forward flip on the balance beam. The top row of images shows frames sampled uniformly in time from the video. Column (a) shows the action template along with three frames (beginning, middle, and end) from the action. Columns (b)-(e) show the top four matches. The sequence contains three true positives, which are the top three matches. In this example, which involves rapid and complex object motion as well as camera motion and background clutter, there are no errors. (a) Template. (b) First match. (c) Second match. (d) Third match. (e) Fourth match.

Fig. 15. The top 30 detections for a 8-frame *swing* template action in an 8000-frame video *golf*. There are 10 hits (with duplicates) of 14 true positives and 11 false positives, yielding 71 percent recall and 63 percent precision. This is a difficult video with few true positives compared to negatives, and much camera motion.

linear relaxation. The DP and ICM results are poor, despite the fact that both methods were given the advantage of having the template prescaled to the target scale.

After we find the matching of template shape flow with targets, we further decide whether the matched target is the same action. The energy function that we minimize to find an action in a video frame is effective at locating the best match within a frame. However, the energy values cannot be meaningfully compared across frames. There are two reasons for this. First, the flow line match cost in the linear program is totally scale invariant, which is too lenient for a cross-frame match score. And second, the energy will be artificially low when the template is matched against a partially similar action; for example, a template of a person waving one arm will match well to a target waving two arms, but should not be scored high. To address these issues, we formulate a more robust similarity measure between the template shape flow and the matched target shape flow. First, we normalize the flow lines within each shape flow by the mean flow line length, and translate each flow line start point to the origin. The distance between these two bundles of normalized and shifted flow lines is defined as the average minimum distance between individual flow lines across bundles, which is a symmetric measure. The distance between individual flow lines is again defined as the euclidean distance between the flow lines' spatial coordinate vectors.

We first present action detection results for single actions in two extended videos: Fig. 14 for the *gymnastics* video and Fig. 15 for the *golf* video. In all cases, the top matches are determined using the shape flow distance measure. These videos involve fast object motion, camera motion, and background clutter. In addition, the template and target are always of different people, which introduces scale variation, pose variation, and intraclass variation.

Fig. 14 shows match results for a complex 15-frame action in the 2,091-frame *gymnastics* video. After applying non-minima suppression in the time dimension to the per-frame match scores, the three true positives appear as the top three matches. Non-minima suppression is not necessary, but duplicate matches are pruned from the top list. Despite background clutter, constant camera motion, and significant intraclass variation, the proposed method is successful.

Fig. 15 shows results for an 8-frame *swing* action in the 8,000-frame *golf* video. The top 30 matches are shown over the entire sequence. There are 10 hits of 14 true positives with 11 false positives, yielding 71 percent recall and 63 percent precision (with duplicate hits removed). This sequence involves much camera motion, a variety of individuals as targets, and highly variable background clutter. In addition, there are many distractor frames in which there is no relevant object present.

We also report results for the action recognition data set of Blank et al. [28] in Fig. 16. The data set consists of 93 single action video clips for 10 actions performed by various subjects. We extract a single 15-frame shape flow template for each action by randomly choosing a 15-frame sequence from a randomly chosen clip. Each template is then matched against each frame in the set of test clips (excluding the template clip). The match cost for a clip is taken as the minimum match cost across frames in the clip. Fig. 16 shows that the top matching clips have the correct class, yielding high performance precision recall curves for 8 of 10 classes; the *FJump* and *FHop* classes are not distinguished well by our detector, but that is because they are visually extremely similar. The equal precision-recall point across the data set is 90 percent. Excluding the *FJump* and *FHop* categories, the equal PR point is at 95 percent. Fig. 17 shows the results of the proposed method for the more challenging KTH action data set, which contains 599 video clips and 6 actions. The sequence contains camera motions that degrade the result mildly. Using a single template, we achieve about 70 percent

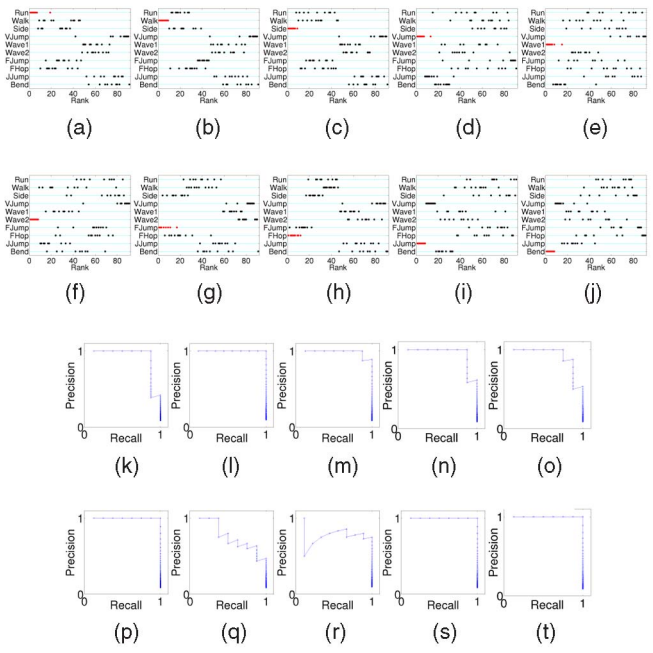


Fig. 16. Detecting actions in the video data set of Blank et al. [28]. The data set consists of 93 single action video clips for 10 actions. The 10 actions are running (*Run*), walking (*Walk*), side stepping (*Side*), jumping in place (*VJump*), waving one arm (*Wave1*), waving two arms (*Wave2*), forward jumping (*FJump*), forward hopping (*FHop*), jumping jack (*JJump*), and bending (*Bend*). We use a single action template for each action class. Graphs (a)-(j) show, for each action, all 92 clips (template clip excluded) sorted by match score when matched against that action template; the y-axis places each clip into its ground truth category. Most of the same category target clips (red dots) are ranked first, which is the goal. Graphs (k)-(t) show the corresponding precision recall (PR) curves for the 10 actions. We achieve high precision and recall for 8 of 10 actions; the *FJump* and *FHop* actions are extremely similar visually, and panels (g), (h) show that they get confused with each other. (a) Run. (b) Walk. (c) Side. (d) VJump. (e) Wave1. (f) Wave2. (g) FJump. (h) FHop. (i) JJump. (j) Bend. (k) Run. (l) Walk. (m) Side. (n) VJump. (o) Wave1. (p) Wave2. (q) FJump. (r) FHop. (s) JJump. (t) Bend.

detection rate. This result is encouraging considering that we only use one exemplar from each action class.

4.2 Object Matching

Another natural application of the proposed invariant matching scheme is object matching, in which we would like to find the correspondence of feature points on an object in an exemplar image to the target object in other images. The target object may have different rotations, scales, and may be deformed relative to the template object.

In object matching, we first compute the matching cost for each pair of template and target features. The proposed linear method is then used to optimize the matching. Both invariant and noninvariant matching costs can be used in the proposed method. Even when the feature matching cost c is invariant to s or R or both, it is not necessary that the features be scale and rotation invariant, e.g., SIFT [3]. For noninvariant features, we can compute an invariant matching cost as follows: For model point $\mathbf{p}$ and target point $\mathbf{t}$, we compute the features for $\mathbf{t}$ at multiple scales s and angles θ and use the minimal distance between the features of $\mathbf{t}$ and $\mathbf{p}$ as the matching cost $c(\mathbf{p}, \mathbf{t})$:

$$c(\mathbf{p}, \mathbf{t}) = \min_{s,\theta} \text{distance}(\text{feature}(\mathbf{p}),\ \text{feature}(\mathbf{t}; s, \theta)). \quad (7)$$

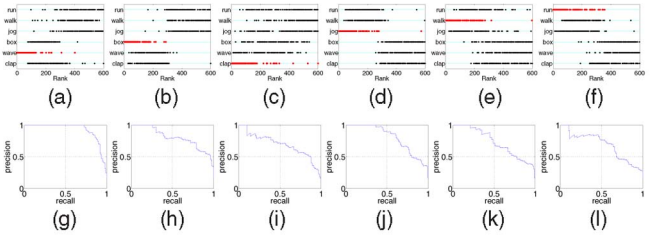


Fig. 17. Action detection for the KTH data set. First row: ranks of video clips (red dots indicate target clips). Second row: precision recall curves. (a) Wave. (b) Box. (c) Clap. (d) Jog. (e) Walk. (f) Run. (g) Wave. (h) Box. (i) Clap. (j) Jog. (k) Walk. (l) Run.

Fig. 18 shows an example of object matching using shape context and the proposed linear method. The scale and rotation invariant matching cost is computed as that used in ground truth point set matching. The model graph is generated by the Delaunay triangulation of the model points. The target object is a locally deformed, globally scaled, and rotated version of the template with 100 additional random noise points. In this example, ICM fails to find the right correspondences even with the correct scale and rotation (Fig. 18j); our algorithm gets roughly the right scale and orientation after the initial iteration (Fig. 18b). As we narrow down the trust region from 200×200 to 10×10 (Figs. 18c, 18d, 18e, 18f, 18g, 18h, and 18i), the match is progressively refined in scale, rotation, and correspondence. Fig. 19 shows another example in which object matching uses color images with SIFT features. The proposed method achieves near perfect result after a few iterations.

We applied our method to shape matching, in which we find similar shapes to an exemplar. Fig. 20 illustrates how our method can handle rotation, scale changes, and large deformations in matching edge points from a template to a target. We further test our method using the Brown shape data set [27], which contains 149 shapes. These shapes have large rotations, scale changes, and deformations. We use shape context as the feature, and the shape similarity is computed by linearly combining the shape context difference and the shape deformation. Shape deformation is defined as the ratio of feature point pairwise length changes to the model point pairwise lengths after scale normalization. We randomly select 50 percent of the edge points in the template and target images. An exemplar shape is

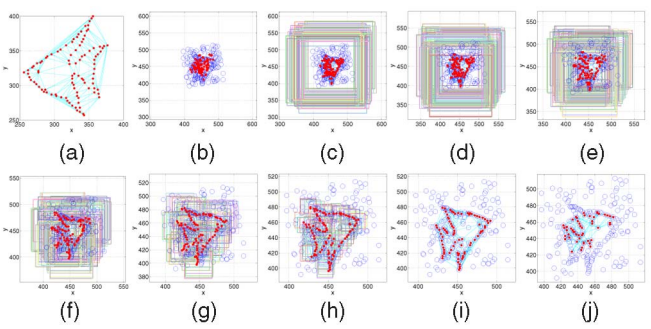


Fig. 18. An example of matching a scaled and rotated deformable shape with the proposed method. (a) A deformable fish shape template. (b) Our initial match when the trust region is the entire image. (c)-(i) Our results over iterations that shrink the trust regions. The algorithm converges to the nearly perfect match. (j) Match found by ICM with known scale and rotation.

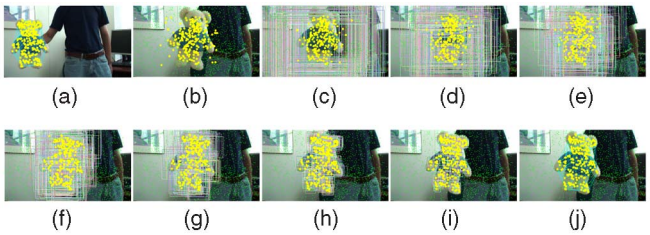


Fig. 19. Matching bear using SIFT. (a) Template mesh. (b) Initial matching. (c)-(i) Shrink trust regions to refine the matching. (j) The result mesh on the target.

Fig. 20. Shape matching sample results.

randomly chosen from each test category. Fig. 21 shows the short list for each enquiry and Fig. 22 shows the precision recall curves. Our method accurately detects similar shapes.

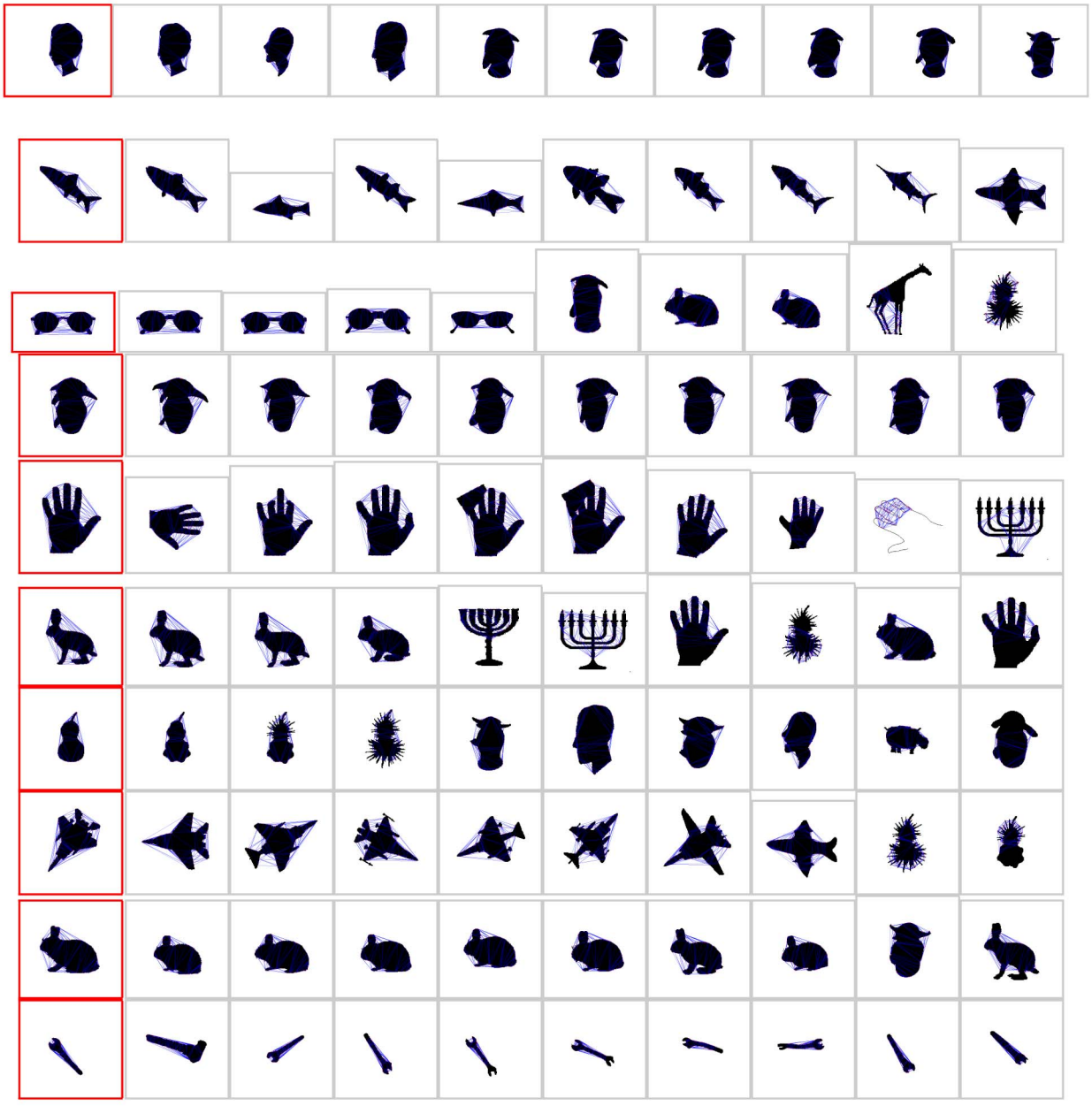


Fig. 21. Shape matching short lists for the Brown shape data set [27]. The objects inside red boxes are the query objects. The blue meshes illustrate the matching results.

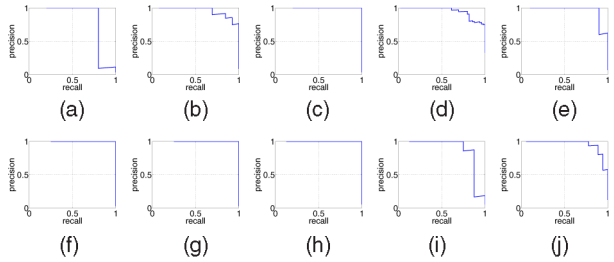


Fig. 22. The precision recall curves for shape matching on the Brown shape data set (149 shapes) [27]. (a) Face. (b) Fish. (c) Glass. (d) Greeble. (e) Hand. (f) Hare. (g) Pear. (h) Plane. (i) Rabbit. (j) Tool.

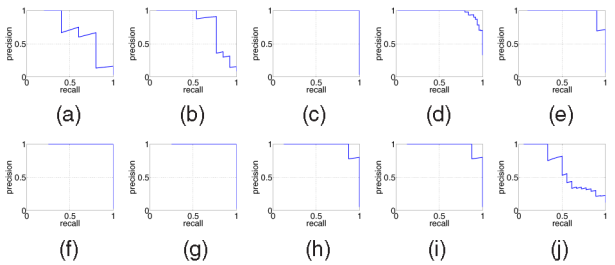


Fig. 23. The precision recall curves after adding 50 percent noise points in shape matching on the Brown shape data set [27]. (a) Face. (b) Fish. (c) Glass. (d) Greeble. (e) Hand. (f) Hare. (g) Pear. (h) Plane. (i) Rabbit. (j) Tool.

The equal PR point is above 95 percent on average. As shown in Fig. 23, after adding 50 percent noise points, the shape matching result only has mild degradation.

Fig. 24 shows matching results for the Caltech face data set that contains 410 face images of 26 people. We construct a template for each person and test how the proposed method can be used to match other faces of the same person. We use local color blocks as features. The test images are randomly rotated and scaled in 0.5-2. The success matching rate is 86 percent. The proposed method performs well in finding faces in clutter with large rotations and scale changes.

We further test our method on different videos: our own videos (*book, magazine, and bear*) demonstrating scaling, rotation, deformation, and occlusion, and YouTube videos (*butterfly, bee, and fish*) of animals in their natural habitats. Given a sample image for each video (Fig. 25 Row 1), we first label the object region, and then build a model graph with interest points and their neighboring connections through Delaunay triangulation. Such a process involves only selecting a region of interest; the feature points are randomly selected and the model graph is constructed automatically. We use SIFT points for all of the videos except the fish, for which small image patches on randomly selected edge points within the object region are used instead. Sample matching results for the testing sequences are shown in Fig. 26. These video sequences have a large range of scaling and rotation. Our single frame-based matching algorithm requires no initialization and can track a deformable object undergoing large and complex motion over long video sequences (Fig. 25, Row 3). The shapes of these long tracks are characteristic of the object's deformation and movement patterns, which are useful for activity recognition.

We test the performance of the proposed method on occlusion resistance with a hand in front of the book in the book sequence. The proposed method is found to be resistant to partial occlusions. Since an occluded model point matches everywhere with a high cost, its target point location is therefore mostly determined by spatial constraint from its neighbors. The energy minimization will favor the spatial constraint such that the overall spatial layout does not change drastically. The proposed method therefore can

Fig. 24. Matching faces in the Caltech face data set. The first row shows the sample results. The rest of the images are automatically cropped from the original images based on the matching result. There are 26 people and 410 images in the data set. The detection rate is 86 percent.

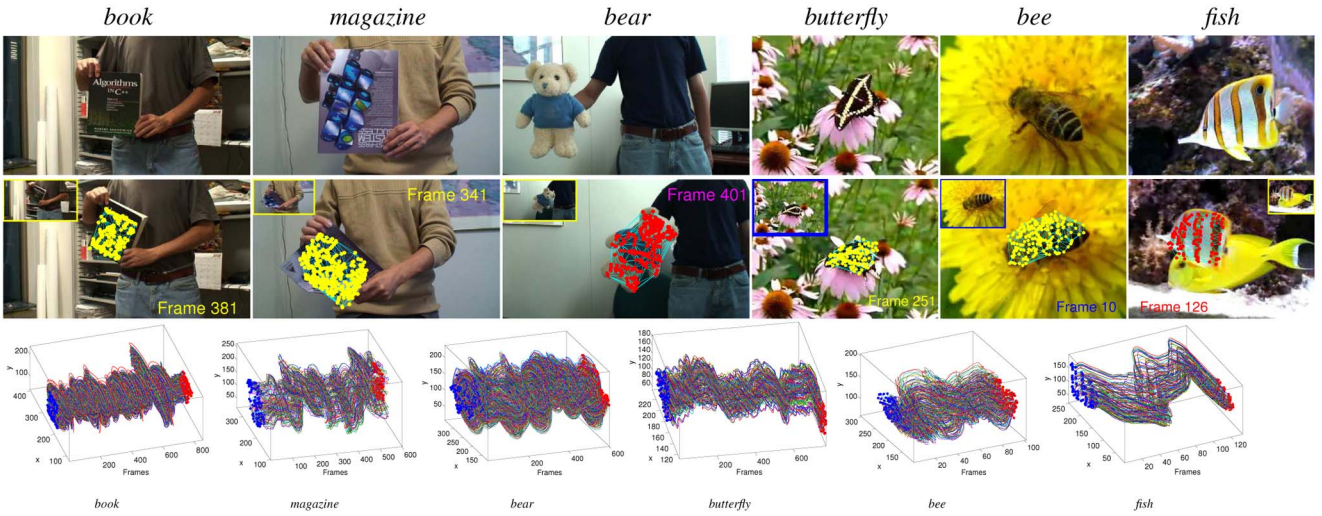


Fig. 25. Our LP is accurate and robust in matching objects in challenging real videos. **Row 1**: Images used to construct object templates from the Delaunay triangulation of detected interest points. **Row 2:** Sample results of the proposed method. **Row 3**: Point trajectories from the first (blue dots) to the last frame (red dots). The sequences involve scaling and rotation (*book* and *bear*), complex local warping (*magazine* and *butterfly*), and segments of smooth movement (*bee* and *fish*).

Fig. 26. Sample matching results. These objects all have large scaling and rotation. In particular, *book* could be occluded, *magazine* has large warping, *bear* is textureless, *butterfly* flaps wings, *bee* is striated and circling in depth, and *fish* has weakly distinguishable features.

video	*book*	*magazine*	*bear*	*butterfly*	*bee*	*fish*
#frames	856	601	601	771	101	131
#model	151	409	235	124	206	130
#target	2143	1724	1683	1405	1029	7316
time	1.6	11	2.2	1	2	0.9
accuracy	99%	97%	88%	95%	79%	95%

Fig. 27. Performance statistics of our LP method on real videos. The five rows give the total number of frames, the number of model points, the average number of target points per frame, the typical running time measured by the number of seconds that one LP iteration takes on a 2.8 GHz PC, and the accuracy measured by the detection rate over the entire video.

correctly hallucinate the mesh on the occluded part of the target object. Our deformable model can also deal with the significant warping of the magazine as shown on the second row of Fig. 26. It is also robust to weak local SIFT features and robustly matches the textureless furry toy bear.

The last three examples in Fig. 26 are YouTube videos of animals moving naturally in their habitats. The image quality is low due to heavy compression. In addition, we test matching performance on large deformations with wing flapping of the butterfly, on appearance with indistinguishable texture features with the bee and the fish, and on large viewpoint changes with the out-of-plane rotation of the bee.

The results in Fig. 27 show that our LP solution is accurate and robust in face of all these challenges. The running time is largely dependent on the number of model points, and insensitive to the number of target points. The complexity of the LP is roughly $O(n^p)$, where n is the number of model points and p is between two and three. For problems with about 100 model points, the running time for the LP takes about 1 second; when the number of model points is 400, the running time of the LP goes up to about 11 seconds. The detection rate is largely dependent on the distinctiveness of features that allows the objective function to tell model points apart from each other: It is higher than 95 percent for the book, magazine, butterfly, and fish, lower for the textureless bear, and lowest for the striated bee. Matching the tropical fish is a challenge because there are few distinctive SIFT features. We choose to detect edges instead to locate target points, and then use small image patches as their features. The feature matching cost is defined as the minimum color block euclidean distance at different rotations, which is roughly

Fig. 28. Comparison with RANSAC. The first two rows show the result of RANSAC; the last two rows show the result of the proposed method. RANSAC has 66 percent detection rate comparing with the 88 percent detection rate of the proposed method.

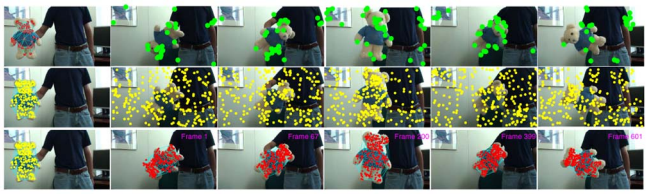


Fig. 29. Comparison of graph matching [33] and TPS matching [20] with the proposed method on the bear sequence. The first row shows the result of graph matching, the second row shows the result of TPS matching, and the third row shows the result of the proposed method. The first image in each row shows the template image with the template mesh overlaid on the top.

scale invariant for these stripes. Our method works well, even with such crude features, considering the large changes in scale, rotation, and color. This example also demonstrates that our method is versatile and robust with various features and matching cost functions.

We have compared the result of the proposed method with those of RANSAC, graph matching [33] and TPS matching [20] in the point data set experiments. We have shown that the proposed method outperforms these schemes in the ground truth tests. Our method also performs better in real image matching. Fig. 28 shows how the proposed method improves the result over RANSAC for the bear sequence. In this experiment, RANSAC exhaustively samples the local match pairs and therefore it is the best result that RANSAC can achieve. Due to weak features on the target object, RANSAC often loses the target. The results of spectral graph matching and TPS matching are shown in Fig. 29. For the spectral graph method, because of its high time-space complexity, we have to use a higher SIFT threshold to reduce the number of model points. The TPS method uses the same zero threshold as the proposed

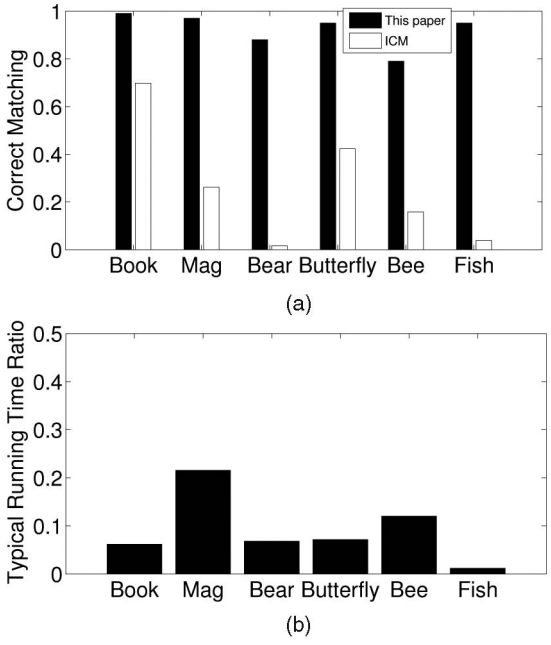


Fig. 30. Our LP method (black bar) has a much higher detection rate than ICM (white bar) yet requires a fraction of the running time. (a) Detection rate in a video. (b) Running time ratio of the proposed method to ICM.

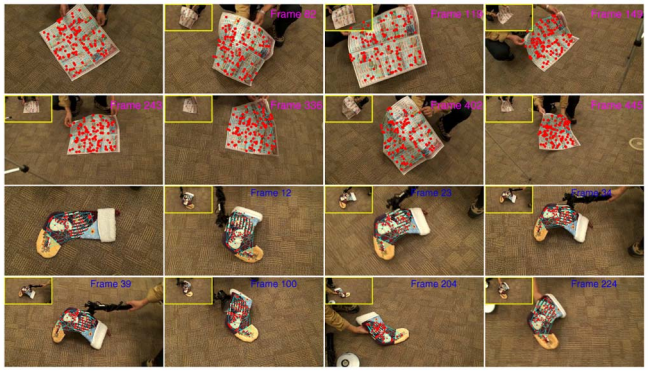


Fig. 31. Image matching results of the proposed method on objects with large deformations, rotations, and scale changes. The first images at row 1 and row 3 are the template images. The flyer images are sampled from 524 matched images; the sock images are sampled from 256 matched images.

method in detecting SIFT features. The spectral graph method and the TPS method are not able to match the target in the bear sequence due to the weak features and strong clutter. The proposed method works much better. It is also much faster than the spectral graph and TPS methods.

We further benchmark the efficiency of the proposed method. We use ICM, a simple greedy scheme, as a reference method. To apply ICM to these real videos, with no ground truth for scale and rotation available, we have to employ exhaustive search for s (seven scales with step 0.25 from 0.5 to 2) and R (every 30°). Since ICM is rather efficient, its performance is an indicator of the complexity of the matching problem. Our LP solution greatly outperforms ICM in terms of robustness (Fig. 30a) and efficiency (Fig. 30b) of matching all the sequences.

Fig. 31 shows our results on more challenging video sequences with large object deformation and in-plane/out-of-plane object rotations. Despite these challenges, our method reliably matches the target object.

5 Summary

Scale and rotation invariant object matching is generally an NP hard problem. We develop a linear solution with computational complexity insensitive to the number of target points, making it suitable for large scale matching problems.

Our results on both synthetic and real data demonstrate the accuracy, robustness, and efficiency of our method. It can be directly used to track an object with large shape deformations and geometrical transformations, find actions in clutter, and may be applied to object and activity recognition.

Acknowledgments

This work is supported in part by US National Science Foundation Grants 0644204 and 1018641.

References

[1] J.M. Gonzlez-Linares, N. Guil, and E.L. Zapata, "An Efficient 2D Deformable Objects Detection and Location Algorithm," *Pattern Recognition,* vol. 36, no. 11, pp. 2543-2556, 2003.

[2] C. Schmid and R. Mohr, "Local Grayvalue Invariants for Image Retrieval," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 5, pp. 530-535, May 1997.

[3] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int'l J. Computer Vision,* vol. 60, no. 2, pp. 91-110, 2004.

[4] P.F. Felzenszwalb and D.P. Huttenlocher, "Pictorial Structures for Object Recognition," *Int'l J. Computer Vision,* vol. 61, no. 1, pp. 55-79, 2005.

[5] S. Roy and I.J. Cox, "A Maximum-Flow Formulation of the N-Camera Stereo Correspondence Problem," *Proc. Int'l Conf. Computer Vision,* 1998.

[6] W.E.L. Grimson, "The Combinatorics of Object Recognition in Cluttered Environment Using Constrained Search," A.I. Memo No. 1019, Feb. 1988.

[7] J. Besag, "On the Statistical Analysis of Dirty Pictures," *J. Royal Statistical Soc.,* vol. B-48, no. 3, pp. 259-302, 1986.

[8] V. Kolmogorov and R. Zabih, "What Energy Functions Can Be Minimized via Graph Cuts?" *Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 2, pp. 147-159, Feb. 2004.

[9] V. Kolmogorov and R. Zabih, "Computing Visual Correspondence with Occlusions Using Graph Guts," *Proc. IEEE Int'l Conf. Computer Vision,* 2001.

[10] Y. Weiss and W.T. Freeman, "On the Optimality of Solutions of the Max-Product Belief Propagation Algorithm in Arbitrary Graphs," *IEEE Trans. Information Theory,* vol. 47, no. 2, p. 736, Feb. 2001.

[11] J. Sun, N.N. Zheng, and H.Y. Shum, "Stereo Matching Using Belief Propagation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 7, pp. 787-800, July 2003.

[12] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky, "Distributed Occlusion Reasoning for Tracking with Nonparametric Belief Propagation," *Proc. Conf. Neural Information Processing Systems,* 2004.

[13] D. Ramanan and C. Sminchisescu, "Training Deformable Models for Localization," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2006.

[14] A. Quattoni, M. Collins, and T. Darrell, "Conditional Random Fields for Object Recognition," *Proc. Conf. Neural Information Processing Systems,* 2004.

[15] H. Chui and A. Rangarajan, "A New Algorithm for Non-Rigid Point Matching," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2000.

[16] Z. Tu and A. Yuille, "Shape Matching and Recognition-Using Generative Models and Informative Features," *Proc. European Conf. Computer Vision,* 2004.

[17] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient Belief Propagation for Early Vision," *Int'l J. Computer Vision,* vol. 70, no. 1, pp. 41-54, 2006.

[18] A.C. Berg, T.L. Berg, and J. Malik, "Shape Matching and Object Recognition Using Low Distortion Correspondence," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2005.

[19] H. Jiang, M.S. Drew, and Z.N. Li, "Matching by Linear Programming and Successive Convexification," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 29, no. 6, pp. 959-975, June 2007.

[20] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 4, pp. 509-522, Apr. 2002.

[21] V. Chvátal, *Linear Programming.* W.H. Freeman and Co. 1983.

[22] B. Glocker, N. Komodakis, G. Tziritas, N. Navab, and N. Paragios, "Dense Image Registration through MRFs and Efficient Linear Programming" *Medical Image Analysis,* 2008.

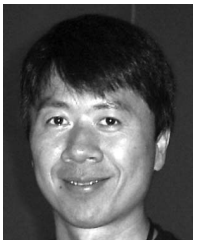
[23] C.J. Taylor and A. Bhusnurmath, "Solving Image Registration Problems Using Interior Point Methods," *Proc. European Conf. Computer Vision,* 2008.

[24] N. Komodakis and G. Tziritas, "Approximate Labeling via Graph-Cuts Based on Linear Programming," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 29, no. 8, pp. 1436-1453, Aug. 2007.

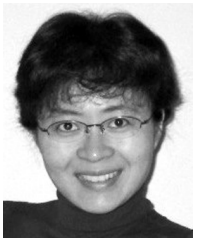
[25] C. Chekuri, S. Khanna, J. Naor, and L. Zosin, "Approximation Algorithms for the Metric Labeling Problem via a New Linear Programming Formulation," *Proc. 12th Ann. ACM-SIAM Symp. Discrete Algorithms,* pp. 109-118, 2001.

[26] A. Shekhovtsov, I. Kovtun, and V. Hlavac, "Efficient MRF Deformation Model for Non-Rigid Image Matching," *Computer Vision and Image Understanding Archive,* vol. 112, no. 1, pp. 91-99, Oct. 2008.

[27] D. Sharvit, J. Chan, H. Tek, and B.B. Kimia, "Symmetry-Based Indexing of Image Databases," *J. Visual Comm. and Image Representation,* vol. 9, no. 4, pp. 366-380, 1998.
[28] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *Proc. IEEE Int'l Conf. Computer Vision,* 2005.
[29] D. Kenwright and G. Mallinson, "A 3-D Streamline Tracking Algorithm Using Dual Stream Functions," *Proc. Conf. Visualization,* pp. 62-68, 1992.
[30] H. Jiang and D.R. Martin, "Finding Actions Using Shape Flows," *Proc. European Conf. Computer Vision,* 2008.
[31] H. Jiang and S.X. Yu, "Linear Solution to Scale and Rotation Invariant Object Matching," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2009.
[32] J. Maciel and J.P. Costeira, "A Global Solution to Sparse Correspondence Problems," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 2, pp. 187-199, Feb. 2003.
[33] O. Duchenne, F. Bach, I. Kweon, and J. Ponce, "Tensor-Based Algorithm for High-Order Graph Matching," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2009.
[34] P.H.S. Torr and D.W. Murray, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix," *Int'l J. Computer Vision,* vol. 24, no.3, pp. 271-300, 1997.
[35] M.A. Fischler and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM,* vol. 24, no. 5, pp. 381-395. 1981.
[36] R.O. Duda and P.E. Hart, "Use of the Hough Transform to Detect Lines and Curves in Pictures," *Comm. ACM,* vol. 15, no. 1, pp. 11-15, 1972.
[37] P.H.S. Torr, "Solving Markov Random Fields Using Semi Definite Programming," *Proc. Ninth Int'l Workshop Artificial Intelligence and Statistics,* 2003.
[38] C. Schellewald and C. Schnrr, "Probabilistic Subgraph Matching Based on Convex Relaxation," *Proc. Int'l Workshop Energy Minimization Methods in Computer Vision and Pattern Recognition,* pp. 171-186, 2005.
[39] L. Torresani, V. Kolmogorov, and C. Rother, "Feature Correspondence via Graph Matching: Models and Global Optimization Export," *Proc. European Conf. Computer Vision,* 2008.
[40] N. Komodakis and N. Paragios, "Beyond Loose LP-Relaxations: Optimizing MRFs by Repairing Cycles," *Proc. European Conf. Computer Vision,* 2008.
[41] J. Duchi, D. Tarlow, G. Elidan, and D. Koller, "Using Combinatorial Optimization within Max-Product Belief Propagation," *Proc. Conf. Neural Information Processing Systems,* 2006.
[42] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge Univ. Press.



Hao Jiang received the BSc and MSc degrees in electronic engineering from Harbin Engineering University in 1993 and 1995, respectively. He received the PhD degree in computer science from Simon Fraser University in 2006. He was a postdoctoral research fellow at the University of British Columbia from 2006 to 2007. In 2007, he joined the Department of Computer Science at Boston College as an assistant professor. His research interests include object matching, human pose detection, action recognition, object tracking, and computer graphics. He is a member of the IEEE.



Stella X. Yu received the PhD degree in 2003 from the School of Computer Science at Carnegie Mellon University, where she studied robotics at the Robotics Institute and vision science at the Center for the Neural Basis of Cognition. She continued her computer vision research as a postdoctoral researcher at the Computer Science Department at the University of California Berkeley. Since she joined the faculty of Boston College on a Clair Booth Luce Professorship in 2005, she has been developing an interdisciplinary curriculum and research agenda on Art and Vision, for which she received the US National Science Foundation (NSF) CAREER award in 2007. Her research interests include spectral graph theory, perceptual organization, visual attention, image segmentation, brightness modeling, scene classification, object matching, and nonphotorealistic rendering. She is a member of the IEEE and the IEEE Computer Society.

David R. Martin received the BSE degree in computer science from Princeton University in 1996 and the PhD degree in computer vision from the University of California Berkeley in 2002. He was an assistant professor in the Computer Science Department at Boston College from 2003 to 2009. Since 2009, he has worked in the 3D Vision Group on the Streetview project at Google in Mountain View, California.