Scale and Rotation Invariant Matching Using Linearly Augmented Trees

Hao Jiang, Tai-Peng Tian, and Stan Sclaroff, *Senior Member, IEEE*

Abstract—We propose a novel linearly augmented tree method for efficient scale and rotation invariant object matching. The proposed method enforces pairwise matching consistency defined on trees, and high-order constraints on all the sites of a template. The pairwise constraints admit arbitrary metrics while the high-order constraints use L1 norms and therefore can be linearized. Such a linearly augmented tree formulation introduces hyperedges and loops into the basic tree structure. But, different from a general loopy graph, its special structure allows us to relax and decompose the optimization into a sequence of tree matching problems that are efficiently solvable by dynamic programming. The proposed method also works on continuous scale and rotation parameters; we can match with a scale up to any large value with the same efficiency. Our experiments on ground truth data and a variety of real images and videos show that the proposed method is efficient, accurate and reliable.

Index Terms—Object matching, scale and rotation invariance, high-order model, linearly augmented tree, linear optimization, decomposition method

1 Introduction

Matching objects in cluttered images is a challenging task because the target object may appear rotated, scaled and locally deformed. To handle shape variation, object matching is naturally formulated as a graph matching problem, in which the object is divided into parts represented by graph nodes and coupling between parts is represented by graph edges. The task of matching is to assign a target candidate to each graph node so that the assignment has low cost and the matching is consistent with the constraints defined on the graph edges. Graph matching is NP-hard in general. The loops and high-order coupling among graph nodes exacerbate graph assignment in object matching.

We propose a novel formulation for scale and rotation invariant object matching. In our model, the object parts follow basic tree relations and we also introduce global constraints that couple all the tree nodes. These global constraints can be linearized and we call this class of constraints *linearly augmented tree* (LAT) constraints. We solve the object matching problem with the LAT constraints by decomposing it into a sequence of tree matching problems, each of which can be efficiently solved by dynamic programming (DP).

Object matching has been intensively studied. A large class of graph matching techniques are based on discrete energy minimization. If the energy function is submodular then it can be efficiently minimized using max-flow algorithms [18], [15]. Alternatively, if the underlying graph is a tree then dynamic programming can be used [14]. The inference on the tree structure is efficient; however, tree structure models are not sufficient for many real world problems. Scale and rotation invariant matching require a non-tree model since additional constraints are required to enforce all the model points to follow roughly the same global transformation. Matching with loopy graph structures is NP-hard in general. Different approximation methods have been proposed. Popular techniques include loopy Belief Propagation [19], convergent Tree Reweighted Message Passing [27], integer quadratic programming [21], interior point method [25], primal-dual techniques [26] and dual decomposition [35], [34]. These methods have been successfully applied in tackling different vision problems such as image matching and segmentation.

Other optimization methods such as convex-concave programming [1], concave programming [5], eigendecomposition [2], and linear programming [3] have also been studied to tackle graph matching problems. These graph matching methods try to find a permutation matrix that quantifies the matching from a template graph to a target graph. Spectral graph methods [29], [7], [8] use quadratic programming to find the matching by solving eigenvalue problems. Other methods for quadratic programming in graph matching employ successive iterative projection [10], graduated assignment [9], integer projected fixed point [11], random walks [12] and game theory [39]. These previous methods handle only discrete variables, e.g. binary assignment variables, even though continuous procedures are often used during optimization. It is difficult to generalize these minimization techniques to handle a mixture of discrete assignment and continuous variables in a matching problem that has LAT constraints, e.g., the scale and rotation parameters are continuous while the point assignment is discrete. A typical workaround is to quantize these

- *H. Jiang is with the Computer Science Department, Boston College, Chestnut Hill, MA.*
- *T.-P. Tian is with the GE Global Research, Niskayuna, NY.*
- *S. Sclaroff is with the Department of Computer Science, Boston University, Boston, MA.*

Manuscript received 11 Aug. 2014; revised 15 Jan. 2015; accepted 18 Feb. 2015. Date of publication 3 Mar. 2015; date of current version 6 Nov. 2015.
Recommended for acceptance by H. Ishikawa.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2015.2409880

continuous variables and then apply the matching algorithm in each of the quantized discrete settings. However, such a simple solution is not ideal. The quantization approach generates many matching cases and is very slow. Quantization also introduces errors and affects the result.

Our optimization algorithm avoids such an ad-hoc quantization step by using a decomposition method that splits the relaxed problem into a master and slave optimization. The master problem optimizes over the set of continuous variables, and the slave problem performs efficient combinatorial optimization over the discrete variables in order to generate proposals for the master problem. Furthermore, the previous cited works only model low-order constraints (typically up to two or three) and in contrast, we use the LAT to model high-order constraints that couple all the model points in the matching.

Other matching techniques do not use an explicit graph template. The Hough Transform [16] is a robust and efficient voting method. However, it needs a careful quantization of the parameter space. Soft assign [20] alternates point matching and global transformation estimation. RANSAC [33] randomly generates and verifies a large set of possible matches. These methods do not need to quantize the global transformation parameters, but their performance deteriorates rapidly when clutter increases and features weaken. In our experiments, we show that matching using LAT constraints is reliable even when the scene is highly cluttered and the features used for matching are weak.

Two closely related works are linear matching [28] and the local affine [36], [30] method. These methods have different drawbacks. In [28], the pairwise constraints must use the L1 norm, and the scaling range must be known beforehand so that the scale parameter can be quantized. Our proposed method uses a different strategy to achieve an efficient solution: we use a tree decomposition and dynamic programming in this paper, whereas in [28] we use the lower convex hull approximation trick and discard ineffective target points. The lower convex hull approximation is not preferable if the features are very weak. The lower convex hulls of weak features are flat; they do not have enough structure to guide the search to the global optimum. The method in this paper does not need to approximate the target matching surfaces with convex hulls; as a result, it is more robust when handling weak features and local matching ambiguity. Moreover, the method in [36], [30] tends to match small structures when features are weak, whereas our proposed method eliminates such a bias.

In summary, the contribution of this paper is threefold:

- *Novel linear augmented tree (LAT) model*. We propose a new graph model to tackle object matching problems in computer vision. Such a model allows arbitrary metrics for the pairwise costs on trees and it also allows powerful high-order constraints that couple all the nodes. The LAT model is as powerful as the more complex non-tree high-order models and, at the same time, the inference on the model is efficient.
- *New formulation for rotation and scale invariance*. We propose a new linear formulation to tackle rotation and scale invariant matching. We show that the formulation is a special case of the LAT inference problem. Our method finds point matching and global rotation and scale simultaneously. It does not need to quantize the scale and rotation angle because they are directly handled in the continuous domain. The proposed method also allows virtually unbounded scaling so that users do not have to guess the scale range.
- *Efficient matching algorithm*. Our algorithm efficiently solves the matching with LAT constraints by relaxing the problem and decomposing it into a sequence of efficient dynamic programming problems. Furthermore, the relaxed problem can be solved optimally.

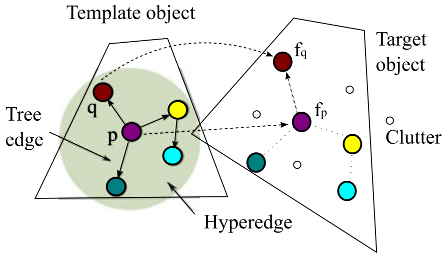


Fig. 1. Matching a template to a target object using linearly augmented tree (LAT) model. Our method allows arbitrary pairwise constraints defined on the basic tree edges and linear high-order constraints that couple all the model nodes.

2 Scale and Rotation Invariant Matching

We formulate scale and rotation invariant matching using linearly augmented tree constraints (Fig. 1). Given a set of template points $\mathcal{I}$ and target candidate points $\mathcal{J}$, the matching problem is formulated to search for three items, namely the mapping from model parts to the target candidates $f : \mathcal{I} \rightarrow \mathcal{J}$, rotation angle θ_0 and scale s_0 to minimize the objective function

$$\mathbf{c}(f, \theta_0, s_0) = \mathbf{c}_u(f) + \mathbf{c}_t(f, \theta_0, s_0) + \mathbf{c}_g(f, \theta_0, s_0). \tag{1}$$

The unary cost term

$$\mathbf{c}_u(f) = \lambda \sum_{i \in \mathcal{I}} c(i, f_i) \tag{2}$$

is the sum of matching costs $c(i, f_i)$ between each model point i and its target point f_i, where λ is a weight coefficient. The scale and rotation term,

$$\begin{aligned} \mathbf{c}_t(f, \theta_0, s_0) = \mu & \sum_{(p,q) \in \mathcal{N}} d(\theta((p,q),(f_p, f_q)), \theta_0) \\ & + \gamma \sum_{(p,q) \in \mathcal{N}} |s((p,q),(f_p, f_q)) - s_0|, \end{aligned} \tag{3}$$

encourages pairs of model points to have similar rotation angle θ_0 and scale factor s_0 in the matching. $\mathcal{N}$, the set of neighboring model points, corresponds to the edges of a tree. $\theta((p,q),(f_p,f_q))$ is the rotation angle from vector $\overrightarrow{pq}$ to $\overrightarrow{f_p f_q}$, and $s((p,q),(f_p,f_q))$ is the scaling factor between the two vectors. Fig. 1 illustrates the matching of a pair of model points. In Eq. (3), $d(.)$ computes the difference of two angles, and coefficients μ and γ control the weight. The optional term

$$\mathbf{c}_g(f, \theta_0, s_0) = \phi g(s_0, \theta_0, h(1), \ldots, h(n), h(f_1), \ldots, h(f_n)), \tag{4}$$

introduces an extra constraint across all the model points ($|\mathcal{I}| = n$), and $h(.)$ is a function that maps model points and target points to some quantities, e.g., the coordinates. We require that $g(.)$ in Eq. (4) contains only the L1 norm term with positive coefficients and linear operations on quantities of the model and target points. ϕ is a coefficient. This optional term gives the formulation flexibility to adapt to problems that require more complex constraints. As shown later, the scale-rotation term $\mathbf{c}_t$ and the term $\mathbf{c}_g$ can be linearized and form hyperedges on the basic tree nodes. Thus, the formulation follows an LAT model.

Even though the basic structure of an LAT model is a tree, the linear high-order constraints make the optimization difficult to solve. A naïve discretization method is infeasible if the scale upper bound is unknown; quantizing rotation angles and scales would result in too many discrete cases. Instead, we propose to encode the problem as a mixed integer linear program and show how to exploit its special LAT structure to design an efficient algorithm.

2.1 Linearization

We now describe how to encode the minimization of $\mathbf{c}(f, \theta_0, s_0)$ (Eq. (1)) as a mixed integer linear program. Assume that there are n model points and m target points. Let $[[\pi]] = 1$ if the predicate π holds and 0 otherwise. We introduce an $n \times m$ matrix X and $m \times m$ matrix $Y_{p,q}$ whose elements

$$x_{i,j} = [[f_i = j]] \qquad \text{and} \qquad y^{p,q}_{i,j} = [[f_p = i \wedge f_q = j]].$$

The matrix X indicates the matching of model points to target points, and the matrix $Y_{p,q}$ indicates the matching of a model point pair $(p, q) \in \mathcal{N}$ to target point pairs. Note that for each Y matrix i, j are in fact the element indexes in the matrix, p and q are fixed. We enforce X to be an assignment matrix with the unity constraint $X 1_m = 1_n$, where 1_m is an all one element vector of length m. The X and Y matrices are related by

$$X^T e_p = Y_{p,q} 1_m, \quad \text{and} \quad X^T e_q = Y^T_{p,q} 1_m,$$

where the n-vector $e_p = [0, 0, \dots, 1, 0, \dots, 0]^T$ has a single unity element at p.

The unary cost term defined in Eq. (2) can be represented as $\mathrm{tr}(C^T X)$ where $C = [c(i, j)]$ is the matching cost matrix whose element $c(i, j)$ is the matching cost from model point i to target point j.

The rotation term. For a model point pair $(p, q) \in \mathcal{N}$, we assume p matches target point i and q matches j. Let the rotation angle from vector $\overrightarrow{pq}$ to $\overrightarrow{ij}$ be $\theta^{p,q}_{i,j}$ and the $m \times m$ rotation angle matrix $\Theta_{p,q} = [\theta^{p,q}_{i,j}]$. If the target vector $\overrightarrow{ij}$ degenerates to a single point then $\theta^{p,q}_{i,j}$ is assigned a random number in $[0, 2\pi]$. The rotation angle for the model point pair (p, q) can be represented as $\mathrm{tr}(Y^T_{p,q}\Theta_{p,q})$. We require that all the model point pairs share similar rotation in the matching so that the object's spatial structure is maintained. To this end, we may minimize $\sum_{(p,q)\in\mathcal{N}} |\mathrm{tr}(Y^T_{p,q}\Theta_{p,q}) - \theta_0|$, where θ_0 is the overall (unknown) rotation angle, but this method does not work near the boundary between angle 0 and 2π. To avoid the difficulty, we split the rotation term into cosine and sine terms:

$$\sum_{(p,q)\in\mathcal{N}} \left\{ \left|\mathrm{tr}\left(Y^T_{p,q}\cos\left(\Theta_{p,q}\right)\right) - u_0\right| + \left|\mathrm{tr}\left(Y^T_{p,q}\sin\left(\Theta_{p,q}\right)\right) - v_0\right| \right\},$$

where u_0 and v_0 correspond to the cosine and sine of the unknown rotation angle θ_0, and $\cos(.)$ and $\sin(.)$ apply to each element of matrix $\Theta_{p,q}$. The absolute value terms are converted into linear functions by using a standard auxiliary variable trick [24]. Essentially, by introducing two non-negative auxiliary variables y^+ and y^-, $\min|x|$ is equivalent to $\min(y^+ + y^-)$, s.t. $x = y^+ - y^-$. It is easy to verify that either y^+ or y^- has to be zero and therefore $|x| = y^+ + y^-$ when the optimum is achieved.

The scaling term. The spatial consistency constraint further enforces that the line segments between neighboring model points should scale uniformly. Similar to the rotation matrix $\Theta_{p,q}$, we define an $m \times m$ scaling matrix $S_{p,q}$ for each pair $(p, q) \in \mathcal{N}$. The scaling for model point pair (p, q) is therefore $\mathrm{tr}(Y^T_{p,q}S_{p,q})$. To enforce the scaling consistency, we minimize

$$\sum_{(p,q)\in\mathcal{N}} \left|\mathrm{tr}\left(Y^T_{p,q}S_{p,q}\right) - s_0\right|,$$

where s_0 is the global scaling factor. We can linearize this term with auxiliary variable tricks similar to the rotation term.

Other optional terms. Apart from the above terms, we can also introduce the optional term $\mathbf{c}_g$ in Eq. (1), which is composed of L1 norms and linear functions of the quantities attached to model and target points. In our formulation, $\mathbf{c}_g$ may have θ_0 and s_0 as parameters. For instance, to match unreliable regions, we can globally constrain the overall target area to be similar to the template area multiplied by a scaling factor. If the optional term follows the aforementioned constraints, it can be linearized: each $|v|$ term in g becomes a summation of two non-negative auxiliary variables in the objective and their difference is set to equal v in the constraints.

We now obtain a mixed integer linear formulation of the nonlinear optimization in Eq. (1):

$$\max\left\{ -\lambda \mathrm{tr}(C^T X) - \sum_{(p,q)\in\mathcal{N}} \left[\mu\left(u^+_{p,q} + u^-_{p,q} + v^+_{p,q} + v^-_{p,q}\right) + \gamma\left(s^+_{p,q} + s^-_{p,q}\right)\right] - \phi g_o(w) \right\}, \tag{5}$$

subject to:

$$\boxed{\begin{gathered} \mathrm{tr}(Y^T_{p,q}\cos(\Theta_{p,q})) - u_0 - u^+_{p,q} + u^-_{p,q} = 0, \\ \mathrm{tr}(Y^T_{p,q}\sin(\Theta_{p,q})) - v_0 - v^+_{p,q} + v^-_{p,q} = 0, \\ \mathrm{tr}(Y^T_{p,q}S_{p,q}) - s_0 - s^+_{p,q} + s^-_{p,q} = 0, \\ g_c(X, w) = 0, \end{gathered}}$$

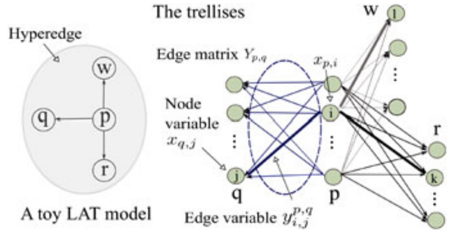


Fig. 2. LAT model and trellises. Thick lines indicate the paths.

$$X^T e_p = Y_{p,q} 1_m,\ X^T e_q = Y^T_{p,q} 1_m, \quad X 1_m = 1_n,$$
$$0 \le u^+_{p,q}, u^-_{p,q}, v^+_{p,q}, v^-_{p,q}, s^+_{p,q}, s^-_{p,q}, w \le M,$$
$$a \le u_0 \le b, c \le v_0 \le d, \epsilon \le s_0 \le L,\ u_0 \pm v_0 = \pm 1,$$

where X and Y are binary matrices. The linear function g_o is induced by the optional constraints; it is a linear function of X and non-negative auxiliary variables w. In the objective, the X terms have been absorbed into the first cost term and we denote g_o as a function of w. The constraint function g_c corresponds to all the L1 norm terms in the optional term.

In Eq. (5), the original minimization is changed to maximization of the negative. An extra constraint $|u_0| + |v_0| = 1$ is included to approximate the orthonormal constraint $u_0^2 + v_0^2 = 1$. The bounds $[a, b]$ and $[c, d]$ are determined by the quadrant of the approximation line. For instance, if $u_0 + v_0 = 1$, we have $a = 0, b = 1$ and $c = 0, d = 1$. We find the optimum among four quadrants. This constraint is optional; when included it improves the quality of the relaxation. We also include an upper bound M for the auxiliary variables; M is a large number to avoid the unbounded solution when the program is decomposed. The scale is upper bounded by L and lower bounded by a small number ϵ. In this paper, $M = L = 1{,}000$, and $\epsilon = 0.001$. Another small change in the optimization is the maximization of the negative of the original objective to achieve the minimization. This does not change the optimum; it is for the clarity of discussion using the *shadow price* concept: we maximize the profit given some resources, where the resources are the righthand side values (the constant part) of the constraints and they can be priced out with shadow prices.

It helps to visualize the mixed integer linear program using coupled trellises as illustrated in Fig. 2. By expanding the augmented tree nodes, we obtain a set of coupled trellises. Each trellis node corresponds to an X variable, and the edges between the candidate nodes of two neighboring model points correspond to a Y matrix. The optimization can thus be treated as searching for the optimal "paths" starting from a tree root node candidate and ending at a candidate of each tree leaf node. If the paths pass a node, the corresponding X variable is 1 and otherwise 0. If the paths pass an edge, the corresponding Y variable is 1 and otherwise 0. The cost of the feasible paths is the summation of the node costs, the scale-rotation cost and other optional costs induced by g. Due to the constraints that couple the tree branches, searching for the optimal paths in the trellises is a hard problem.

The optimization in Eq. (5) can be relaxed into linear programs and solved by the simplex method. However, when the target point number approaches thousands or millions, directly solving the large scale optimization becomes infeasible. Fortunately, with the LAT constraints, it can be decomposed into a sequence of efficient dynamic programming problems.

2.2 Decomposition into Dynamic Programming

It is the scale, rotation and the g constraints, the boxed constraints in Eq. (5), that complicate the optimization. Without the "complex" constraints, the problem turns into an optimization on a tree. The complex constraints introduce links (hyperedges) among all the tree nodes. If we find feasible solutions on the tree, we may use their linear combinations to satisfy the complex constraints and to optimize the objective based on Dantzig-Wolfe decomposition [40]. However, a naïve decomposition of the optimization into a sequence of linear programs slows down the optimization and increases the memory usage. In the following, We show how to use the special LAT structure and convert our problem into a sequence of efficient dynamic programs on trellises.

We rewrite Eq. (5) in a compact format:

$$\max\{c^T x : Ax = r, Bx = e\}. \tag{6}$$

Abusing notation, we use the vector x to indicate the variables in Eq. (5), i.e., x includes the X, Y, u_0, v_0, s_0 and auxiliary variables. We use the vector c to denote the objective coefficients in Eq. (5). The complex constraints (boxed) are denoted as $Ax = r$ and other constraints are denoted as $Bx = e$.

Initialization. We first remove the complex constraints $Ax = r$ and obtain a linear program LP_s. We select the lowest cost target point for each model point to maximize LP_s. The auxiliary variables $u^+_{p,q}, u^-_{p,q}, v^+_{p,q}, v^-_{p,q}, s^+_{p,q}, s^-_{p,q}$ and w are bounded in LP_s. Since their coefficients are negative in the objective function, they all should take their lower bounds. We determine the values of s_0, u_0 and v_0 using the same method.

We obtain two feasible solutions of LP_s such that their linear combination satisfies the *complex* constraints. From the initial solution, we can always reset $u^+_{p,q}, u^-_{p,q}, v^+_{p,q}, v^-_{p,q}$, $s^+_{p,q}, s^-_{p,q}$ and w so that $\alpha_i^T x_1 = r_i + 1$ and $\alpha_i^T x_2 = r_i - 1$, where α_i is the coefficient vector corresponding to the ith row of A and r_i is the ith element of r. The solution $(x_1 + x_2)/2$ is feasible for both the simple and the complex constraints. Thus, x_1 and x_2 serve as the first two proposals.

Updating tree trellis for new proposals. The goal is to find new proposals and the weights so that we can combine the proposals to optimize the objective and satisfy the complex constraints. Assume that we have $k - 1$ proposals and we introduce the kth proposal x_k so that

$$[F_k] : \max_{\lambda_1,\ldots,\lambda_k \ge 0} \left\{ \sum_{j=1}^{k} \lambda_j c^T x_j : \sum_{j=1}^{k} \lambda_j \alpha_i^T x_j = r_i, \sum_{j=1}^{k} \lambda_j = 1 \right\}, \tag{7}$$

where λ is the weight vector for the proposals. For the previous $k - 1$ known proposals, we price out the constraints of $F_{(k-1)}$ with shadow prices, which equal the optimal dual variable values. We denote the ith constraint's shadow price as d_i and the unit sum constraint's shadow price as δ. Based on the simplex method, by introducing the new proposal, the maximal gain of the objective is $\lambda_k(c^T x_k - \sum_i \alpha_i^T d_i x_k - \delta)$, recalling that the shadow price is the change of the objective per unit change of the righthand side (the constant part) of a constraint. To improve the objective, the gain has to be greater than 0. Instead of randomly searching for a new proposal, based on the Dantzig-Wolfe decomposition [40], we choose the x_k that maximizes the gain: stripping away the λ_k and δ, we maximize $\hat{c}(x) = (c^T - \sum_i \alpha_i^T d_i)x$, i.e.,

$$x_k = \arg\max_x \{\hat{c}(x) : Bx = e\}, \tag{8}$$

where the set of constraints includes the tree constraints and other bound constraints.

Using the LAT structure, we solve Eq. (8) via *dynamic programming*. Since the constraint matrix B excluding the columns and rows for variables other than X and Y is totally unimodular, and X, Y and the other variables are separable, we always have integer solutions for X and Y, and the optimization is equivalent to finding the longest paths on trellises expanded from the tree defined by $\mathcal{N}$. We *optimize the trellis paths and other variables separately*: To optimize the paths on the trellises, we first update their edge weights based on the Y variable coefficients in $\hat{c}(x)$ (all X variables have been substituted by Y variables) and then we use dynamic programming to implicitly enumerate all the feasible paths. The auxiliary variables and s_0, u_0, v_0 and w in Eq. (8) take their lower bounds or upper bounds depending on the signs of their coefficients in $\hat{c}(x)$.

Termination condition and looping. We check the optimal objective $\hat{c}(x^*)$ of the dynamic programming and proceed as follows

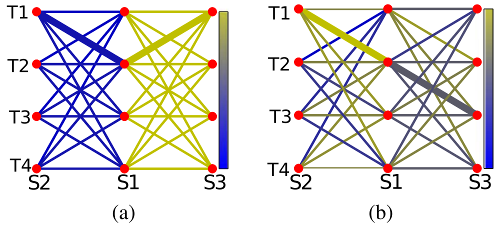
$$\hat{c}(x^*) \begin{cases} > \delta & \text{add } x^* \text{ as a proposal} \\ \leq \delta & \text{terminate.} \end{cases} \tag{9}$$

Therefore, if the gain $\hat{c}(x)$ is greater than δ, we introduce a new proposal, update the trellises and solve a new dynamic program; otherwise, the iteration terminates. The iterative process is finite and terminates with the optimum solution for the relaxed problem [40]. The optimal solution is a linear combination of the proposals.

Obtaining integral solution. The optimal solution for the relaxed solution is fractional. We convert it into an integral solution by solving a mixed-integer program. We solve the mixed-integer program that keeps only the non-zero value target points. We observe that in practice there are very few non-zero assignment variables in the relaxed solution; therefore, the complexity of this stage is negligible. This scheme ensures that if the optimal target point for each model point is non-zero in the relaxation, then the global optimum is achieved. The small mixed-integer program is solved using a branch and bound method. The complete procedure is summarized in Algorithm 1.

Algorithm 1. Scale and rotation invariant matching on linearly augmented tree (LAT) (Eq. (5))

```
Initialize Get feasible solutions $x_1$ and $x_2$ and set $k = 2$.
repeat
    Solve $F_k$ (Eq. (7)) and $k := k + 1$
    Update trellis weight (Eq. (8)) and use dynamic programming to solve for $x_k$.
until convergence (Eq. (9)).
Obtain integral solution.
```



(a) (b)

Fig. 3. The trellises for tree matching in the first (a) and last (b) stage of iteration. S1, S2 and S3 denote the three model points and T1, T2, T3 and T4 denote the four target candidates. The warm color indicates high value and cool color indicates low value. The tree optimization finds the matching such that the total edge value is the highest. The thick edges indicate the optimal matching. Note that links in this illustration are different from the ones in Fig. 5.

As a further remark, our approach is different from the Dual Decomposition [35]. The Dual Decomposition solves a different problem:

$$\min \quad \sum_i f_i(x) \quad \text{subject to} \quad x \in \mathcal{C},$$

where $\mathcal{C}$ is a convex set. It is assumed that optimizing over the individual problems, i.e., $\min\{f_i(x) : x \in \mathcal{C}\}$ is tractable. Its primal and dual problems are:

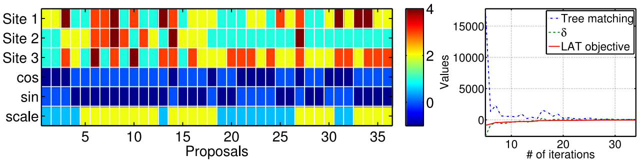
$$[P] \quad \min \sum_i f_i(x_i), \ \text{s.t.}\ x_i = x,\ x_i \in \mathcal{C}\ \ \forall i$$

$$[D] \quad \max_\lambda \min_{x_i, x} \sum_i (f_i(x_i) + \lambda_i(x_i - x)), \ \text{s.t.}\ x \in \mathcal{C}, x_i \in \mathcal{C}\ \ \forall i.$$

The dual problem [D] is separable and the projected subgradients are used to optimize the dual. Our approach is therefore different from the Dual Decomposition [35]. Our method decomposes the constraints and optimizes on a tree.

Toy example. We match a three-point template in red to the blue target points (Fig. (5)). The template's basic graph is a tree with $\mathcal{N} = \{(1, 2), (1, 3)\}$. Model point 1 matches target point 1 with cost 10, and all the other points with cost 9. Model points 2 and 3 match every target point with cost 10.

In this example, we ignore the optional term g. However, we still include the hyperedge term that involves scale and rotation consistency; it is non-trivial to solve. We construct the proposed model and solve the optimization using Algorithm 1. The solution process involves a sequence of trellis updating. In the following, we show one of the four linear programs that achieve the global optimum. Initially, the trellises are shown as Fig. 3a. The color of the edges illustrates their weight. The assignment on trees can be efficiently computed via dynamic programming: the result is $1 \to 2, 2 \to 1, 3 \to 1$. s_0, u_0, v_0 and auxiliary variables take their lower or upper bounds based on the signs of their objective coefficients. We update the trellises using the proposed scheme so that a new tree solution linearly combined with previous proposals improves the objective. The trellises evolve and at the last stage they are as shown in Fig. 3b. The tree solution is $1 \to 2, 2 \to 1, 3 \to 3$, which is the optimum. Fig. 4a shows the assignment and rotation-scale parameters in different proposals. Fig. 5 shows how the floating-point assignments for model points 1, 2, 3 and the values for s_0, u_0, v_0 change in each iteration. Fig. 4b shows the convergence process: the dynamic programming



(a) (b)

Fig. 4. (a): The proposals for the toy example. Sites 1-3 are nodes for the three template points, each of which may take value 1, 2, 3 or 4 that corresponds to the label of four target points. The proposal variables also include cosine and sin of the rotation angle and the scale. The auxiliary variables are in fact also proposal variables; their values are not shown. (b): Convergence of the toy example.

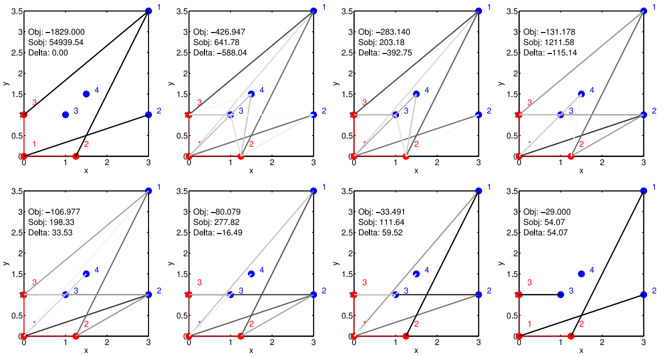


Fig. 5. Matching a three-point template. The proposed algorithm achieves optimum in 34 iterations, in which eight samples are shown. The gray levels of lines indicate the assignment strength.

solution approaches δ and the gap approaches zero as the objective improves. The proposed method yields integral solution; it is the global optimum.

Real image example. In Fig. 6 we match a fluffy animal. We find the region matching from the template to the target object. It is challenging to match these unreliable regions due to random superpixel splitting and merging. The proposed method constructs four linear programs. In Fig. 6, rows 2-5 show the intermediate matching during iterations in the optimization. The proposed method gives an accurate matching result when the iteration terminates. It is interesting to see that non-optimal solutions in the intermediate stages may give results quite far from the true matching. It is thus necessary that we try to approach the global optimum. The matching improves as more proposals are included. The last subfigure in Fig. 6 row 5 shows the result from further along in the branch and bound procedure. Fig. 8a shows the trend of the objective function, tree objective function and δ, as the iteration proceeds. The objective is optimized as the gap between the tree objective and δ approaches 0. The optimal linear program takes 525 iterations in this example.

Fig. 7 shows another example of matching the regions of a person in real images. There is large rotation and scale change in this example. Our method yields accurate matching. Fig. 8b shows the trend of energy evolution as the iteration converges to the optimal matching.

Complexity. The complexity of the proposed method depends on the size of the tree structure matching and the linear program for fusing the proposals. A standard dynamic programming solution for tree matching is $O(nm^2)$ where n is the number of model points and m is the number of target points. If we can embed the target points

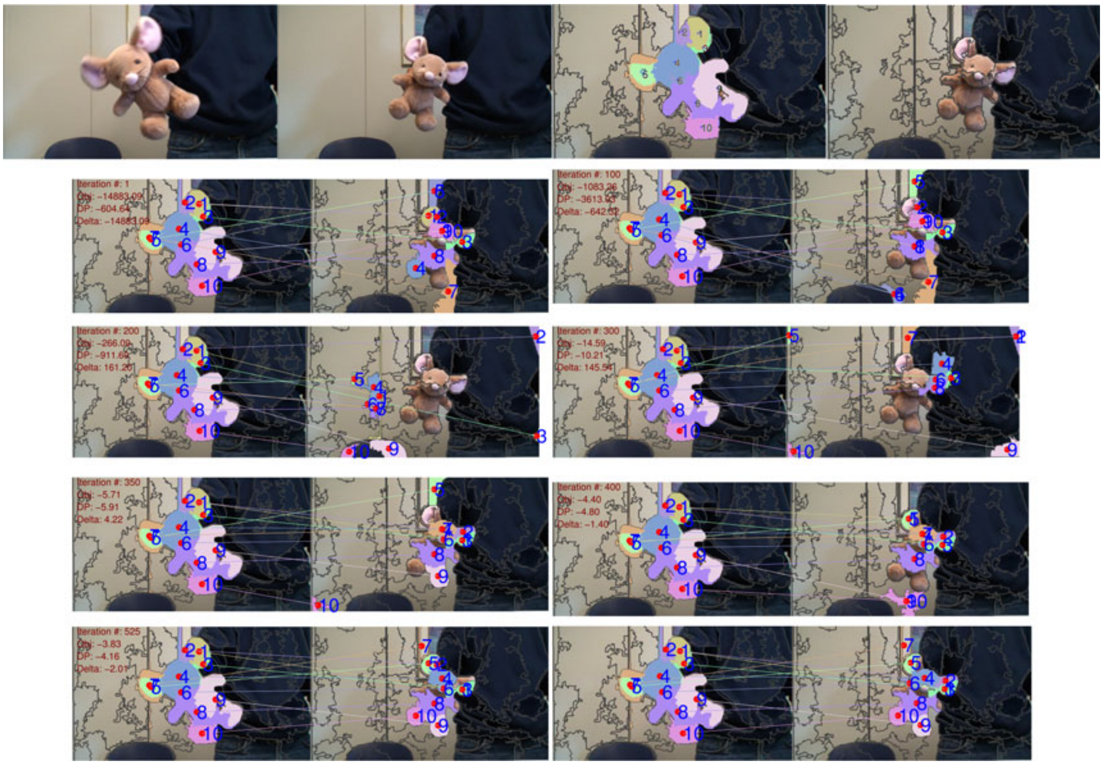


Fig. 6. Matching a fluffy animal. Row 1, from left to right: the template image, target image, model overlaid on the template image and target image superpixels. Rows 2-5: The matching improves as more proposals are included. Here we show the iterations of the optimal linear program (four linear programs are solved in the optimization). The matching result is obtained by rounding the floating point solution. The final matching result (in row five) is from the branch and bound procedure.

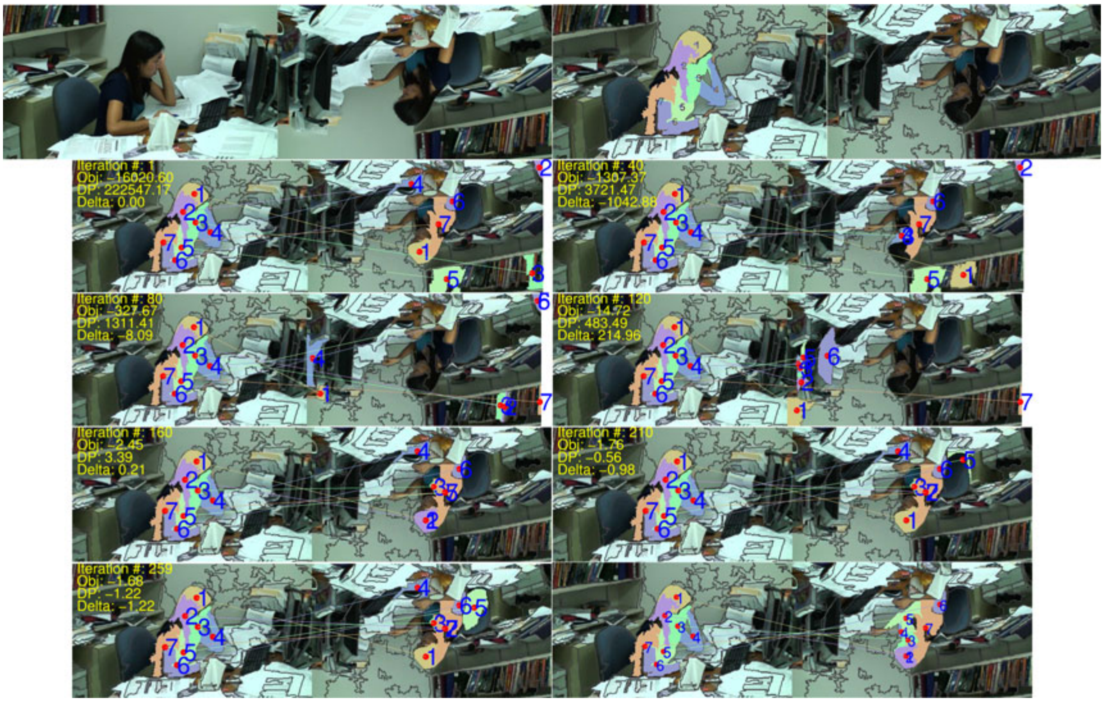
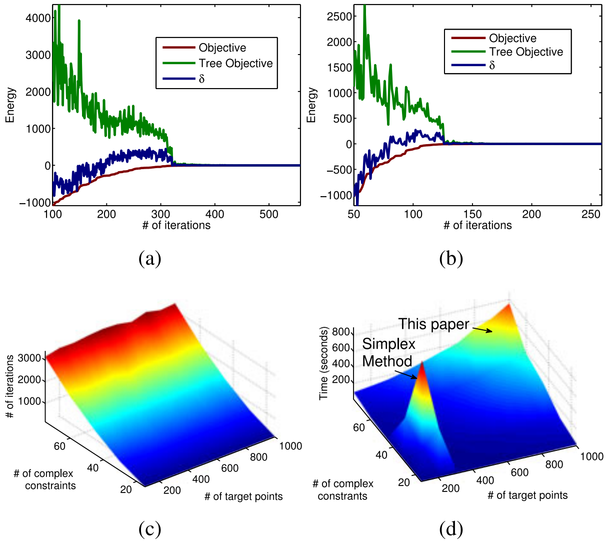


Fig. 7. Matching a person. Row 1: The template image, target image, model overlaid on the template image, and target image superpixels. Rows 2-5 show the iterations as more proposals are included. The matching result is obtained by rounding the floating point solution. The last matching result in row 5 is from the branch and bound procedure.

on grids, the complexity can be reduced to $O(nm)$. The complexity of F_k in Eq. (7) is independent of the n and m. It is mostly determined by the number of complex constraints t and the number of proposals k. With the simplex method, the average complexity is roughly $t\log(k)$ [24]. Therefore, the overall complexity of is $O(k(nm^2 + t\log(k)))$. For most problems, the optimization converges quickly. Figs. 8c and 8d illustrate the complexity of the proposed method based on the statistics of a large number of synthetic problems. The proposed method is much more efficient than a direct linear programming using the simplex method. By embedding target points on grids and using the distance transform trick [32] to solve the dynamic programming on trees, it becomes possible to solve problems with millions of target points.



(a)

(b)

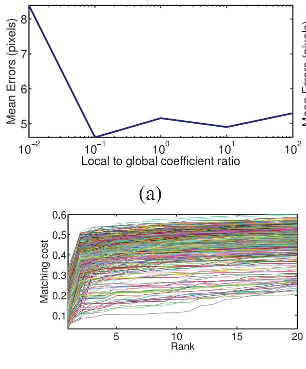
(c)

(d)

Fig. 8. (a) Energy curve for matching a fluffy animal. (b) Energy curve for matching person. (c) The number of iterations is determined by the number of complex constraints but is independent of the number of target points. (d) The proposed method is more efficient than the simplex method.

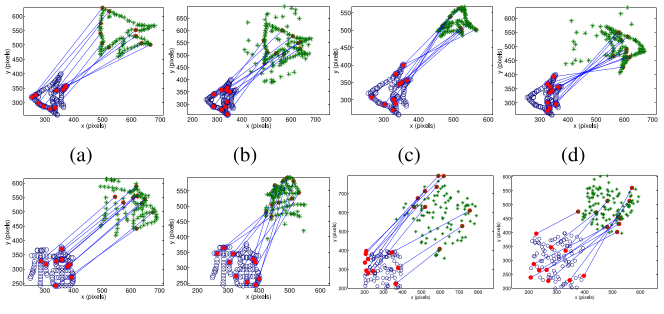
Sensitivity analysis. The proposed method has four coefficients λ, μ, γ and ϕ that control the weight for different terms in the objective function. Sensitivity analysis on linear programming has been well known [37]. One byproduct of the simplex method is that we can further obtain the range in which the parameters in the objective can vary without affecting the optimal solution. Sensitivity analysis on mixed integer linear programming is a much harder problem [38]. In this paper, we use a statistical approach to analyze sensitivity of these coefficients. We test the sensitivity of setting these parameters using ground truth point set matching. To simplify the process, we vary a single parameter in each test while keeping others fixed. In the first test, we change λ, the coefficient of the local matching cost, to test how changing the relative weight between the local term and global terms affects the result. In the second test, we change γ, the coefficient of the scale consistency term and test how changing the relative weight between the scale term and rotation term affects the result. We set the weight of the optional term, ϕ, to be zero.

For each parameter setting, we run the random point matching experiment 500 times and we compute the mean matching error. In this test, the template image contains 30 points and we randomly select 10 as the template points. The size of the template object is 200 pixels. We set the noise level to $1/3$ and the distortion parameter to 0.01. The



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

Fig. 9. (a-b): We use the ground truth point matching to test how sensitive our matching result is to the parameter setting. The template object's size is 200 pixels. As shown in the two test cases, the proposed method is not sensitive to parameter settings. (c) and (d) show the matching costs of strong and weak features on the INRIA Graffiti test images one and two. (c): Top 20 matching costs of strong features, whose lowest cost matching corresponds to the correct target. Each line corresponds to a feature point. (d): Top 20 matching costs of weak features, whose lowest cost matching does not find the target. Each line corresponds to a feature point.



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 10. Example matching of the proposed method on synthetic data: (a) fish one with zero clutter, (b) fish one with 50 clutter points, (c) fish two with zero clutter, (d) fish two with 25 clutter points, (e) Chinese character with zero clutter, (f) Chinese character with 10 clutter points, (g) random points with zero clutter and 0.1 distortion, and (h) random points with 25 percent clutter and 0.1 distortion. Ten points are randomly selected on each template for matching. We introduce different levels of clutter to the target images in the experiments. The object deformation for the fish and Chinese character is fixed. The distortion for random dot targets varies in different test cases.

distortion is defined as the ratio of the range of perturbation of each point to 200, the size of the object. Figs. 9a and 9b illustrate how the average matching error varies as the parameters change. As shown in Figs. 9a and 9b, the mean error changes gradually as the weights change in a large range; the variation of the matching error is small relative to the object size. The proposed method is thus not sensitive to small perturbations of the parameter settings.

Manual tuning can be used to adjust these few parameters λ, μ, γ and ϕ, or they can be optimized if training data is available. When the features are weak, features match to everywhere with similar costs. The local feature matching cost thus does not count much in the matching; the spatial consistency term (the global term) dominates in this case. As shown in Figs. 9c and 9d weak SIFT features have quite flat matching cost curves. For matching that involves such weak features, we do not have to change the weight between the global term and the local term; the optimization handles them implicitly. In the extreme case where most features are "bad" (local features match wrong targets with much lower costs than matching to the correct points), we need to either give a much higher weight on the spatial terms or completely discard the local cost terms. For different model graphs, ideally the parameters should be retrained to achieve the best result even though a typical setting usually works well across graph models. In this paper, the parameter setting for all the synthetic point matching experiments is $\lambda = 1, \mu = 10, \gamma = 10$ and $\phi = 0$. For all the real image matching experiments, the parameter setting is $\lambda = 1, \mu = 100, \gamma = 100$. In the SIFT feature matching and the region matching $\phi = 10^{-6}$ and in the real image patch matching $\phi = 0$.

Parameter optimization. If we have enough ground truth training samples, we can also optimize these parameters. Since we minimize the energy function in Eq. (1), we need to set the parameters so that for each training image the correct matching has lower energy than incorrect ones. For ease of notation, we use v to represent the vector of λ, μ, γ and ϕ, vector $u_p^{(i)}$ to represent the positive matching's energy values in the objective function weighted by these coefficients for training image i, and $u_n^{(i,j)}$ to represent the jth incorrect matching's energy values in the objective function weighted by these coefficients for training image i. Here, we use a similar formulation as [17]. We optimize the following linear program:

$$\begin{aligned} &\max \sum_i t_i \\ &s.t.\ v^T\left(u_n^{(i,j)} - u_p^{(i)}\right) \geq t_i,\ \forall i, j \\ &\quad v^T 1 = 1,\ v \geq 0. \end{aligned}$$

The optimal coefficients v maximize the margin between the objective function of the positive and negative samples. To avoid bias, the training samples should have the same number of template points. The optimization can be efficiently solved using the simplex or interior point method. The parameter setting is not sensitive and therefore the trained parameters are expected to be able to be generalized to real applications.

3 Benchmarking Using Ground Truth Data

We benchmark the performance of the proposed method on synthetic point data sets, which have been widely used in testing matching performance. There are two sets of test patterns: one is the fish and Chinese characters in [20], [23], the second set of test patterns consists of random dots. Example matchings provided by the proposed method on these patterns are illustrated in Fig. 10. In each experiment, we randomly select 10 model points from the template image to form a template graph. The matching is a challenging task even for clean target images since other points act as clutter points and there are 10 times more clutter points than model points. The target patterns of the fish and character are smoothly deformed from their templates, while the target points of random dots are randomly perturbed to simulate deformation. We use a distortion factor to quantify the perturbation range: a distortion of 0.1 corresponds to range 0-20 pixels and a distortion of 0.01 corresponds to range

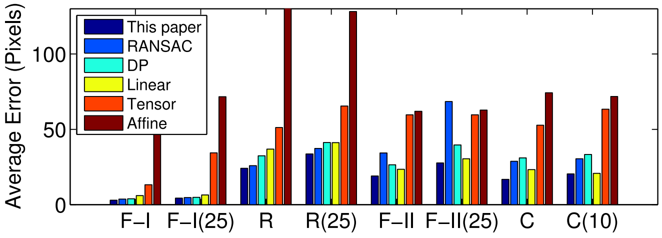


Fig. 11. Using ground truth synthetic data, we compare the proposed method with image matching approaches: tensor method [29], RANSAC [33], linear matching [28], local affine invariant matching [36] and DP on quantized scales and rotation angles. F-I and F-II are two fish pattern matching tests that include zero clutter. F-I(25) and F-II(25) use two fish patterns and introduce 25 clutter points in the target. C is the Chinese character matching test with zero clutter and C(10) includes 10 clutter points in the target. R is the random dot test in which distortion is 0.1 and clutter is 0. R(25) is the random dot test in which distortion is 0.1 and clutter is 25 percent.

0-2 pixels. Clutter points are also included in the target patterns. We randomly rotate and scale them to form the final target images. The rotation angle is from 0 to 360 degrees and the scale is from 0.5 to 2. Our method does not need to specify a small upper bound for the scale; the range $[0.5, 2]$ is required by some competing methods. Our method searches for the scale in the range of $[0.001, 1{,}000]$. For each pair of template and target candidate points, the matching cost is the lowest χ^2 distance between their shape contexts [23] over the range of scales and rotations.

We first compare the proposed method with a simple method without using global constraints such as rotation and scaling consistency. This greedy approach finds the best match for each template point solely based on the local matching cost. We test the methods using the random point test. When there is zero clutter and distortion setting is 0.001, mean matching errors for the greedy method and ours are 78 and 8.6 pixels respectively. At the level of 20 percent clutter and 0.1 distortion, the mean errors for the greedy method and ours are 98 and 19 pixels. The size of the template is around 200 pixels. These numbers show that the local matching itself is unreliable and we need the global constraints for good results.

We compare our method with state-of-the-art image matching methods, including the tensor method [29], RANSAC [33], linear matching [28] and local affine invariant matching [36]. The code for [29] is modified to include the local matching costs. In the comparisons, our method uses only the rotation and scaling global constraints, while the g term is set to 0. A dynamic programming approach is also compared: by quantizing the scale and rotation angle, each discrete case contains only unary and pairwise constraints and can be solved by dynamic programming. This DP approach is in fact a variant of the Generalized Hough Transform. The quantization intervals for the scale and rotation are 0.1 and 5 degrees respectively. The DP method uses the same set of parameters as the proposed method in the objective.

We randomly generate 500 matching problems for each test case and we use the average matching errors to quantify the performance of each method. As shown in Fig. 11 and Table 1, the proposed method has the lowest average matching error in all the tests. Interestingly, our method outperforms the discretized "exhaustive" search method (DP) under the same parameter setting: search in the continuous domain helps.

We further compare the proposed method with different generic graph matching methods such as GA [9], PM [10] , SM [7], SMAC [8], IPFP [11], RRWM [12], U [2], Rank [4] , QCV [1], Path [1], and FGM [6]. The code for these methods is from the authors' websites, apart from the implementations of methods of Rank, U, QCV and Path which are from GraphM [22]. We compare with different graph matching methods using the random point matching ground truth test. The template contains 10 points in the template image. Similar to the previous random point matching test, a target is a scaled, rotated and perturbed version of a template point set with extra clutter points included. We use the shape context distance in different rotations and scales to measure the local matching costs. We adjust the combination weight between the local term and the global term for the competing methods to achieve their best results. We change the clutter level and distortion setting and repeat each experiment for 500 times. We use the average matching error to quantify the performance. As shown in Fig. 12 and Table 2, when there is no clutter, FGM gives the lowest matching error and when the clutter increases from 0 to 20 and 50 percent, our method out-performs all the competing methods. The proposed method is also several orders faster than FGM.

4 Evaluation on Real Images and Videos

We evaluate the proposed method using a variety of videos on different features including SIFT [13], image patches, and unreliable regions. Using a randomly selected template in each experiment, we use the proposed method to match the target object in cluttered videos. We also compare

TABLE 1
Matching Error Comparison with Image Matching Methods

	This Paper	RANSAC	DP	Linear	Tensor	Affine
Fish I (Clutter point # = 0)	**3.1447**	3.7736	4.1929	6.2893	13.417	48.637
Fish I (Clutter point # = 25)	**4.6122**	5.0314	5.2411	6.499	34.591	71.908
Random Dots (C = 0, D = 0.1)	**24.319**	26.205	32.704	36.897	51.363	130.4
Random Dots (C = 25%, D = 0.1)	**33.753**	37.317	41.3	41.09	65.618	128.3
Fish II (Clutter point # = 0)	**19.078**	34.591	26.415	23.48	59.958	62.055
Fish-II (Clutter point # = 25)	**27.883**	68.553	39.623	30.608	59.748	62.893
Character (Clutter point # = 0)	**16.981**	28.931	31.027	23.27	52.83	74.423
Character (Clutter point # = 10)	**20.545**	30.608	33.333	20.755	63.312	71.908

Best method for each test is shown in bold.

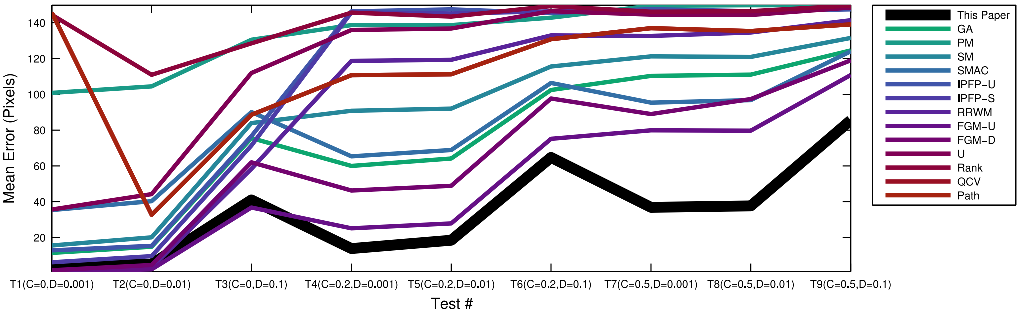


Fig. 12. We compare the proposed method with GA [9], PM [10] , SM [7], SMAC [8], IPFP-U and IPFP-S [11], RRWM [12], FGM-U and FGM-D [6], U [2], Rank [4] , QCV [1], and Path [1]. In test T1-3, clutter C is 0 and distortion D is from 0.001 to 0.1. In test T4-6, clutter C is 20 percent and distortion D is from 0.001 to 0.1. In test T7-9, clutter C is 50 percent and distortion D is from 0.001 to 0.1. IPFP-U and IPFP-S use undirected and directed graph respectively. FGM-U and FGM-D also use undirected and directed graph.

TABLE 2
Matching Error Comparison with Graph Matching Methods

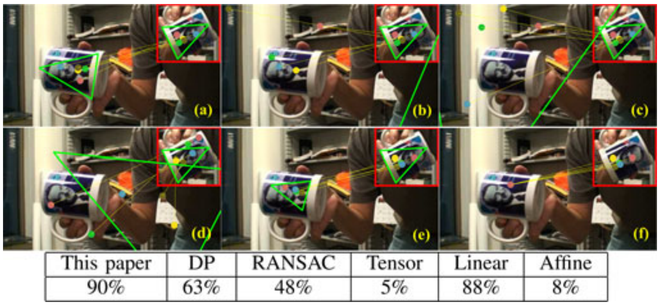
	This paper	GA	PM	SM	SMAC	IPFP-U	IPFP-S	RRWM	FGM-U	FGM-D	U	Rank	QCV	Path
T1 ($C = 0$, $D = 0.001$)	4.4328	11.37	100.81	15.531	35.305	12.747	6.1172	1.166	**0.9532**	1.638	35.598	144.51	145.69	145.69
T2 ($C = 0$, $D = 0.01$)	5.0628	14.822	104.48	20.043	40.315	15.29	9.5672	3.252	**2.178**	4.5524	44.232	110.94	32.701	32.701
T3 ($C = 0$, $D = 0.1$)	40.951	75.533	130.56	83.946	90.13	76.702	71.383	58.363	**36.753**	62.006	111.97	128.5	88.57	88.57
T4 ($C = 0.2$, $D = 0.001$)	**13.801**	59.994	138.71	90.798	65.322	146.42	145.78	118.68	25.109	46.279	135.98	145.7	110.73	110.73
T5 ($C = 0.2$, $D = 0.01$)	**18.414**	64.083	138.72	92.012	68.894	147.48	145.24	119.33	27.798	48.87	136.78	143.44	111.2	111.2
T6 ($C = 0.2$, $D = 0.1$)	**64.708**	102.4	142.85	115.61	106.38	145.53	146.46	132.97	75.14	97.684	146.48	149.33	130.8	130.8
T7 ($C = 0.5$, $D = 0.001$)	**36.863**	110.3	149.51	121.26	95.368	147.62	146.22	132.64	79.851	88.988	144.55	146.21	137.06	137.06
T8 ($C = 0.5$, $D = 0.01$)	**37.697**	111.08	149.82	120.92	96.812	146.52	145.64	134.6	79.687	97.386	144.4	146.48	135.36	135.36
T9 ($C = 0.5$, $D = 0.1$)	**85.572**	124.56	149.85	131.6	124.09	147.56	148.86	141.48	110.79	119.04	148.49	149.64	139.15	139.15

Best method for each test is shown in bold.

with the competing methods used in the synthetic data experiments.

Matching SIFT. We first match SIFT features and test whether the proposed method still has an advantage over the competing image matching methods. In this experiment, the proposed method also uses an optional affine constraint g, i.e., the target of the root model point is constrained to be close to a point that is the linear combination of all the other target points; the coefficients are determined from the layout of the model points. We select the top five model points with the lowest best and second-best matching candidate cost ratio [13] to form the model graph. To simulate challenging situations, for each model point, we corrupt the best matching candidate cost and let it equal the second best matching cost. Fig. 13 shows the comparison results for the 429-frame cup sequence. Due to the complexity of the tensor method [29] we have to use a higher threshold to reduce the number of SIFT features. We use visual inspection to quantify the detection rate: if all the model points match correctly, we have a correct detection. In this experiment, the proposed method achieves a 90 percent detection rate, which is the highest. It also has a complexity similar to the efficient linear [28] and affine [36] methods.

Matching image patches. We test the reliability of the algorithm when using non-distinctive features. We use edge pixels and image patches for matching. The target candidate points include all the edge pixels in the target image. The local matching cost is the lowest cost of the image patch matching at different rotations; because the image patch is small it is roughly scale invariant. We run the algorithm on a 400-frame sequence of a person running (Fig. 14). With such rough features, the proposed method still reliably matches the target with a 97 percent



This paper	DP	RANSAC	Tensor	Linear	Affine
90%	63%	48%	5%	88%	8%

Fig. 13. Matching 429-frame cup sequence. The sample result shows how the proposed method (a) improves the result over DP (b), RANSAC (c), tensor [29] (d), linear [28] (e) and local affine [36] (f) method. The table summarizes the detection rates for the video.

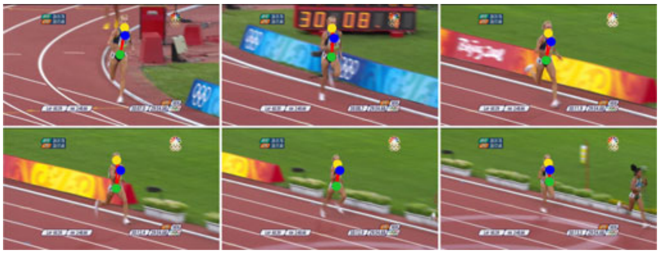


Fig. 14. Matching using edge pixels on a 400-frame sequence of a person running. The detection rate is 97 percent. Average running time is 0.7 s per frame.

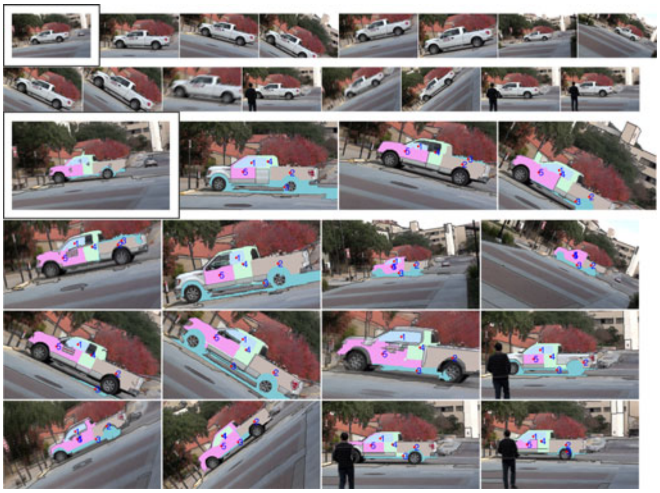


Fig. 15. Matching a car using superpixels. The first images in rows 1 and 3 show the template image and the region template. Five regions are found on the template and form a graph model. Using the template we match regions in the target images. Original target images are shown in rows 1-2. The matching results are shown in rows 3-6.

detection rate. It is also efficient: the typical running time of the optimization is $0.7\,s$ per frame.

Matching unreliable regions. We demonstrate the ability of LAT method on using simple features to match unreliable regions. This setting enables fast object matching. However, drastic region variation, non-distinctive features and strong clutter also pose a great challenge. Previous techniques [41], [42] rely on strong features that are expensive to compute or hierarchical region matching that has high complexity.

We over-segment images into superpixels using [31]. The model points and target points are superpixel weight centers in the template and target images. The weak features we use include: the average chromaticity of each superpixel, and a shape feature defined as the ratio between the two eigenvalues of the xy coordinate covariance matrix. Apart from the global constraint of rotation and scale, we also use a linear constraint g to enforce the total area consistency. This term is optional. We use it mainly to demonstrate the flexibility of the proposed method when introducing more constraints. Even though superpixels may change size arbitrarily, the overall object size equals the template size scaled by a factor. g is defined as $g = |\mathrm{tr}(R^T X) - s_0^2 t_a|$ where R is the target area matrix and t_a is the template area. The global constraint function g can be linearized by letting $g_c = \mathrm{tr}(R^T X) - s_0^2 t_a - w^+ + w^-$ and $g_o = \phi(w^+ + w^-)$ in Eq. (5), where ϕ is a constant coefficient. s_0^2 can in fact be replaced by a linear term s_0 and at the same time we need to square the constant scaling matrices S in Eq. (5).

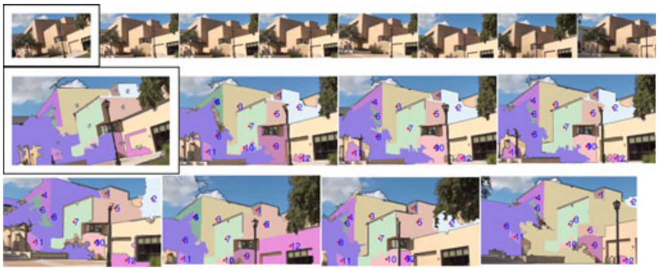


Fig. 17. Matching buildings using superpixels. The first images in rows 1 and 2 show the template image and the region template. Twelve regions are found on the template and form a graph model. Using the template we match regions in the target images. Original target images are shown in row 1. The matching results are shown in rows 2-3.

Fig. 15 shows the result of matching a car in a video sequence. The template image is randomly chosen. The template graph is automatically generated from superpixels in a manually labeled foreground area. Using the single template, the proposed method finds the target cars in all the target images. The superpixels, as shown in Fig. 15, are not guaranteed to be consistent. There is unknown scaling and rotation of the target object in these images. The fast motion of the camera causes considerable motion blur and a person passing the car also causes partial occlusion. It is challenging to match the target object. Our proposed method matches the target car reliably in the video. In Fig. 16, we show how the proposed method can be used to successfully

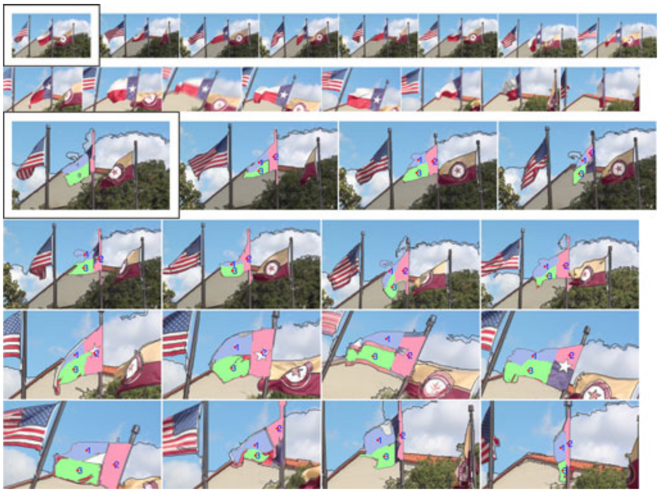


Fig. 16. Matching a flag using superpixels. The first images in rows 1 and 3 show the template image and the region template. Three regions are found on the template and form a graph model. Using the template we match regions in the target images. Original target images are shown in rows 1-2. The matching results are shown in rows 3-6.

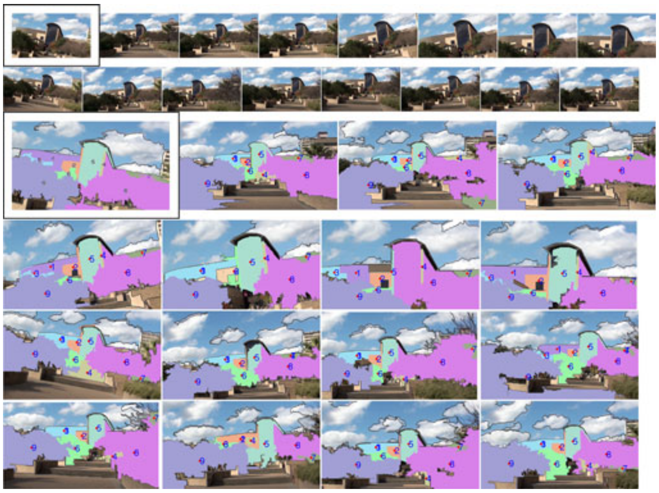
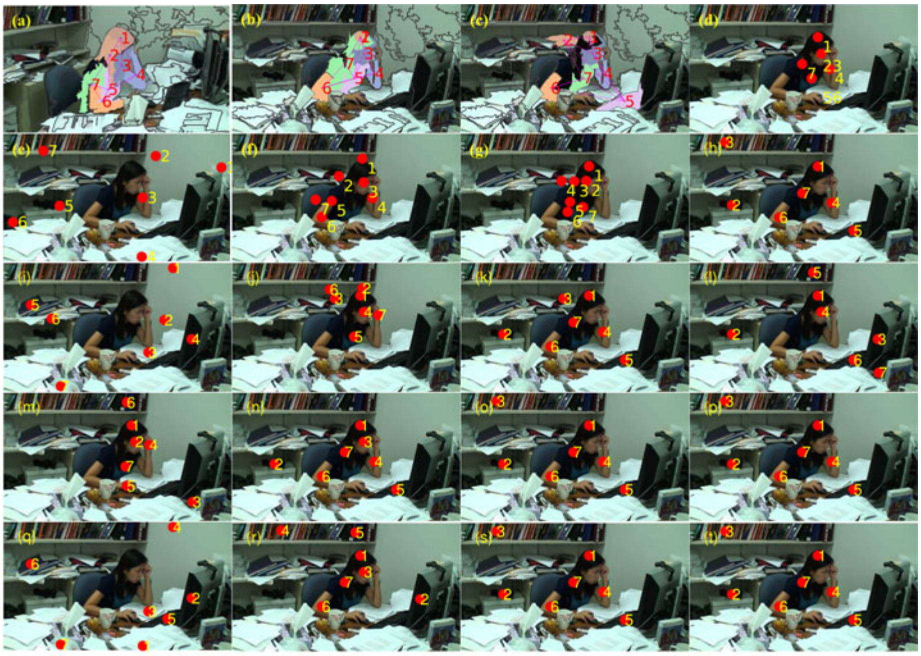


Fig. 18. Matching an outdoor scene using superpixels. The first images in rows 1 and 3 show the template image and the region template. Nine regions are found on the template and form a graph model. Using the template we match regions in the target images. Original target images are shown in rows 1-2. The matching results are shown in rows 3-6.

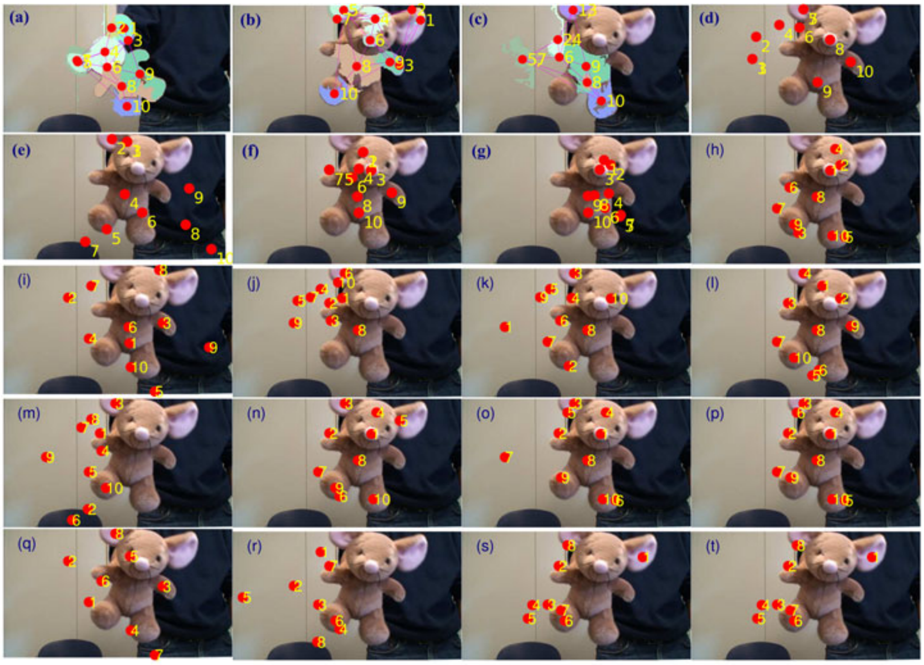


This paper	DP	RANSAC	Tensor	Linear	Affine	GA	PM	SM	SMAC	IPFP-U	IPFP-S	RRWM	FGM-U	FGM-D	U	Rank	QCV	Path
91%	73%	43%	11%	74%	14%	2.3%	0	2.3%	1.5%	0	3.4%	2.3%	0.8%	1.1%	0	0.38%	11%	11%

Fig. 19. Matching using superpixels on the 264 frame girl sequence. The girl sequence has strong background clutter and unstable superpixels. (a) is the template. The sample result of (b) the proposed method is superior to (c) DP, (d) RANSAC, (e) tensor [29], (f) linear [28], (g) local affine [36], (h) GA [9], (i) PM [10], (j) SM [7], (k) SMAC [8], (l) IPFP-U [11], (m) IPFP-S [11], (n) RRWM [12], (o) FGM-U [6], (p) FGM-D [6], (q) U [2], (r) Rank [4], (s) QCV [1] and (t) Path [1] method. The table summarizes the detection rates in the whole sequence.

match a flag, a deformable object, using a single graph template. The target object also has large unknown scale changes. In Figs. 17 and 18, we match buildings and an outdoor scene layout. In this experiment, we have more template parts. The proposed method still maintains the efficiency and gives reliable results.

We further compare our method with 18 competing methods on the challenging girl (Fig. 19) and mouse



This paper	DP	RANSAC	Tensor	Linear	Affine	GA	PM	SM	SMAC	IPFP-U	IPFP-S	RRWM	FGM-U	FGM-D	U	Rank	QCV	Path
90%	86%	20%	16%	34%	1%	0.2%	0	1.2%	0.5%	0	0	1.8%	5.3%	9.8%	0.91%	0	5.9%	5.9%

Fig. 20. Matching using superpixels on the 551-frames mouse sequence. The superpixels are unstable and change drastically from frame to frame due to the subtle color difference on the object and shading changes when the object rotates. The color of the mouse is also similar to the superpixels on the wall. (a) is the template. In the target frame, (b) the proposed method succeeds, but competing methods (c) DP, (d) RANSAC, (e) tensor [29], (f) linear matching [28], (g) local affine matching [36], (h) GA [9], (i) PM [10], (j) SM [7], (k) SMAC [8], (l) IPFP-U [11], (m) IPFP-S [11], (n) RRWM [12], (o) FGM-U [6], (p) FGM-D [6], (q) U [2], (r) Rank [4], (s) QCV [1] and (t) Path [1] fail. The table summarizes the detection rates in the video.

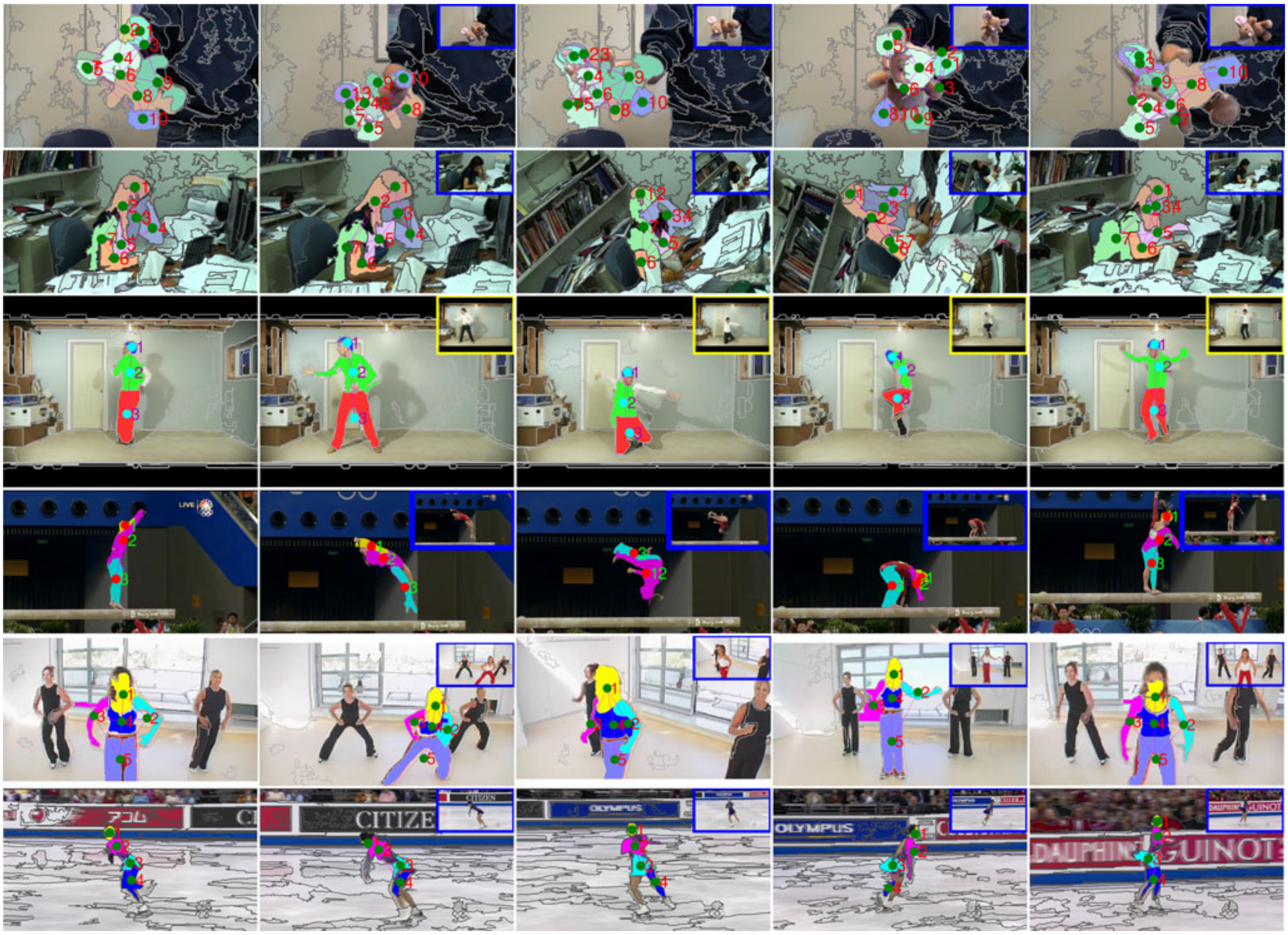


Fig. 21. Results of the proposed method on real world data. The first column shows the randomly selected templates. Row 1: `mouse` (551 frames) has drastic superpixel changes and similar foreground to the background. Row 2: `girl` (264 frames) has strong clutter. Row 3-6: `dance-I` (713 frames), `gym` (386 frames), `dance-II` (792 frames), and `skate` (472 frames) have complex articulated movement, large deformation and self-occlusion. `Dance-II` also includes a few human subjects with similar shapes and colors that form hard structured clutter. The videos are at http://www.cs.bc.edu/ ~hjiang/details/cvpr11/index.html.

(Fig. 20) sequences. These sequences involve rotation and large scale changes. The human subject in the `girl` sequence also includes articulated movement. In this test, the detection rate is determined by visual inspection. Due to unreliable segmentation, we check the overall detection result, i.e., we transform the model points using an affine transformation that is based on the region center correspondence and examine whether the matching is correct. Consistent with the results from the ground truth experiments, the matching results of the proposed method are significantly better than all the competing methods. The proposed method still works reliably even when the local features become quite weak. The LAT model enables the proposed method to maintain the performance in challenging situations. It is also efficient: the optimization takes less than a second for a target image with hundreds of superpixels.

In this test, RANSAC gives more reliable results than many more complex matching schemes. This is in fact not a surprise, RANSAC can still work in highly cluttered images, because it only requires two feature points to have roughly correct matching to succeed. Many graph matching methods are more sensitive to clutter. The second best method is the brute force exhaustive search in each quantized scale and rotation. The downside of this approach is that it needs detailed quantization intervals to give good results and therefore the procedure that enumerates all these cases is several orders of magnitude slower than the proposed method. The proposed method works directly in the continuous domain of scale and rotation and is equally efficient regardless of what the scaling range is.

We apply the proposed method on four other challenging video sequences downloaded from YouTube (Fig. 21). The detection rates and average running time of the proposed method when applied to the six different videos are listed in Fig. 22. The proposed method robustly matches the target in these sequences with a detection rate from 90 to 98 percent. In the dancing sequence, our method matches the articulated person correctly because region matching reduces articulated matching to deformable object matching problems. Our method can also deal with partial occlusion

	Mouse	Girl	Dance-I	Gym	Dance-II	Skate
Num. Frames	551	264	713	386	792	472
Rate	90%	91%	98%	90%	94%	91%
Avg. Time (s)	0.78	0.42	0.03	0.02	0.07	0.05

Fig. 22. The average running time for optimization in one frame is measured on a 2.8 GHZ machine.

by enforcing the global spatial constraint. When feature points are occluded in target images, the corresponding template points match all the candidates with similar costs and global spatial constraints help find the correct matching. The few failure cases in these tests are due to the simple features we use; more sophisticated features will further improve the performance.

5 Conclusion

We propose a novel formulation for scale and rotation invariant matching using Linearly Augmented Tree constraints. Due to LAT's special structure, we can solve the relaxed matching problem efficiently by solving a sequence of easier dynamic programming problems. The proposed method operates in the continuous domain and therefore avoids the problem of quantizing scale and rotation parameters. The optimization algorithm also searches in virtually unbounded scaling ranges with the same efficiency. Our experimental results on ground truth data and real images demonstrate that the proposed method is more reliable than previous methods. The experiments confirm that our method attains high performance even on very weak features such as unreliable regions. We believe our method is generic and can be adapted to solve problems in other application domains including pose estimation and object tracking.

Acknowledgments

This work was supported in part through US National Science Foundation grants 0713168, 0910908, 1029430 and 1018641. H. Jiang is the corresponding author.

References

[1] M. Zaslavskiy, F. Bach, and J.-P. Vert, "A path following algorithm for the graph matching problem," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 31, no. 12, pp. 2227–2242, Dec. 2009.

[2] S. Umeyama, "An eigendecomposition approach to weighted graph matching problems," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 10, no. 5, pp. 695–703, Sep. 1988.

[3] H. A. Almohamad and S. O. Duffuaa, "A linear programming approach for the weighted graph matching problem," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 15, no. 5, pp. 522–525, May 1993.

[4] R. Singh, J. Xu, and B. Berger, "Pairwise global alignment of protein interaction networks by matching neighborhood topology," in *Proc. 11th Annu. Int. Conf. Res. Comput. Molecular Biol.*, 2007, pp. 16–31.

[5] J. Maciel and J. P. Costeira, "A global solution to sparse correspondence problems," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 25, no. 2, pp. 187–199, Feb. 2003.

[6] F. Zhou and F. De la Torre, "Deformable graph matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 2922–2929.

[7] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *Proc. 10th Int. Conf. Comput. Vis.*, 2005, pp. 1482–1489.

[8] T. Cour, P. Srinivasan, and J. Shi, "Balanced graph matching," in *Proc. Adv. Neural Inform. Process. Syst.*, 2006, pp. 313–320.

[9] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 18, no. 4, pp. 377–388, Apr. 1996.

[10] R. Zass and A. Shashua, "Probabilistic graph and hypergraph matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[11] M. Leordeanu, M. Hebert, and R. Sukthankar, "An integer projected fixed point method for graph matching and MAP inference," in *Proc. Adv. Neural Inform. Process. Syst.*, 2009, pp. 1114–1122.

[12] M. Cho, J. Lee, and K. Lee, "Reweighted random walks for graph matching," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 492–505.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput.Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[14] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *Int. J. Comput.Vis.*, vol. 61, no. 1, pp. 55–79, 2005.

[15] S. Roy and I. J. Cox, "A maximum-flow formulation of the n-Camera stereo correspondence problem," in *Proc. Int. Conf. Comput. Vis.*, 1998, pp. 492–499.

[16] R. O. Duda and P. E. Hart, "Use of the hough transform to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[17] M. Szummer, P. Kohli, and D. Hoiem, "Learning CRFs using graph cuts," in *Proc. 10th Eur. Conf. Comput. Vis.*, 2008, pp. 582–595.

[18] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[19] Y. Weiss and W. T. Freeman, "On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 723–735, Feb. 2001.

[20] H. Chui and A. Rangarajan, "A new algorithm for non-rigid point matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2000, vol. 2, pp. 44–51.

[21] A. C. Berg, T. L. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondence," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 26–33.

[22] GraphM-Graph matching package, (2011). [Online]. Available: http://cbio.ensmp.fr/graphm

[23] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 24, no. 4, pp. 509–522, Apr. 2002.

[24] V. Chvátal, *Linear Programming*. New York, NY, USA: Freeman, 1983.

[25] C. J. Taylor and A. Bhusnurmath, "Solving image registration problems using interior point methods," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 638–651.

[26] N. Komodakis and G. Tziritas, "Approximate labeling via graph-cuts based on linear programming," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol, 29, no. 8, pp. 1436–1453, Aug. 2007.

[27] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 28, no. 10, pp. 1568–1583, Oct. 2006.

[28] H. Jiang and S. X. Yu, "Linear solution to scale and rotation invariant object matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 2474–2481.

[29] O. Duchenne, F. Bach, I. Kweon, and J. Ponce, "Tensor-based algorithm for high-order graph matching," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 2383–2395.

[30] H. Li, X. Huang, and L. He, "Object matching using a locally affine invariant and linear programming techniques," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 36, no. 2, pp. 411–424, Feb. 2013.

[31] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, 2004.

[32] P. Felzenszwalb and D. Huttenlocher, "Efficient matching of pictorial structures," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2000, pp. 66–73.

[33] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[34] L. Torresani, V. Kolmogorov, and C. Rother, "Feature correspondence via graph matching: Models and global optimization export," *Eur. Conf. Comput. Vis.*, 2008.

[35] N. Komodakis, N. Paragios, and G. Tziritas, "MRF energy minimization and beyond via dual decomposition," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 33, no. 3, pp. 531–552, Mar. 2011.

[36] H. Li, E. Kim, X. Huang, and L. He, "Object matching with a locally affine-invariant constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 1641–1648.

[37] S. P. Bradley, A. C. Hax, and T. L. Magnanti, *Applied Mathematical Programming*. Reading, MA, USA: Addison-Wesley, 1977.

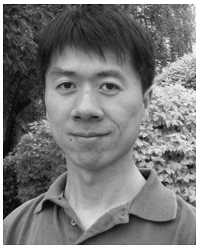
[38] M. W. Dawande and J. N. Hooker, "Inference-based sensitivity analysis for mixed integer/linear programming," *Oper. Res.*, vol. 48, no. 4, pp. 623–634, Jul. 2000.

[39] A. Albarelli, E. Rodolà, and A. Torsello, "Imposing semi-local geometric constraints for accurate correspondences selection in structure from motion: a game-theoretic perspective," *Int. J. Comput. Vis.*, vol. 97, no. 1, pp. 36–53, Mar. 2012.

[40] G. B. Dantzig and P. Wolfe, "Decomposition principle for linear programs," *Oper. Res.*, vol. 8, no. 1, pp. 101–111, 1960.

[41] V. Hedau, H. Arora, and N. Ahuja, "Matching images under unstable segmentations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[42] S. Todorovic and M. C. Nechyba, "Dynamic trees for unsupervised segmentation and matching of image regions," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 27 no. 11, pp. 1762–1777, Nov. 2005.



Hao Jiang received the PhD degree from Simon Fraser University in 2006. He is currently an associate professor in the Computer Science Department at Boston College. Before he joined BC, he was an associate researcher at Microsoft Research Asia and a postdoctoral research fellow at the University of British Columbia. His current research interests include object matching and detection, tracking, human pose estimation, and action recognition.

Tai-Peng Tian received the PhD degree from Boston University in 2011. Currently, he is a research scientist in the Computer Vision Lab at the General Electric Global Research Center. He was a research scientist at Singapore General Hospital, where his work focused on medical image analysis. His recent research interests include human motion analysis, discrete, and combinatorial optimization.

Stan Sclaroff received the bachelors degree in computer science and English from Tufts University, in 1984, and the masters and PhD degrees from the MIT Media Lab, in 1991 and 1995, respectively. He is currently a professor in the Department of Computer Science at Boston University. He chaired the Department from 2007 to 2013. He is a founder and co-head of the Image and Video Computing research group at Boston University. His current research interests include object tracking and recognition, analysis of human activity and gesture, and image/video database indexing, retrieval and data mining methods.