Matching by Linear Programming and Successive Convexification

Hao Jiang, *Member*, *IEEE*, Mark S. Drew, *Member*, *IEEE*, and Ze-Nian Li, *Member*, *IEEE*

Abstract—We present a novel convex programming scheme to solve matching problems, focusing on the challenging problem of matching in a large search range and with cluttered background. Matching is formulated as metric labeling with L_1 regularization terms, for which we propose a novel linear programming relaxation method and an efficient successive convexification implementation. The unique feature of the proposed relaxation scheme is that a much smaller set of basis labels is used to represent the original label space. This greatly reduces the size of the searching space. A successive convexification scheme solves the labeling problem in a coarse to fine manner. Importantly, the *original* cost function is reconvexified at each stage, in the new focus region only, and the focus region is updated so as to refine the searching result. This makes the method well-suited for large label set matching. Experiments demonstrate successful applications of the proposed matching scheme in object detection, motion estimation, and tracking.

Index Terms—Matching, correspondence, linear programming, successive relaxation.

1 Introduction

Matching is one of the most important tasks in computer vision. It has many applications in object recognition, 3D object reconstruction, motion estimation, and tracking. Matching can be mathematically formulated as a metric labeling problem, for assigning labels (e.g., the corresponding target pixel or displacement vector) to sites (e.g., a subset of the feature points on the template) such that a predefined energy function is minimized. In metric labeling, labels are defined in a metric space, yielding a label distance measure. Although simple in concept, metric labeling is NP-hard in general. For some special cases, for instance, when sites have linear or tree order, dynamic programming [1] can be used to solve the labeling problem in polynomial time. Another special case is when labels for each site have linear order and the metric defined in the label space is convex. In this case, polynomial-time max-flow schemes [2], [3] can be applied. Other searching schemes, e.g., branch and bound schemes [4], whose worst and average complexities are exponential, have also been applied to medium-size matching problems. For general metric labeling, approximation algorithms are preferred. Relaxation labeling (RL) [5] is one of the earliest methods for solving labeling problems and has had a great deal of influence on later matching schemes. RL uses local search and, therefore, relies on a good initialization. Iterative Conditional Modes (ICM) [6]—another widely applied method for solving labeling problems—is greedy and has been found to be easily trapped in a local minimum. In recent years, Graph Cut (GC) [7] and Belief Propagation (BP) [8], [9], [10] have become popular methods for metric labeling in vision. Graph Cut has been successfully applied to stereo [11], motion [12], and segmentation [13]. Loopy Belief Propagation has also been widely applied to stereo [14] and object matching [15]. GC and BP are more robust than traditional labeling schemes and are also found to be faster than methods based on stochastic annealing [16]. But, GC and BP are again very complex for large scale problems that involve a large number of labels.

The work most related to the proposed scheme is the mathematical programming schemes, which have received much interest in formulating and solving labeling problems. The early RL schemes belong to this class. A major challenge in developing an optimization algorithm is overcoming the problem of local minima in the searching process, and different schemes have been proposed. Deterministic annealing schemes [17], [18] have been successfully applied to matching point sets and graphs. Quadratic programming [19] and, most recently, semidefinite programming schemes [20] have also been proposed for image matching. To date, these methods have only been usefully applied to small scale problems. Because of its efficiency, Linear Programming (LP) has been applied in many vision problems, such as in estimating the motion of rigid scenes [21]. A linear programming formulation [22] has been presented for uniform labeling and for approximating general problems by tree metrics. Another general LP scheme studied in [23] is quite similar to the linear relaxation labeling formulation [5]. This LP formulation is found to be only applicable to small problems because of the large number of constraints and variables involved. Another major problem for traditional LP relaxation schemes is that they try to solve the labeling problem by a single relaxation process, which is usually followed by a rounding process. For complex vision problems, a single linear relaxation is usually not sufficient to capture the highly nonlinear nature of the problem.

Although intensively studied, the large scale matching problem is still unsolved. The difficulty is essentially due to the nonconvexity of the matching cost function and very large set of labels. In this paper, we present a linear programming

- *H. Jiang is with the Department of Electrical and Computer Engineering, University of British Columbia, 2332 Main Mall, Vancouver, BC Canada V6T 1Z4. E-mail: haoj@ece.ubc.ca.*
- *M.S. Drew and Z.-N. Li are with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada V5A 1S6. E-mail: {mark, li}@cs.sfu.ca.*

Manuscript received 3 Jan. 2006; revised 1 July 2006; accepted 14 Aug. 2006; published online 18 Jan. 2007.
Recommended for acceptance by R. Zabih.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0003-0106.
Digital Object Identifier no. 10.1109/TPAMI.2007.1048.

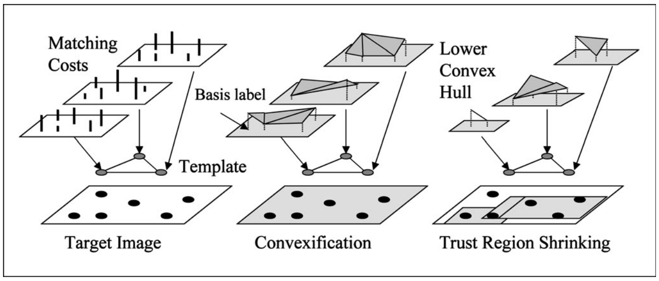


Fig. 1. Successive convexification. The basic operations include matching costs convexification and trust region shrinking.

and successive convexification method (SC-LP) for the class of metric labeling problems with L_1 regularization terms. Different from other methods in which all of the candidate labels are involved in the solution process, the proposed scheme uses a much smaller number of basis labels to represent the matching space and also progressively approaches the optimum. These components of the method greatly speed up the algorithm. In our scheme, *basis labels* correspond to the vertex coordinates of 3D lower convex hull of the matching cost surface associated with each template site. We propose a successive relaxation scheme to increase the accuracy of the approximation iteratively. During the iteration, we shrink the trust region for each site and locate a new trust region based on the previous relaxation solution, but reconvexify the *original* cost function in the refined search region. This process continues until the trust region for each site becomes small. Since the convexification process eliminates many false local minima in the earlier stages of the solution process, the proposed scheme is able to find a good approximated solution quickly. Iteratively, the successive relaxation process refines the labeling result. Fig. 1 illustrates the method of successive convexification.

The proposed successive convexification scheme is different from the well-known GNC (graduated nonconvexity) scheme. GNC is a special case of the continuation method [24]. GNC schemes have been used to convexify objective functions [25], where nonconvexity is caused by explicit nonconvex functions, e.g., the truncated L_2 distance function. GNC has also been used to convexify feasible regions for graph matching [18], where the objective functions are still nonconvex. For image matching, GNC has not been used for convexifying objective functions due to the difficulty in convexifying matching cost surfaces not defined by explicit functions. Here, we develop our successive convexification scheme to convexify the matching cost terms, making it amenable to solution by robust convex programming routines. In successive convexification, we shrink the trust region for each site and reconvexify the surfaces in these smaller trust regions.

The proposed convexification and trust region shrinking scheme is in fact quite general and can be used to improve the results of many mathematical programming-based schemes. For example, a metric labeling problem with an L_2 norm regularization term can be solved using successive convexification and convex quadratic programming solvers. We focus on linear programming schemes in this paper because the proposed LP relaxation presents an elegant and robust framework to implement the convexification and focus region concept.

2 Matching by Linear Programming and Successive Convexification

In general, deformable template matching can be stated as the following metric labeling problem. We wish to find matching function $\mathbf{f}$ such that the objective function is minimized:

$$\min_{\mathbf{f}} \left\{ \sum_{\mathbf{s}\in S} c(\mathbf{s},\mathbf{f_s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}} d(\mathbf{f_p}-\mathbf{p},\mathbf{f_q}-\mathbf{q}) \right\}, \tag{1}$$

where $c(\mathbf{s},\mathbf{f_s})$ is the cost of assigning target point $\mathbf{f_s}$ to feature point $\mathbf{s}$ on the template; $d(\cdot)$ is a convex function and $d(\mathbf{f_p}-\mathbf{p},\mathbf{f_q}-\mathbf{q})$ serves to quantify the discrepancy of matching for neighboring sites $\mathbf{p}$ and $\mathbf{q}$ in S; S is a finite set of feature points on the template; $\mathcal{N}$ is the set of nonordered neighboring site pairs, connected by edges in the Delaunay graph of point set S. In the objective function, the first term is the matching cost; the second term is a regularization term to smooth the matching for nearby feature points. Coefficients $\lambda_{\mathbf{p},\mathbf{q}}$ control the weight of the regularization term. For matching problems, the target point set can be discrete or continuous. When the target point candidates are discrete, we denote a problem as discrete matching and, otherwise, as continuous matching. In discrete matching, for each feature point $\mathbf{s}$, we can interpolate the costs $c(\mathbf{s},\mathbf{t})$ piecewise-linearly over $\mathbf{t}$ such that $c(\mathbf{s},\mathbf{t})$ become surfaces and allow $\mathbf{f_s}$ to take on continuous values in the convex hull spanned by the discrete target points: We thus obtain the *continuous extension* of a discrete matching problem. Continuous labeling such as motion estimation or template matching can be well approximated by such a continuous extension of a discrete system. In the following discussions, without loss of generality, we assume both the set of feature points S and the target point set $\mathcal{L}_\mathbf{s}$ for each $\mathbf{s}\in S$ to be discrete. In this paper, we focus on the subset of metric labeling problems in which $d(\mathbf{u},\mathbf{v}) = \|\mathbf{u}-\mathbf{v}\|$, with $\|\cdot\|$ being the L_1 norm. When matching degenerates into 1D, the max-flow scheme [3] can be used to solve the matching problem. If the dimensionality of target point space is greater than 1, the problem becomes much more complex (even though no proof is found for the NP-hardness of such problems, they are very likely to be so). In the following, we assume that $\mathbf{f}$ vectors are 2D (x and y). The methods proposed below can be easily extended to cases where the labels have higher dimensionality. To simplify notation, given $\mathbf{s}$, $c(\mathbf{s},\mathbf{t})$ over $\mathbf{t}$ is also used to represent the continuous extension matching cost surface for site $\mathbf{s}$.

2.1 Approximation by Linear Programming

The above energy optimization problem is nonlinear and usually nonconvex, which makes it difficult to solve in this original form without a good initialization process. We now show how to approximate the problem by a linear programming formulation via linear approximation and variable relaxation, as we outlined in [27], [28]. To *linearize the first term* in (1), the following scheme is applied. A "*basis*" $\mathcal{B}_\mathbf{s}$ is selected for the target points for each template site $\mathbf{s}$. Typically, the basis is the set of target pixels at the lower convex hull vertices of the matching cost surfaces. Any 2D target point $\mathbf{f_s}$ can certainly be represented as a linear combination of the basis via $\mathbf{f_s} = \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s},\mathbf{j}}\mathbf{j}$, where $\xi_{\mathbf{s},\mathbf{j}}$ are real-valued weighting coefficients. The labeling cost for $\mathbf{f_s}$ can then be arguably approximated by the linear combination of the original, basis matching costs $c(\mathbf{s}, \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s,j}}\mathbf{j}) \approx \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s,j}} c(\mathbf{s},\mathbf{j})$. We also set constraints $\xi_{\mathbf{s,j}} \geq 0$ and $\sum_{\mathbf{j}\in\mathcal{B}_s} \xi_{\mathbf{s,j}} = 1$ for each site s. Clearly, if $\xi_{\mathbf{s,j}}$ are constrained to be 1 or 0 and the basis contains all the target point candidates, i.e., $\mathcal{B}_\mathbf{s} = \mathcal{L}_\mathbf{s}$ (e.g., the whole target image), the above representation becomes exact. Note that $\mathbf{f_s}$ are *not* constrained to the basis labels, but can be any convex combination. To *linearize the regularization terms* in (1), we can represent a variable in the absolute value function by the difference of two nonnegative auxiliary variables and introduce the sum of the auxiliary variables into the objective function. If the problem is properly formulated, the sum will approach the absolute value of the variable when the linear program is indeed optimized.

Based on this linearization process, a linear programming approximation of the problem can be stated in terms of the two vector components $m = 1, 2$ (in place of x and y) as

$$\min\left\{\begin{array}{c} \sum_{\mathbf{s}\in S} \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} c(\mathbf{s},\mathbf{j})\, \xi_{\mathbf{s,j}} + \\ \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \sum_{m=1}^{2} (f^+_{\mathbf{p,q},m} + f^-_{\mathbf{p,q},m}) \end{array}\right\} \tag{2}$$

with constraints

$$\begin{aligned} &\sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s,j}} = 1\,, \ \forall \mathbf{s}\in S, \\ &\sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s,j}}\, \phi_m(\mathbf{j}) = f_{\mathbf{s},m}, \\ &\forall \mathbf{s}\in S\,,\ \phi_m(\mathbf{j}) = (m\text{th component of } \mathbf{j})\,, m = 1,2, \\ &f_{\mathbf{p},m} - f_{\mathbf{q},m} - \phi_m(\mathbf{p}) + \phi_m(\mathbf{q}) = f^+_{\mathbf{p,q},m} - f^-_{\mathbf{p,q},m}\,, \\ &\qquad \forall\, \{\mathbf{p},\mathbf{q}\}\in\mathcal{N}\,, m = 1,2, \\ &\qquad \xi_{\mathbf{s,j}},\ f^+_{\mathbf{p,q},m},\ f^-_{\mathbf{p,q},m} \geq 0. \end{aligned}$$

The matching target point $\mathbf{f_s} = (f_{\mathbf{s},1}, f_{\mathbf{s},2})$. $f^+_{\mathbf{p,q},m}$ and $f^-_{\mathbf{p,q},m}$ are auxiliary variables.

In the linear program, one of the auxiliary variable pairs $f^+_{\mathbf{p,q},m}, f^-_{\mathbf{p,q},m}$ must be zero when the linear programming is optimized. Otherwise, we can subtract the minimum of the value pair and get a new solution $f^+_{\mathbf{p,q},m} - \min(f^+_{\mathbf{p,q},m}, f^-_{\mathbf{p,q},m})$ and $f^-_{\mathbf{p,q},m} - \min(f^+_{\mathbf{p,q},m}, f^-_{\mathbf{p,q},m})$. The new solution is still feasible and has at least one zero in each auxiliary variable pair; the summation also becomes smaller. Thus, we have a better feasible solution and this contradicts the assumption that the linear program is optimized. Because one of $f^+_{\mathbf{p,q},m}$ and $f^-_{\mathbf{p,q},m}$ is zero, we must have $|f_{\mathbf{p},m} - f_{\mathbf{q},m} - \phi_m(\mathbf{p}) + \phi_m(\mathbf{q})| = f^+_{\mathbf{p,q},m} + f^-_{\mathbf{p,q},m}$. Thus, the linear program is exactly equivalent to the general nonlinear formulation if the linearization assumption, $c(\mathbf{s}, \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s,j}}\mathbf{j}) = \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s,j}} c(\mathbf{s},\mathbf{j})$, holds. This would be true for problems with convex cost surfaces.

Property 1. *If $\mathcal{B}_\mathbf{s} = \mathcal{L}_\mathbf{s}$, where $\mathcal{L}_\mathbf{s}$ is the entire label set of* s*, and the continuous extension cost function $c(\mathbf{s},\mathbf{t})$ is convex with respect to* t*, $\forall \mathbf{s}\in S$, LP exactly solves the continuous extension of the discrete labeling problem.*

Proof. We simply need to show that when LP is optimized, the minimizing LP configuration $\{\mathbf{f}^*_\mathbf{s} = \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi^*_{\mathbf{s,j}}\mathbf{j}\}$ also solves the continuous extension of the nonlinear problem. Since $c(\mathbf{s},\mathbf{t})$ is convex over $\mathbf{t}$, $\sum_{\mathbf{j}\in\mathcal{L}_\mathbf{s}} c(\mathbf{s},\mathbf{j})\xi^*_{\mathbf{s,j}} \geq c(\mathbf{s},\mathbf{f}^*_\mathbf{s})$. When the LP is minimized, we have $\sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \sum_{m=1}^{2}(f^+_{\mathbf{p,q},m} + f^-_{\mathbf{p,q},m}) \equiv \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \|\mathbf{f}^*_\mathbf{p} - \mathbf{p} - \mathbf{f}^*_\mathbf{q} + \mathbf{q}\|$. Therefore,

$$\begin{aligned} &\min\left\{\begin{array}{c} \sum_{\mathbf{s}\in S,\mathbf{j}\in\mathcal{L}_s} c(\mathbf{s},\mathbf{j})\xi_{\mathbf{s,j}} + \\ \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \sum_{m=1}^{2}(f^+_{\mathbf{p,q},m} + f^-_{\mathbf{p,q},m}) \end{array}\right\} \\ &\geq \sum_{\mathbf{s}\in S} c(\mathbf{s},\mathbf{f}^*_\mathbf{s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \|\mathbf{f}^*_\mathbf{p} - \mathbf{p} - \mathbf{f}^*_\mathbf{q} + \mathbf{q}\|. \end{aligned}$$

As well, we can always construct a feasible solution of LP that has no greater objective function than that of the continuous extension of the nonlinear optimization. Assuming solutions $\mathbf{f}'_\mathbf{s}$ minimize the continuous extension of the nonlinear problem, based on the definition, $\mathbf{f}'_\mathbf{s}$ must be located in the convex hull of $\mathcal{L}_s$. For each site s, we find feasible ξ for the LP by solving the following minimization problem for each site:

$$\begin{aligned} &\min\left\{\sum_{\mathbf{j}\in\mathcal{L}_s} c(\mathbf{s},\mathbf{j})\xi_{\mathbf{s,j}}\right\} \\ &s.t. \quad \sum_{\mathbf{j}\in\mathcal{L}_s} \xi_{\mathbf{s,j}}\, \phi_m(\mathbf{j}) = \phi_m(\mathbf{f}'_\mathbf{s}),\ m = 1,2 \\ &\qquad \sum_{\mathbf{j}\in\mathcal{L}_s} \xi_{\mathbf{s,j}} = 1,\ \xi_{\mathbf{s,j}} \geq 0, \forall \mathbf{s}\in S. \end{aligned}$$

We can set feasible values for $f^+_{\mathbf{p,q},m}$ and $f^-_{\mathbf{p,q},m}$: If $\phi_m(\mathbf{f}'_\mathbf{p}) - \phi_m(\mathbf{f}'_\mathbf{q}) - \phi_m(\mathbf{p}) + \phi_m(\mathbf{q}) \geq 0$, $f^+_{\mathbf{p,q},m} = |\phi_m(\mathbf{f}'_\mathbf{p}) - \phi_m(\mathbf{f}'_\mathbf{q}) - \phi_m(\mathbf{p}) + \phi_m(\mathbf{q})|$, and $f^-_{\mathbf{p,q},m} = 0$; else, $f^-_{\mathbf{p,q},m} = |\phi_m(\mathbf{f}'_\mathbf{p}) - \phi_m(\mathbf{f}'_\mathbf{q}) - \phi_m(\mathbf{p}) + \phi_m(\mathbf{q})|$ and $f^+_{\mathbf{p,q},m} = 0$. Based on the definition of continuous extension surface, we have $\min\left\{\sum_{\mathbf{j}\in\mathcal{L}_s} c(\mathbf{s},\mathbf{j})\xi_{\mathbf{s,j}}\right\} \leq c(\mathbf{s},\mathbf{f}'_\mathbf{s})$ for each s and

$$\begin{aligned} &\min\left\{\begin{array}{c} \sum_{\mathbf{s}\in S,\mathbf{j}\in\mathcal{L}_s} c(\mathbf{s},\mathbf{j})\xi_{\mathbf{s,j}} + \\ \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \sum_{m=1}^{2}(f^+_{\mathbf{p,q},m} + f^-_{\mathbf{p,q},m}) \end{array}\right\} \\ &\leq \sum_{\mathbf{s}\in S} c(\mathbf{s},\mathbf{f}'_\mathbf{s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \|\mathbf{f}'_\mathbf{p} - \mathbf{p} - \mathbf{f}'_\mathbf{q} + \mathbf{q}\|. \end{aligned}$$

Therefore, $\mathbf{f}^*_\mathbf{s}$ optimizes the continuous extension of the nonlinear problem:

$$\begin{aligned} &\sum_{\mathbf{s}\in S} c(\mathbf{s},\mathbf{f}^*_\mathbf{s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \|\mathbf{f}^*_\mathbf{p} - \mathbf{p} - \mathbf{f}^*_\mathbf{q} + \mathbf{q}\| \\ &= \sum_{\mathbf{s}\in S} c(\mathbf{s},\mathbf{f}'_\mathbf{s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p,q}} \|\mathbf{f}'_\mathbf{p} - \mathbf{p} - \mathbf{f}'_\mathbf{q} + \mathbf{q}\|. \end{aligned}$$

The property follows. □

In practice, the cost function $c(\mathbf{s},\mathbf{t})$ is usually highly nonconvex over $\mathbf{t}$ for each site s. In this situation, the linear program approximates the original nonconvex problem:

Property 2. *The linear program solves the continuous extension of the reformulated discrete labeling problem, with $c(\mathbf{s},\mathbf{j}), \mathbf{j}\in\mathcal{L}_\mathbf{s}$ replaced by the costs on the lower convex hull surfaces $\breve{c}(\mathbf{s},\mathbf{j})$ for each site* s*. $\breve{c}(\mathbf{s},\mathbf{t})$ is the lower convex hull surface of 3D points $(\phi_1(\mathbf{j}), \phi_2(\mathbf{j}), c(\mathbf{s},\mathbf{j}))$, $\mathbf{j}\in\mathcal{L}_\mathbf{s}$.*

The proof is similar to that of Property 1, by replacing $c(\mathbf{s},\mathbf{t})$ in the nonlinear function with its lower convex hull. An example for the lower convex hull and the coordinates of the lower convex hull vertices is illustrated in Fig. 2. For matching applications, the surface is the matching cost surface. Note that, in general, the surface may have holes or consist only of irregular discrete 3D points in the label

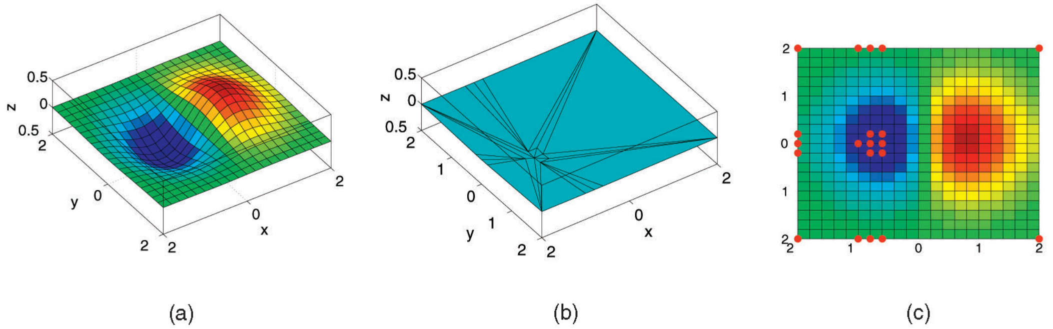


Fig. 2. Lower convex hull. (a) A cost surface. (b) Lower convex hull facets. (c) The label basis $\mathcal{B}_s$ coordinates of the lower convex hull vertices (solid dots are basis points).

versus cost space, e.g., if we only select edge points in the target images for matching. Each site $\mathbf{s}$ is associated with its own energy surface for the first term in (1).

Property 3. *For each site* $\mathbf{s} \in S$*, we need only consider the basis set* $\mathcal{B}_\mathbf{s}$ *comprised of the vertex coordinates of the lower convex hull of* $(\phi_1(\mathbf{j}), \phi_2(\mathbf{j}), c(\mathbf{s}, \mathbf{j}))$*,* $\mathbf{j} \in \mathcal{L}_\mathbf{s}$*.*

Proof. Removing some variables $\xi_{\mathbf{s},\mathbf{j}}$ from LP is equivalent to setting these variables to zero in the constraints. Since more constraints are included, the linear program involving only $\xi_{\mathbf{s},\mathbf{j}}$ corresponding to the lower convex hull vertices must have no smaller objective function when optimized. On the other hand, based on the definition of lower convex hull, for $\mathbf{f}_\mathbf{s}$ that minimize the original LP relaxation, we can find a feasible solution of the simplified linear program that has an objective function not greater than that of the original LP. Therefore, these two linear programs must be equivalent. The property follows. □

Thus, we can use only the smallest basis set—there is no need to include all the labeling assignment costs in the optimization. This is one of the key steps to speeding up the algorithm. After the convexification process, the original nonconvex optimization turns into a convex one and an efficient linear programming method can be used to yield a global optimal solution for the approximated problem. It should be noted that, although this is a convex problem, standard local optimization schemes are found to work poorly because of quantization noise and large flat areas in the convexified objective function.

Approximating the matching cost by its lower convex hull is intuitively attractive since, in the ideal case, when model (1) holds exactly, the true matching will have the lowest cost and the convexified optimum will be the exact optimum. The proposed solution of the relaxation scheme also has the following structure property:

Property 4. *If the lower convex hull of the cost function* $c(\mathbf{s}, \mathbf{t})$ *is* strictly convex over $\mathbf{t}$ for each $\mathbf{s}$, *its nonzero basis target points must be "adjacent."*

Proof. Here, "adjacent" means the convex hull of the nonzero-weighted basis target points cannot contain other basis points. Assume this does not hold for site $\mathbf{s}$ and the non-zero-weight basis points are $\mathbf{j}_k$, $k = 1 \ldots K$. Then, there exists a basis point $\mathbf{j}_r$ located inside the convex hull of $\mathbf{j}_k$, $k = 1 \ldots K$. Thus, $\exists \alpha_k$ such that $\mathbf{j}_r = \sum_{k=1}^K \alpha_k \mathbf{j}_k$ with $\sum_{k=1}^K \alpha_k = 1$, $\alpha_k \geq 0$. According to the *Karush-Kuhn-Tucker Condition* there exist $\lambda_1, \lambda_2, \lambda_3$, and $\mu_\mathbf{j}$ such that

$$c(\mathbf{s}, \mathbf{j}) + \lambda_1 + \lambda_2 \phi_1(\mathbf{j}) + \lambda_3 \phi_2(\mathbf{j}) - \mu_\mathbf{j} = 0 \text{ and}$$
$$\xi_{\mathbf{s},\mathbf{j}} \cdot \mu_\mathbf{j} = 0, \mu_\mathbf{j} \geq 0, \forall \mathbf{j} \in \mathcal{B}_\mathbf{s}.$$

This results from the Euler equation with respect to $\xi_{\mathbf{s},\mathbf{j}}$ in (2), taking into account the constraints.

Therefore, we have,

$$c(\mathbf{s}, \mathbf{j}_k) + \lambda_1 + \lambda_2 \phi_1(\mathbf{j}_k) + \lambda_3 \phi_2(\mathbf{j}_k) = 0, k = 1 \ldots K,$$
$$c(\mathbf{s}, \mathbf{j}_r) + \lambda_1 + \lambda_2 \phi_1(\mathbf{j}_r) + \lambda_3 \phi_2(\mathbf{j}_r) \geq 0.$$

On the other hand,

$$\begin{aligned} & c(\mathbf{s}, \mathbf{j}_r) + \lambda_1 + \lambda_2 \phi_1(\mathbf{j}_r) + \lambda_3 \phi_2(\mathbf{j}_r) \\ & = c\left(\mathbf{s}, \sum\nolimits_{k=1}^K \alpha_k \mathbf{j}_k\right) + \lambda_1 + \\ & \lambda_2 \phi_1\left(\sum\nolimits_{k=1}^K \alpha_k \mathbf{j}_k\right) + \lambda_3 \phi_2\left(\sum\nolimits_{k=1}^K \alpha_k \mathbf{j}_k\right) \\ & < \sum\nolimits_{k=1}^K \alpha_k c(\mathbf{s}, \mathbf{j}_k) + \lambda_1 + \\ & \lambda_2 \sum\nolimits_{k=1}^K \alpha_k \phi_1(\mathbf{j}_k) + \lambda_3 \sum\nolimits_{k=1}^K \alpha_k \phi_2(\mathbf{j}_k) = 0, \end{aligned}$$

which contradicts the *KKT* condition. The property follows. □

If we use Dantzig's simplex method [26] to solve the LP, we have the following property:

Property 5. *Using the simplex method, there will be at most three non-zero-weight basis labels for each site.*

Proof. This property is due to the basic linear programming property: If the optimum of an LP exists, the optimum must be located at one of the "extreme" points of the feasible region. The extreme points of linear programming correspond to the basic feasible solutions of LP. We denote the constraints of our linear program by $A\mathbf{x} = b$, $\mathbf{x} \geq \mathbf{0}$. Each basic feasible solution of LP has the format $[B^{-1}b, 0]^T$, where B is an invertible matrix composed of the columns of matrix A corresponding to the basic variables. For site $\mathbf{s}$, variable $\xi_{\mathbf{s},\mathbf{j}}$ introduces a column $[0, \ldots, 0, 1, \phi_x(\mathbf{j}), \ \phi_y(\mathbf{j}), 0, \ldots, 0]^T$ in A. It is not difficult to show that the submatrix generated by these columns for a single site has a rank at most 3. Therefore, we can have at most three ξ for each site in the basic variable set. This implies that *the optimum solution has at most three nonzero* ξ *for each site*.

The initial basic variables are set as follows:

- Only one $\xi_{\mathbf{s},\mathbf{j}}$ is selected as a basic LP variable for each site $\mathbf{s}$.

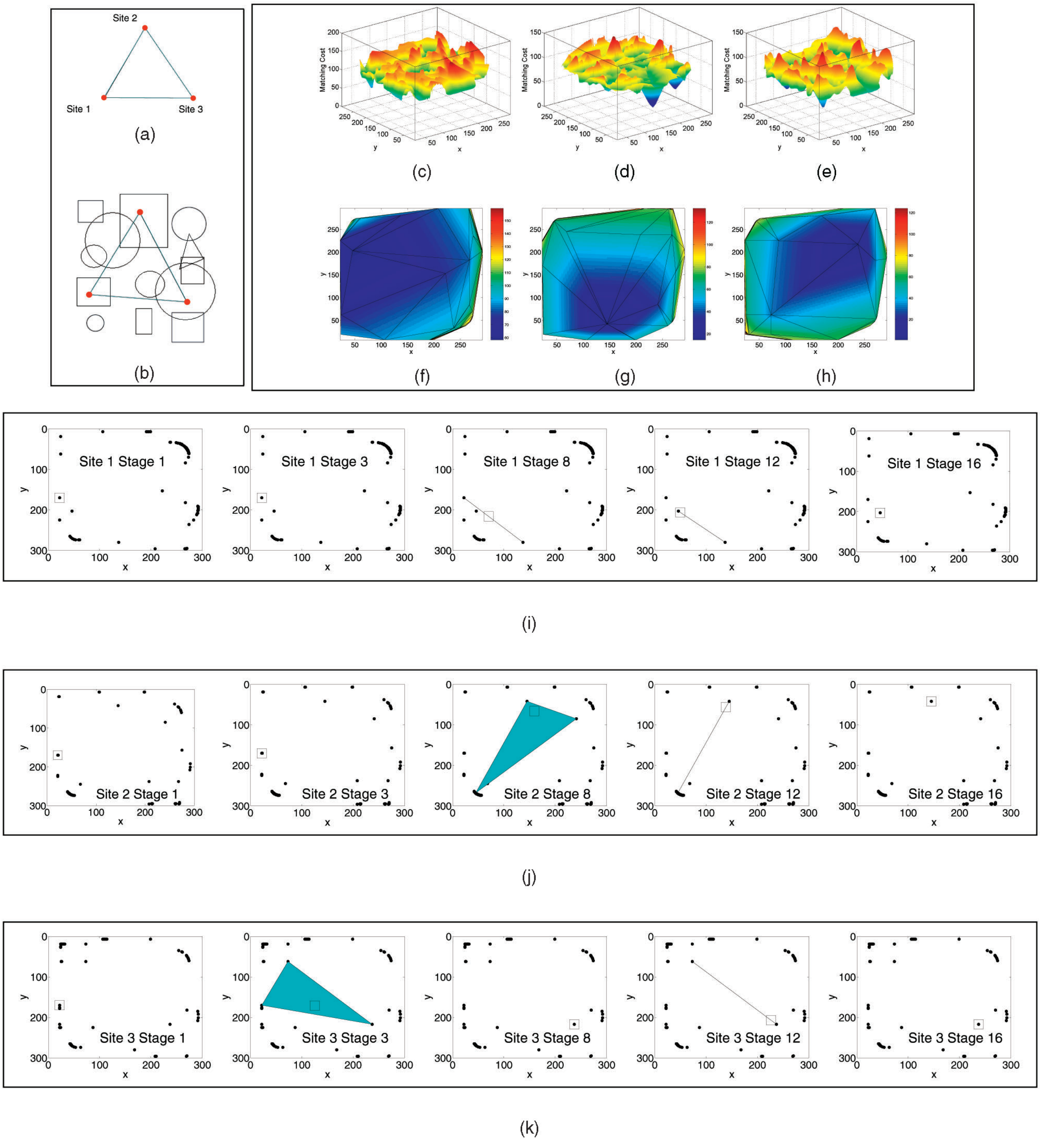


Fig. 3. Example of matching. (a) Template image. (b) Target image and LP matching result. (c), (d), and (e) Matching cost surface for sites 1, 2, and 3. (f), (g), and (h) Lower convex hull of the matching cost surface for sites 1, 2, and 3. (i), (j), and (k) Triangular basis updating for sites 1, 2, and 3 (noting that triangles often degenerate).

- $f_{\mathbf{s},m}, m = 1..2$, are basic LP variables for each site $\mathbf{s}$.
- Based on the above basic label selection, for $m = 1..2$, if $f_{\mathbf{p},m} - \phi_m(\mathbf{p}) - f_{\mathbf{q},m} + \phi_m(\mathbf{q}) \geq 0$ and $\{\mathbf{p}, \mathbf{q}\} \in \mathcal{N}$, $f^+_{\mathbf{p},\mathbf{q},m}$ is basic; else $f^-_{\mathbf{p},\mathbf{q},m}$ is set to be basic. □

Example 1 (Matching Triangles). Fig. 3 illustrates the solution procedure of the simplex method for an image matching problem. In this simple example, three feature points are selected on the object and form a triangular graph template. All of the dark pixels in the target image are matching candidates for each of the feature points. Figs. 3c, 3d, and 3e show the matching cost surfaces for each of the three points on the template, based on the normalized mean absolute difference of 7 × 7 image blocks of the distance transformation of the template and target images. Such a measure is used in this paper for binary object matching. More details about the feature are presented in Section 2.6. Figs. 3f, 3g, and 3h are the lower convex hull surfaces for the respective cost surfaces for each of the three sites. We formulate the linear program based on the proposed scheme and we use the simplex method to solve the LP. Figs. 3i, 3j, and 3k show the basic label updating process. The black dots indicate the target points located at the coordinates of the lower convex hull vertices. The target points corresponding to the basic variables are connected by lines. The small rectangle is the weighted linear combination of the target points corresponding to the basic labels at each stage. As expected, the simplex method for the proposed LP only checks triangles or their degenerates formed by basis target points. In each iteration, only one vertex of the triangle may change. When the search terminates, the patch generated by the basic variables for each site must correspond to one of the facets (edges or vertices) of the lower convex hull for each site.

2.2 Rounding to Discrete Labels

Although the continuous-weight solution obtained by LP is itself a good solution to the original problem for many applications, e.g., estimating medium scale motion [27], in many cases we would like to obtain the discrete solution in which the matching target points are only extracted from the original, discrete, target point candidate set. In the following discussions, we present bounds for the approximation scheme when the continuous solution of the linear programming is converted to a discrete one. Such a "rounding" process is also useful in estimating an upper bound and finding *anchors* (see below) in successive convexification. The bounds also partially reflect the quality of the approximation scheme.

If the label costs are not bounded above, there is no upper bound for the energy of direct LP solution (or that by directly rounded to integers) when applied to the continuous extension nonlinear problem. But, we can take a different approach. When converting a continuous solution to a solution feasible in the discrete domain, we enforce that $\xi_{\mathbf{s},\mathbf{j}}$ have only one single 1 for each site. In the following, we denote rounded version of $\xi_{\mathbf{s},\mathbf{j}}$ by $\hat{\xi}_{\mathbf{s},\mathbf{j}}$. There are two methods to enforce the constraint. The first one converts the largest $\xi_{\mathbf{s},\mathbf{j}}$ for each site to 1 and others to 0. The following proposition gives an upper bound for the rounding process:

Proposition 1. *For each $\mathbf{s} \in S$, if we round the largest $\xi_{\mathbf{s},\mathbf{j}}$ to 1 and the rest to zero, thus producing a rounded version $\hat{\mathbf{f}}_\mathbf{s}$ of the continuous solution $\mathbf{f}_\mathbf{s}$, we have an approximation algorithm bounded above by*

$$3E_{opt} + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}}(\|\hat{\mathbf{f}}_\mathbf{p} - \mathbf{f}_\mathbf{p}\| + \|\hat{\mathbf{f}}_\mathbf{q} - \mathbf{f}_\mathbf{q}\|).$$

E_{opt} is the energy of the optimum solution. $\hat{\mathbf{f}}_\mathbf{s} = \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \hat{\xi}_{\mathbf{s},\mathbf{j}}\mathbf{j}$ and $\mathbf{f}_\mathbf{s} = \sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} \xi_{\mathbf{s},\mathbf{j}}\mathbf{j}$.

Proof. First,

$$\begin{aligned}&\sum_\mathbf{s} c(\mathbf{s},\hat{\mathbf{f}}_\mathbf{s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}}\|\hat{\mathbf{f}}_\mathbf{p} - \mathbf{p} - \hat{\mathbf{f}}_\mathbf{q} + \mathbf{q}\| \\ &\le \sum_\mathbf{s} c(\mathbf{s},\hat{\mathbf{f}}_\mathbf{s}) + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}}(\|\hat{\mathbf{f}}_\mathbf{p} - \mathbf{f}_\mathbf{p}\| \\ &+ \|\mathbf{f}_\mathbf{p} - \mathbf{p} - \mathbf{f}_\mathbf{q} + \mathbf{q}\| + \|\mathbf{f}_\mathbf{q} - \hat{\mathbf{f}}_\mathbf{q}\|),\end{aligned}$$

by the triangle inequality. A direct conclusion from Property 5 is that, for each site $\mathbf{s}$, $\max_\mathbf{j}(\xi_{\mathbf{s},\mathbf{j}}) \ge 1/3$ (else the sum would be necessarily less than 1). Therefore, rounding ξ to $\hat{\xi}$, $\sum_\mathbf{s} c(\mathbf{s},\hat{\mathbf{f}}_\mathbf{s}) \to \sum_{\mathbf{s}\in S}\sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} c(\mathbf{s},\mathbf{j})\hat{\xi}_{\mathbf{s},\mathbf{j}} \le 3\sum_{\mathbf{s}\in S}\sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} c(\mathbf{s},\mathbf{j})\xi_{\mathbf{s},\mathbf{j}}$ (the maximum increase is times 3). Considering the fact that $E_{LP} = \sum_{\mathbf{s}\in S}\sum_{\mathbf{j}\in\mathcal{B}_\mathbf{s}} c(\mathbf{s},\mathbf{j})\xi_{\mathbf{s},\mathbf{j}} + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}} \|\mathbf{f}_\mathbf{p} - \mathbf{p} - \mathbf{f}_\mathbf{q} + \mathbf{q}\| \le E_{opt}$, the proposition follows. For practical computer vision problems, $\lambda_{\mathbf{p},\mathbf{q}}$ are small and most of the $\xi_{\mathbf{s},\mathbf{j}}$ in fact approach 1 or 0 and, therefore, $\hat{\mathbf{f}}_\mathbf{s}$ approaches $\mathbf{f}_\mathbf{s}$, the extra term approaches zero. □

One of the problems of the above rounding process is that it does not consider the neighbor's status and may result in topology changes. To solve this problem, we propose another method for the rounding process: We check the discrete target points and select the one that minimizes the nonlinear objective function, given the configuration of continuous matching results defined by the LP solution of the current stage. This step is similar to a single iteration of an ICM algorithm with LP solution as the initial value. We call the new rounding scheme a *consistent rounding* process.

Let $\mathbf{m}_\mathbf{s}$ be the global optimal solution and $\mathbf{f}_\mathbf{s}$ be the continuous labeling solution of LP.

Proposition 2. *The energy with consistent rounding is bounded above by $3E_{opt} + \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}}(\|\mathbf{m}_\mathbf{p} - \mathbf{f}_\mathbf{p}\| + \|\mathbf{m}_\mathbf{q} - \mathbf{f}_\mathbf{q}\|)$, where E_{opt} is the energy of the optimum solution.*

Consistent rounding behaves better in preserving the adjacency of labels while we shrink the trust regions, and can thus allow *faster trust region shrinking*. In the next section, we set out a successive convexification scheme to improve the linear programming approximation.

2.3 Successive Relaxation Method

Here, we propose a successive relaxation method to solve the nonlinear optimization problem by constructing linear programs recursively based on the previous searching result and gradually shrinking the matching trust region for each site systematically.

Assume $\mathcal{B}^n_\mathbf{s}$ is the basis label set for site $\mathbf{s}$ at stage n linear programming. The trust region $\mathcal{U}^n_\mathbf{s}$ of site $\mathbf{s}$ is determined by the previous relaxation solution $\mathbf{f}^{n-1}_\mathbf{s} = (f^{n-1}_{\mathbf{s},1}, f^{n-1}_{\mathbf{s},2})$ and a trust region diameter d_n. We define $\mathcal{Q}^n_\mathbf{s} = \mathcal{L}_\mathbf{s} \cap \mathcal{U}^n_\mathbf{s}$: the reduced target point space falling within the current diameter. The nth basis $\mathcal{B}^n_\mathbf{s}$ is specified by

$$\begin{aligned}\mathcal{B}^n_\mathbf{s} = \{&\text{the vertex coordinates of the lower convex hull of} \\ &\{(\phi_1(\mathbf{j}), \phi_2(\mathbf{j}), c(\mathbf{s},\mathbf{j})), \forall \mathbf{j} \in \mathcal{Q}^n_\mathbf{s}\}\},\end{aligned}$$

where $c(\mathbf{s},\mathbf{j})$ is the cost of assigning label $\mathbf{j}$ to site $\mathbf{s}$.

Algorithm 1. *Successive Convexification Linear Programming*

```
Set $n = 0$; Set initial (maximum) diameter = $d_0$;
FOREACH($\mathbf{s} \in S$)
    Calculate the cost function $\{c(\mathbf{s},\mathbf{j}), \forall \mathbf{j} \in \mathcal{Q}^0_\mathbf{s}\}$;
    Convexify $\{(\phi_1(\mathbf{j}), \phi_2(\mathbf{j}), c(\mathbf{s},\mathbf{j}))\}, \mathbf{j} \in \mathcal{Q}^0_\mathbf{s}$ and find basis $\mathcal{B}^0_\mathbf{s}$;
Construct and solve $\mathcal{LP}_0$;
WHILE ($d_n \ge d_{min}$)
    $n \leftarrow n+1$;
    $d_n = d_{n-1} - \delta_n$;
    FOREACH($\mathbf{s} \in S$)
        IF ($\mathcal{Q}^n_\mathbf{s}$ is empty) $\mathcal{Q}^n_\mathbf{s} = \mathcal{Q}^{n-1}_\mathbf{s}$; $\mathcal{U}^n_\mathbf{s} = \mathcal{U}^{n-1}_\mathbf{s}$;
        ELSE update $\mathcal{U}^n_\mathbf{s}$, $\mathcal{Q}^n_\mathbf{s}$;
            Re-convexify $\{(\phi_1(\mathbf{j}), \phi_2(\mathbf{j}), c(\mathbf{s},\mathbf{j}))\}, \mathbf{j} \in \mathcal{Q}^\mathbf{n}_\mathbf{s}$
            and redetermine basis $\mathcal{B}^n_\mathbf{s}$;
    Construct and solve $\mathcal{LP}_n$;
Output $\mathbf{f}_\mathbf{s}, \forall \mathbf{s} \in S$;
```

We use an *anchor* to control the trust region for the next iteration. The trust region for one feature point is a rectangular area in the target image. We keep the anchor in

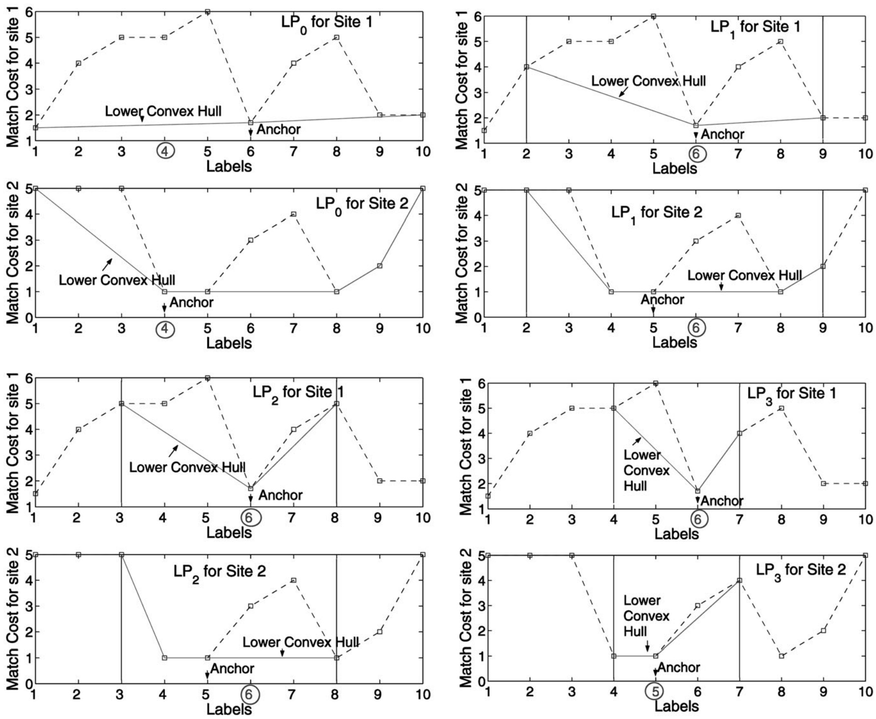


Fig. 4. SC-LP in 1D. Labels in circles are LP continuous solutions.

the new trust region for each site and shrink the boundaries inward. If the anchor is on a boundary of the previous trust region, other boundaries are moved inward. We require that new anchors have energy not greater than the previous estimation: The anchors are updated only if new ones have smaller energy. A simple scheme is to select anchors as the solution of the previous LP. Unfortunately, in the worst case, this simple scheme has solutions whose objective function is arbitrarily far from the optimum. So, instead, we use the consistent rounding solution as presented in the last section for obtaining the anchor. This iterative procedure guarantees that the objective function of the proposed multistep scheme is at least as good as a single relaxation scheme. In the following example, we use a simple 1D labeling problem to illustrate the solution procedure.

Example 2 (A 1D problem). Assume there are two sites {1, 2} and, for each site, the label set is $\{1 \dots 10\}$. The objective function is

$$\min_{\{f_1, f_2\}} \{c(1, f_1) + c(2, f_2) + \lambda |f_1 - f_2|\}.$$

In this example, we assume that

$$\{c(1, f_1)\} = \{1.5, 4, 5, 5, 6, 1.7, 4, 5, 2, 2\},$$

$$\{c(2, f_2)\} = \{5, 5, 5, 1, 1, 3, 4, 1, 2, 5\},$$

and $\lambda = 0.5$.

Based on the proposed scheme, the problem is solved by the 4-step LPs: $\mathcal{LP}_0$, $\mathcal{LP}_1$, $\mathcal{LP}_2$, and $\mathcal{LP}_3$.

- In $\mathcal{LP}_0$, the trust regions for sites 1 and 2 both start as the whole label space [1, 10]. Constructing $\mathcal{LP}_0$ based on the proposed scheme corresponds to solving an approximated problem in which c for site 1 and 2 are replaced by their lower convex hulls, respectively (see Fig. 4). Step $\mathcal{LP}_0$ uses convex hull basis labels {1, 6, 10} for site 1 and {1, 4, 8, 9, 10} for site 2. $\mathcal{LP}_0$ finds a solution with nonzero weights $\xi_{1,1} = 0.4$ and $\xi_{1,6} = 0.6$, $f_1 = \ 0.4 * 1 + 0.6 * 6 = 4$; and $\xi_{2,4} = 1$, and resulting continuous label LP solution $f_2 = 4$. Based on the proposed rules for anchor selection, we fix site 1 at label 4 and search for the best anchor for site 2 in [1, 10] using the nonlinear objective function. This label is 4, which is selected as the anchor for site 2. Similarly, the anchor for site 1 is 6. At this stage, the upper bound $\mathcal{E}^+ = c(1, 6) + c(2, 4) + 0.5 * |6 - 4| = 3.7$.
- Now, the trust region for $\mathcal{LP}_1$ is shrunk to [2, 9] for both of f_1 and f_2 by reducing the previous trust region diameter by a factor of 2. The solution of $\mathcal{LP}_1$ is $f_1 = 6$ and $f_2 = 6$. The anchor site is 6 for site 1 and 5 for site 2, with $\mathcal{E}^+ = 3.2$.
- Based on $\mathcal{LP}_1$, $\mathcal{LP}_2$ has new trust region [3, 8] for both f_1 and f_2 and its solution is $f_1 = 6$ and $f_2 = 6$. The anchors' positions do not change at this stage.
- $\mathcal{LP}_3$ has new trust region [4, 7] for both f_1 and f_2 and its solution is $f_1 = 6$ and $f_2 = 5$. Since LP achieves the upper bound $\mathcal{E}^+$, there is no need to further shrink the trust region and the iteration terminates. It is not difficult to verify that the configuration $f_1 = 6$, $f_2 = 5$ achieves the global minimum. Fig. 4 illustrates the successive convexification process for this example.

Interestingly, for the above example, ICM or even the GC scheme only find a local minimum if initial values are not correctly set. For ICM, if f_2 is set to 8 and the updating is from f_1, the iteration will fall into a local minimum corresponding to $f_1 = 9$ and $f_2 = 8$. The GC scheme based on α-expansion will have the same problem if the initial values of f_1 and f_2 are set to 9 and 8, respectively.

The simplex method has been found to be efficient in applications even though its worst-case complexity is exponential. The average complexity [29] of the simplex method is approximately proportional to the number of constraints and the logarithm of the number of variables. If we assume constant LP stages, an estimate of the average complexity of SC-LP is $O(|S| \cdot (\log |L| + \log |S|))$, where S is the site set and L is the target point set. Experiments also confirm that the average complexity of the proposed optimization scheme increases more slowly with the size of the label set than previous methods such as GC, whose average complexity is linear with respect to $|L|$, and BP, whose average complexity is proportional to $|L|^2$. Moreover, SC-LP only uses a basis label set that is usually much smaller than L.

2.4 Handling Outliers

As illustrated in [27], to model outliers in matching such as occluded targets, we can use the following strategy: We choose a constant cost for labeling a target point as an outlier. A proper constant would be selected larger than the cost of a true match and smaller than that for the wrong matches. If a site is recognized as an outlier, the site is assigned the constant cost and removed from the smoothing term.

So, a linear programming formulation to take into account occlusion inference is as follows:

$$\min \left\{ \begin{array}{c} \sum_{\mathbf{s} \in S, \mathbf{j} \in \mathcal{B}_\mathbf{s}} [c(\mathbf{s}, \mathbf{j})\, \xi_{\mathbf{s},\mathbf{j}} + c_o \pi_\mathbf{s}] + \\ \sum_{\{\mathbf{p},\mathbf{q}\} \in \mathcal{N}} \lambda_{\mathbf{p},\mathbf{q}} \sum_{m=1}^{2} (f^+_{\mathbf{p},\mathbf{q},m} + f^-_{\mathbf{p},\mathbf{q},m}) + \\ \sum_{\{\mathbf{p},\mathbf{q}\} \in \mathcal{N}} \eta_{\mathbf{p},\mathbf{q}} (\pi^+_{\mathbf{p},\mathbf{q}} + \pi^-_{\mathbf{p},\mathbf{q}}) \end{array} \right\},$$

$$\begin{array}{rcl} \sum\limits_{\mathbf{j} \in \mathcal{B}_s} \xi_{\mathbf{s},\mathbf{j}} + \pi_\mathbf{s} & = & 1, \forall \mathbf{s} \in S, \\ \sum\limits_{\mathbf{j} \in \mathcal{B}_\mathbf{s}} \xi_{\mathbf{s},\mathbf{j}} \phi_m(\mathbf{j} - \mathbf{s}) & = & f_{\mathbf{s},m}, \forall \mathbf{s} \in S,\, m = 1, 2, \\ f_{\mathbf{p},m} - f_{\mathbf{q},m} & = & f^+_{\mathbf{p},\mathbf{q},m} - f^-_{\mathbf{p},\mathbf{q},m}\,, \\ \forall\, \{p, q\} & \in & \mathcal{N},\, m = 1, 2, \\ \pi_\mathbf{p} - \pi_\mathbf{q} & = & \pi^+_{\mathbf{p},\mathbf{q}} - \pi^-_{\mathbf{p},\mathbf{q}}, \\ \xi_{\mathbf{s},\mathbf{j}}, \pi_\mathbf{s} & \geq & 0,\; f^+_{\mathbf{p},\mathbf{q},m}, f^-_{\mathbf{p},\mathbf{q},m}, \pi^+_{\mathbf{p},\mathbf{q}}, \pi^-_{\mathbf{p},\mathbf{q}} \geq 0, \end{array}$$

where $\pi_\mathbf{s}$ is a variable denoting whether site s is occluded and c_o is a constant cost for labeling one site as occluded. This schema has similar properties as the previous simpler formulation and the proposed successive convexification scheme can still be used to solve the extended problem.

2.5 Other Extensions of Successive Convexification

The above framework can be easily extended to an n-D case in which each label is an n-D point defined in some metric space. We can follow a procedure very similar to the 2D case to find the basic labels: We augment each label with another dimensionality of matching cost and form $(n+1)$-D vectors. We then find the lower convex hull of the $(n+1)$-D points with respect to the labeling cost axis. The basis labels correspond to the n-D subvectors of the $(n+1)$-D lower convex hull vertices. These n-D subvectors are obtained by stripping off the labeling cost element. The lower convex hull can be computed efficiently for the higher dimensional cases.

Successive convexification can also be applied to labeling problems with general convex smoothness terms. For example, a successive convex quadratic programming can be constructed for L_2 smoothness term labeling problems. The convex quadratic programming relaxation can also be constructed using only the basis labels that correspond to the lower convex hull vertices of the matching cost surfaces. The successive relaxation scheme can still be applied.

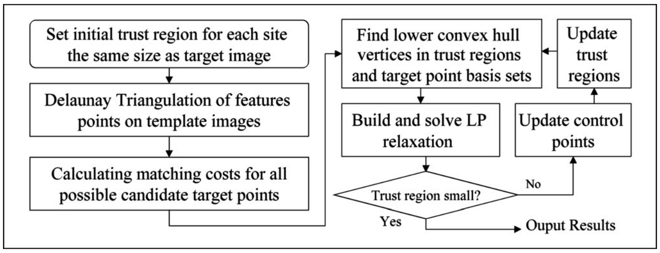

Set initial trust region for each site the same size as target image
Delaunay Triangulation of features points on template images
Calculating matching costs for all possible candidate target points
Find lower convex hull vertices in trust regions and target point basis sets
Update trust regions
Build and solve LP relaxation
Update control points
Trust region small?
No
Yes
Ouput Results

Fig. 5. Object matching using SC-LP.

The proposed scheme can also be extended to approximate metric labeling with nonconvex smoothness term. The procedure is as follows: Initially, we simply replace the nonconvex distance function, e.g., truncated L_1 distance, with a convex L_1 distance function that has a small slope and is bounded by the image dimension. When shrinking trust regions, we can choose smoothness coefficients adaptively so that the L_1 distance function approximates the nonconvex distance function in smaller trust regions. We estimate the range of the matching vector difference for each neighboring site pair when shrinking the trust regions. The slope of the L_1 distance, controlled by the coefficient $\lambda_{\mathbf{p},\mathbf{q}}$, is then adjusted to fit the nonconvex distance in the estimated range. As the trust regions become smaller, the nonconvex distance function can be more accurately approximated by the L_1 distance function. Comparison with BP using truncated L_1 smoothness term is shown in the experiment section.

2.6 Deformable Template Matching

In this section, we consider the application of the proposed general matching scheme for solving image matching problems. Fig. 5 illustrates the proposed matching method. The following discussion sets out a general image matching scheme and some aspects are not used in some specific applications.

For color images, we take as features in the template image the square pixel blocks centered on source pixels. For binary images, we first convert to grayscale via a distance transform, where the intensity of a pixel is proportional to its distance to the nearest foreground binary pixel. We define features of a binary image as the blocks in such distance transform images. In this paper, feature points are randomly selected in the source image. To reduce complexity, the feature points can be selected on the edges of the source and target images. The neighbor relation $\mathcal{N}$ is defined by the edges of the graph resulting from Delaunay triangulation of point set S on the template. For the target image, depending on the application, all of the pixels in the searching window or all the edge pixels in the target image are used in the matching process.

We often need to test the degree of matching, for example, in selecting the best template in multitemplate matching. The following quantities are defined to measure the difference between the template and the matching object: First, we define measure P to be the average of pairwise length changes from the template to the target. To compensate for any global deformation, a global affine transform $\mathcal{A}$ is first estimated based on the matching and

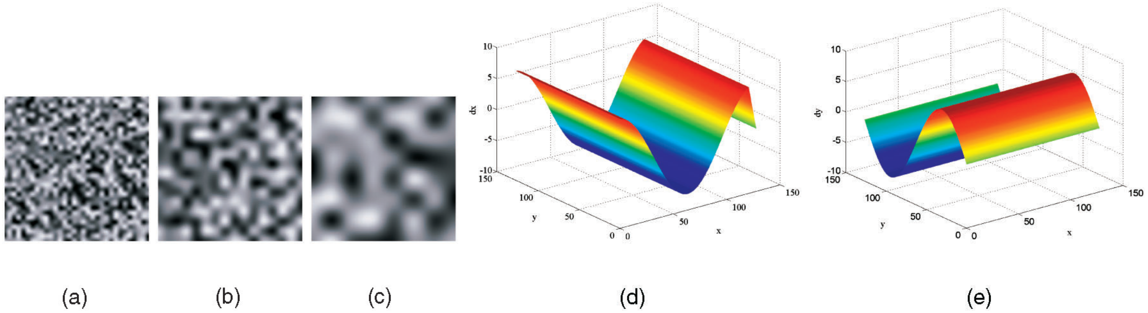


Fig. 6. Random gray-level images in matching. (a), (b), and (c) Three template random images in the three scales used. (d) and (e) Deformation model in the x and y direction.

then applied to the template points before calculating P. The length changes are further normalized by the template edge lengths. The second measure is the average warped template matching cost M, which is taken to be the difference between the target image or distance transformation and the warped reference image or distance transformation in the region of interest. The warping is based on a cubic spline. The total matching cost is simply defined as $M + \alpha P$, where α typically has a value from 0.1 to 0.5. Experiments show that only about 100 randomly selected feature points are needed in calculating P and M.

In object matching applications, target objects usually undergo scale and rotation changes. To simplify the problem, we sequentially estimate a global transformation $\mathcal{G}$ and a local deformation $\mathcal{D}$. $\mathcal{G}$ can be estimated based on LP in the largest trust region. Once $\mathcal{G}$ is fixed, the problem is reduced to the metric labeling addressed in the last section and we can apply the proposed SC-LP scheme to solve for $\mathcal{D}$. We limit $\mathcal{G}$ to rotation and scaling. We found that very sparse quantization is enough for estimating global transformation $\mathcal{G}$.

3 Experimental Results

We test the performance of the proposed matching method in applications of motion estimation, object matching, and tracking, using both synthetic ground truth data and true images. For motion estimation, target candidates are all of the image pixels in trust region for each site. For object detection and object tracking, theoretically we may still use dense pixels in trust regions because successive convexification only uses basis points in the searching process. To reduce the computational load of calculating matching costs, in object matching we restrict both the sites and target points to locating on image edges. For the above applications, the trust region for each site is a rectangular region in the target image. For each site, the target 2D points combined with the matching costs form 3D points. We compute the lower convex hull of these 3D points with respect to the cost axis. Basis labels are 2D points obtained by projecting these vertices to the xy image plane.

TABLE 1
Comparative Results for Random Pattern Matching: Scale 1

	SC-LP	GC	BP	ICM
Mean Absolute Error	0.8324	0.9460	1.0099	2.9268
Standard Deviation	0.0737	0.1618	0.1246	0.3892

3.1 Testing on Synthetic Images

3.1.1 Matching Random Patterns

As a first experiment, we randomly generate gray-scale images and then apply a ground truth deformation model to generate images for matching. In this experiment, we randomly generated 100 images in each of three different scales and then, from each of these, we also generated test images with a given deformation model. Fig. 6 shows the deformation model used in this experiment and three samples of the randomly generated images, at three different scales.

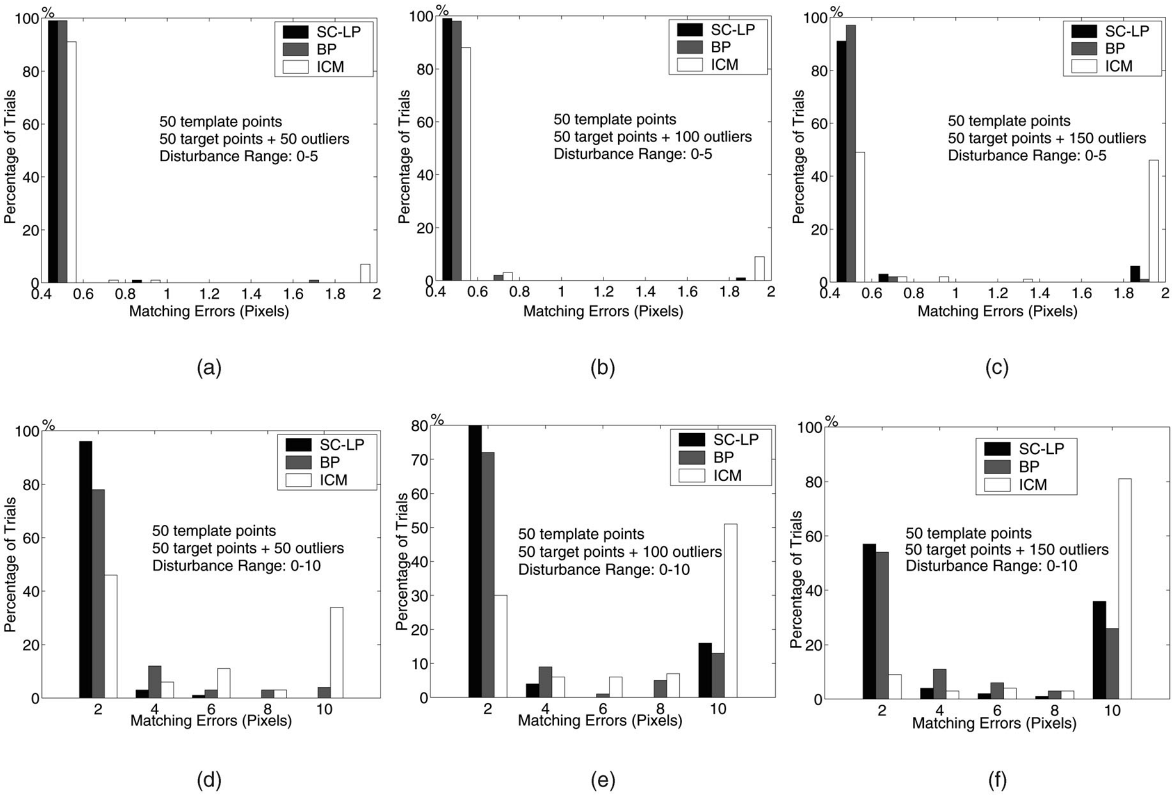
We apply the deformation model to the randomly generated images and generate target images for matching. In the reference image for each image pair, about 300 sites are randomly selected. The block size for calculating matching cost is set to be 3×3. The searching window for each site in the reference image is taken to be $[-10, 10] \times [-10, 10]$. The definition of energy for all methods compared is the same in these experiments. The smoothing coefficients are specified as unity in the matching process. We compare the matching performance of the proposed SC-LP matching scheme with ICM, the GC, and BP for random patterns with three different scales. For each of the three scales, 100 random patterns are used in the experiments. Matching error is the mean absolute distance of the matching points to the ground truth target points. Sufficiently large numbers of iterations are set for ICM, the GC, and BP such that they converge. We use α-expansion for symbol updating in the GC. BP is a baseline implementation without pruning process. The matching error and standard deviation are listed in Tables 1, 2, and 3

TABLE 2
Comparative Results for Random Pattern Matching: Scale 2

	SC-LP	GC	BP	ICM
Mean Absolute Error	0.8926	1.2452	1.1286	1.8322
Standard Deviation	0.0543	0.4343	0.1324	0.2098

TABLE 3
Comparative Results for Random Pattern Matching: Scale 3

	SC-LP	GC	BP	ICM
Mean Absolute Error	1.4954	1.8092	1.7372	2.3045
Standard Deviation	0.2308	0.3230	0.3132	0.3943



(a)

(b)

(c)

(d)

(e)

(f)

Fig. 7. Histograms of errors using SC-LP, BP, and ICM for matching random dots images.

for the three scales examined. SC-LP achieves the smallest matching error and standard deviation in this experiment. BP and GC have comparable results. ICM is the worst in this experiment.

3.1.2 Matching Random Dots

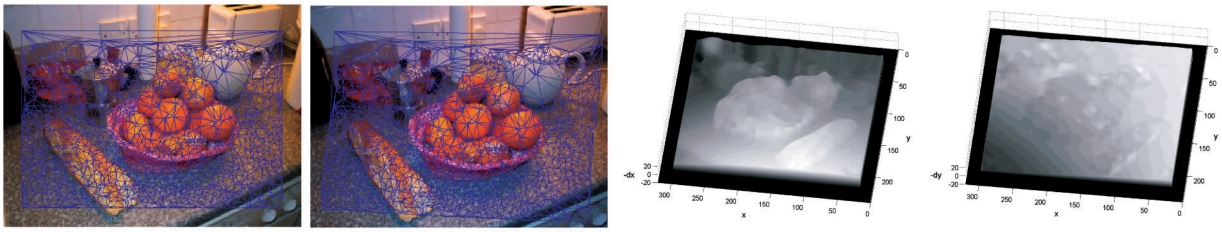
In this experiment, we compare the performance of SC-LP with BP and ICM for a binary object detection in clutter. GC has been designed for motion and stereo and, therefore, is not included in the comparison. In our experiments, the templates are generated by randomly placing 50 black dots into a 128×128 white background image. A 256×256 target image is then synthesized by randomly translating and perturbing the block dot positions from those in the template. The random perturbation ranges are set to be 5 and 10 pixels, respectively, in two experiments. 50, 100, or 150 random noise dots are then added to the target image to simulate background clutter. There are thus six situations to be tested and, in each experiment, we generate 100 template and target images. In this experiment, we match the gray-level distance transformation of the template and target images. Fig. 7 compares results using the proposed matching scheme with using BP and ICM. The histograms show the error distribution of different methods. In this experiment, all the methods use the same energy function. The SC-LP has similar performance as BP and is much better than the greedy scheme of ICM in cases of large distortion and cluttered environments. SC-LP has much less complexity than BP when the number of labels exceeds 1,000.

3.1.3 Matching Random Dots Using Nonconvex Regularization Term

In a different experiment setting for matching random dots, we use a truncated L_1 distance function in the regularization term. In addition to random disturbance of five pixels, there is a large displacement which is 30-pixel in the x direction and a shifting of five pixels in the y direction for the right half of the template points. The nonconvex smoothness term helps to preserve the discontinuity in matching. Using SC-LP, we adaptively change the smoothness coefficients so as to approximate a truncated L_1 distance function which has a linear part with unity slope and a turning point at 15 pixels. The random dot template is composed of 100 points and 50 noise points are added to the target image. We repeat the experiment for 100 times. Different methods in this experiment use the same energy function. The mean matching errors and standard deviations of the testing methods are shown in Table 4. The proposed scheme and BP yield much better results than the greedy methods.

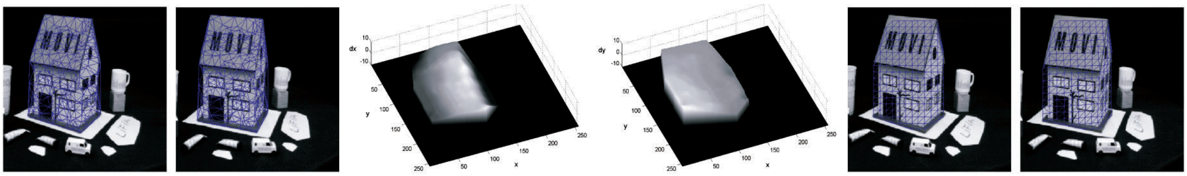
TABLE 4
Comparative Results for Random Dot Matching with Large Discontinuities

	SC-LP	BP	ICM
Mean Absolute Error	6.3227	7.4985	56.8322
Standard Deviation	5.5764	7.4985	12.6827



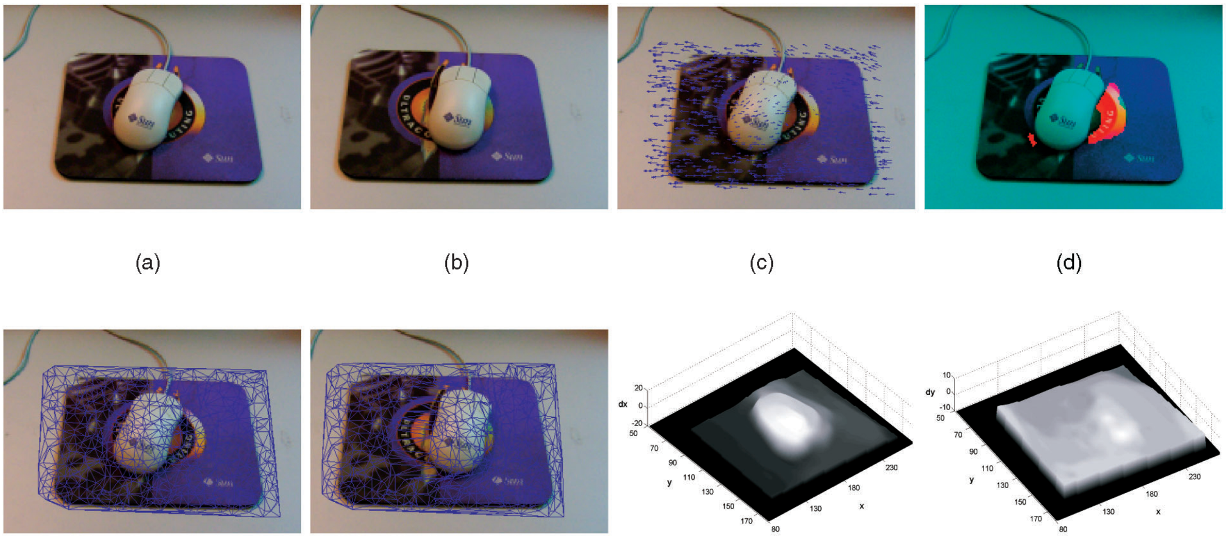
(a) (b) (c) (d)

Fig. 8. Motion estimation for image `Table_Top`. (a) Reference image and mesh. (b) Matching image and matching mesh based on SC-LP. (c) and (d) Dense x-motion and y-motion.



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

Fig. 9. Motion estimation for image `Toy_house`. (a) Reference image and mesh. (b) Matching image and matching mesh based on SC-LP method. (c) Dense x motion. (d) Dense y motion. (e) and (f) Regular mesh matching result based on the dense motion field.



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

(e) (f) (g) (h)

Fig. 10. Motion and occlusion estimation for image `Mouse`. (a) and (b) Reference and matching images. (c) Scaled motion vectors. (d) Occlusion map shown in the red channel. (e) and (f) Matching based on SC-LP. (g) and (h) Dense x-motion and y-motion.

3.2 Motion Estimation

We also test the proposed SC-LP method for estimating motion in a large searching range. Fig. 8 illustrates a motion estimation result for the image `Table_Top`. About 2,000 edge-points and supporting points are detected in the region of interest. Delaunay triangulation is applied to the point set and points connected by the edges of the Delaunay graph are defined as neighbor site pairs. The searching region is $[-20, 20]$ in both the x and y directions. The cost function used in the experiment is the normalized image block absolute difference. Sparse matching and dense matching based on the proposed SC-LP are shown in Figs. 8a and 8b and in Figs. 8c and 8d. Dense matching is by upgrading the sparse matching result with a detail preserving PDE method [27]. More experimental results, using image `Toy_house`, are shown in Fig. 9. To illustrate the accuracy of dense matching, the regular mesh matching result is shown in Figs. 9e and 9f.

Fig. 10 illustrates experiment results for estimating motion and the occlusion map simultaneously, with the SC-LP scheme. For image `Mouse`, camera motion and object motion are both involved. Another challenge in this experiment is that the mouse and mouse pad contain large areas without texture. The scaled motion vector plot is shown in Fig. 10c. A threshold of 0.5 is used to obtain the occlusion map in these experiments. In Fig. 10d, the occlusion map is shown aligned with the reference image by replacing the red channel of the reference image with the occlusion map while keeping intact the green and blue

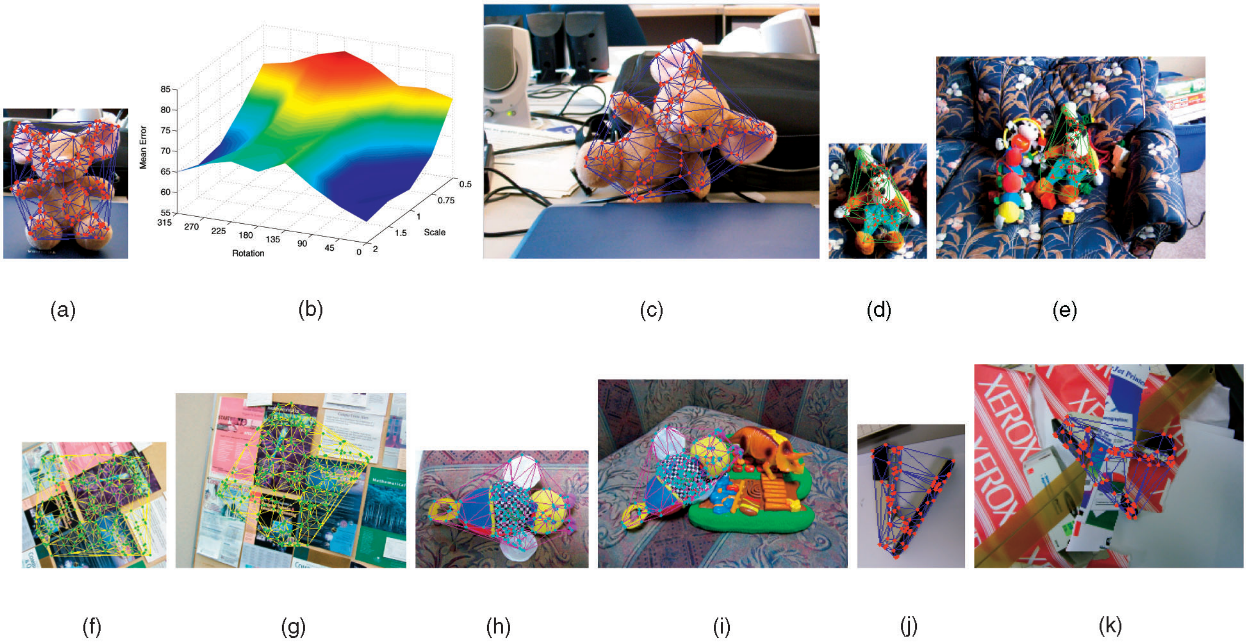


Fig. 11. Color object localization. (a) Template mesh. (b) Interpolated mean matching cost surface for $\mathcal{G}$: $\theta^* = 45°$, $\kappa^* = 1.5$. (c) Matching result with the SC-LP method. (d), (f), (h), and (j) Four other template meshes. (e), (g), (i), and (k) Match results.

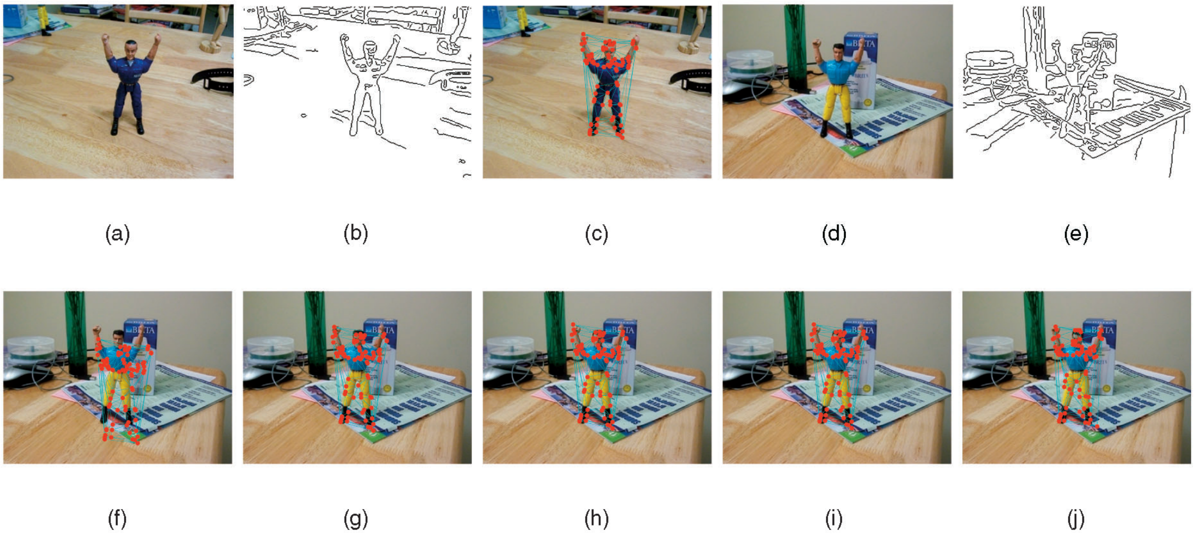


Fig. 12. Binary object localization. (a), (b), and (c) Template image, edge map, and template mesh. (d) and (e) Target image and edge map. (f), (g), (h), (i), and (j) Different stages of SC-LP matching.

channels. Figs. 10e, 10f, 10g, and 10h show the matching result based on the proposed linear programming scheme.

3.3 Object Detection

Color object matching examples are shown in Fig. 11. Fig. 11a shows the template mesh for a toy, with feature points randomly selected on the edge map of the object. In this experiment, the global transform is first estimated in the discretized global transform space with SC-LP in the largest trust region. Rotation is sampled between 0° and 360° at 45° intervals. The scale is sampled in [0.5, 2] with quantization levels 0.5, 0.75, 1, 1.5, and 2. The interpolated mean matching cost surface is shown in Fig. 11b, with optimal rotation angle θ and scale κ of 45° and 1.5, respectively. Local deformation is further estimated by SC-LP method. The final matching result is shown in Fig. 11c. In Figs. 11d and 11e, a toy dog is matched in a very cluttered background. This is a difficult target in that there are many features in the background similar to those on the toy. The proposed method finds the correct global transform and the correct matching. Further results for color object matching are shown in Figs. 11g, 11i, and 11k. Figs. 11j and 11k show matching where large occlusion is involved. Fig. 12 shows an example for binary object matching using the distance transform image patches as features. In this example, SC-LP performs well in strong background clutter.

As a different test of the usefulness of the proposed scheme for object detection, we carried out a face detection experiment using the Caltech database. Each tested image contains one face and we test the chance of SC-LP correctly

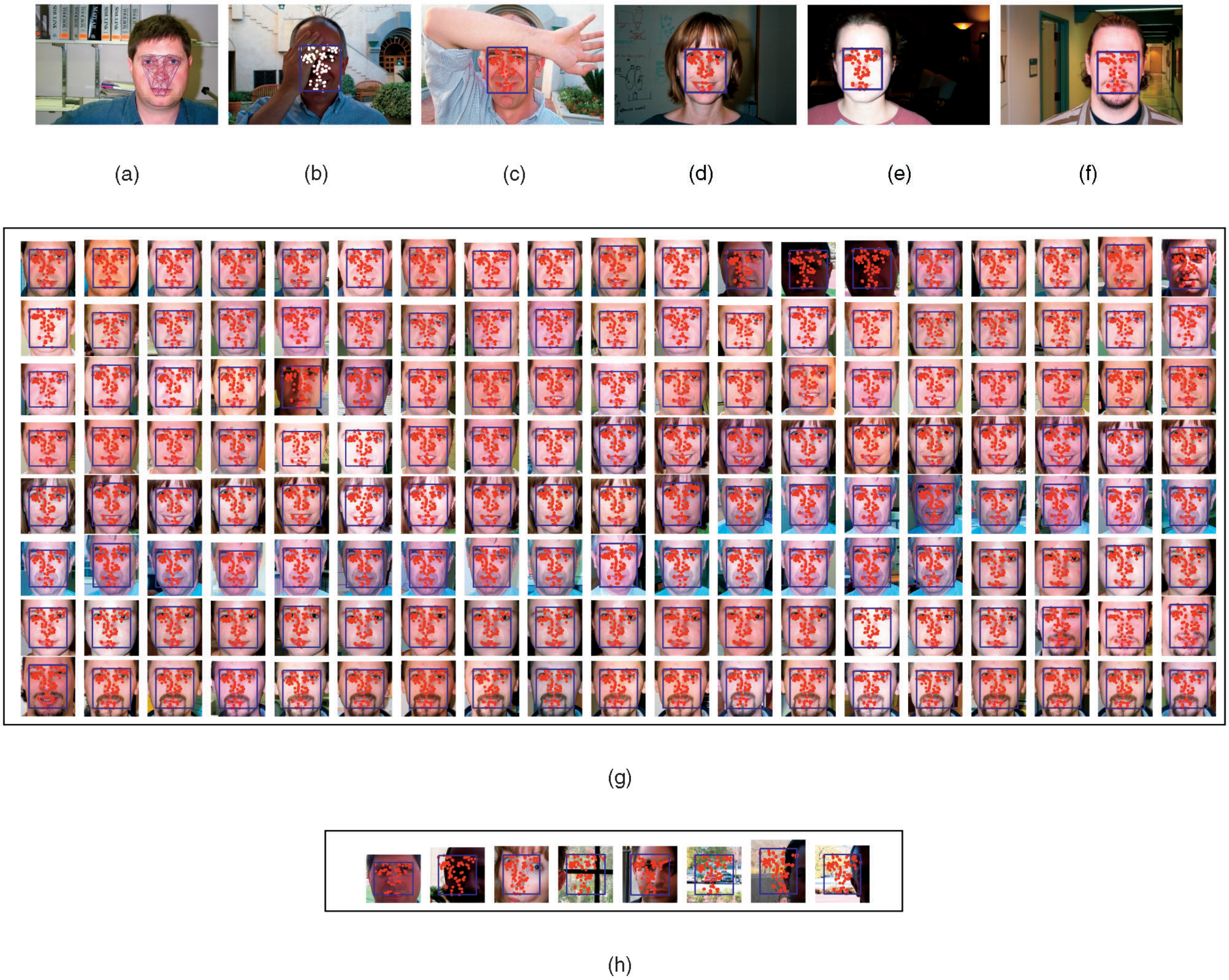


Fig. 13. Color face template matching. (a) Template. (b), (c), (d), (e), and (f) Five example targets—bounding box for match is shown. More correct matches are shown in (g): detail block near bounding box automatically cropped from the target image is shown. (a) Template. (b) Example 1. (c) Example 2. (d) Example 3. (e) Example 4. (f) Example 5. (g) Samples of correct matches. (h) Incorrect matches (eight faces).

locating the object in images. After removing several small repeated images and drawings, these comprise a total of 431 test images. One randomly chosen image is used as a template, testing on the others. We chose to match only the chromaticity of the images, which is more illumination invariant but also has more ambiguities. An ICM-based scheme has a correct matching ratio of 41 percent, while the proposed scheme yields 98.1 percent correct. The matching faces, showing the bounding box of the match, are displayed in Fig. 13. We also experimented on matching car-back images from the Caltech database. We use a randomly chosen image as the template and try to locate the object in other images, based on the gray-scale distance transform of the Canny edge maps. Of 119 cars, the matching precision is 82.4 percent. We show some matching results in Figs. 14g and 14h. Simple ICM schemes work poorly, with only a few correct matches. Another object detection result is shown in Fig. 15, matching leaves. In this experiment, we again use the distance transform of the edge maps for matching. We use six templates, with success for 181 targets out of 185 matching tests. Object detection results are summarized in Table 5.

3.4 Object Tracking

Apart from object localization in still images, we also tested the proposed scheme for object tracking in videos. We use multiple templates in this test. To increase efficiency, templates are further organized into a graph structure in which neighbors represent possible template transitions. When an object changes drastically in a video sequence, it makes sense to test whether switching to another template would produce a better match, likely keeping to the current scale and rotation if possible in applications when several templates are available. An efficient approach is to test against the current template and its neighbors in a template selection process. The template with the lowest matching cost is chosen, its matching result is recorded, and the rotation θ and scale κ are updated based on a smooth-motion object-following model. Fig. 16 shows a car location result with the first frame as the template. The proposed scheme successfully tracks the back of the car in a long video sequence. Fig. 17 shows an object tracking result in which two exemplars are used—the hand undergoes dramatic shape changes between the two gestures. There are also large scale and rotation changes of the hand involved in this sequence. The proposed

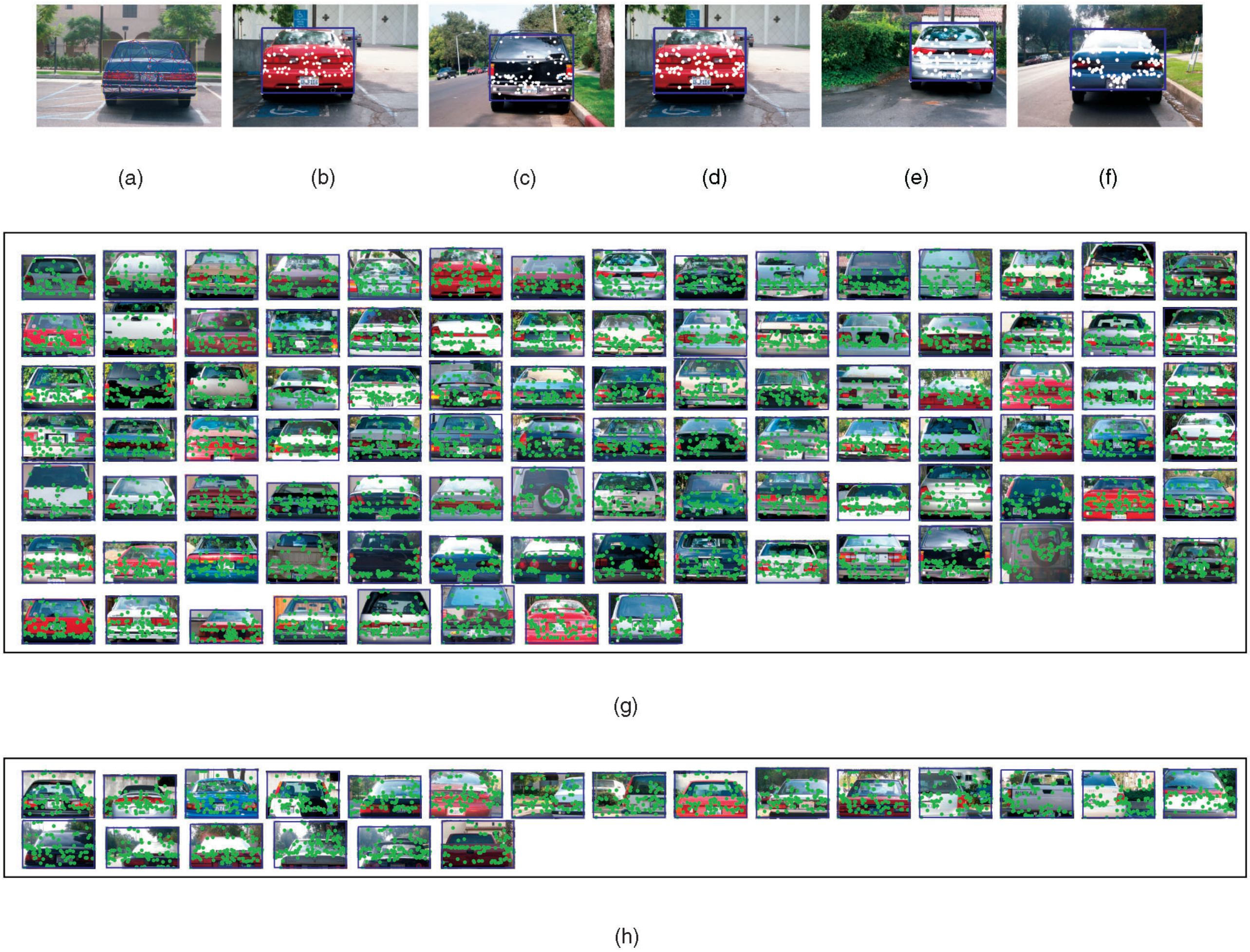


Fig. 14. Matching carbacks in clutter. In (g) and (h), blocks shown are targets automatically cropped from the target images. (a) Template. (b) Example 1. (c) Example 2. (d) Example 3. (e) Example 4. (f) Example 5. (g) Correct matches (98 of 119 cars). (h) Incorrect matches (21 cars).

scheme successfully tracks the movement of the hand. As another example of this approach, Fig. 18 shows a result for tracking a walking person, using three exemplars. The posture of the person walking in the scene is accurately recovered. The template follows the object successfully in this difficult, very complex-background setting. Currently, the tracking is not realtime. We select about 100 features points are on the template. Matching each frame takes about 10s using a 2.6 GHz PC with three LP iterations. About 25 percent of the computation power is spent on calculating the matching costs. For the same problem, BP is several orders slower. ICM is efficient but is not reliable for tracking.

3.5 Running Time Comparison

For large label set matching in which we need to search a whole target image, the proposed scheme has a clear advantage. Table 6 shows typical running times for different methods for matching leaf images in the Caltech database. In this example, the template image has 106 sites, all of the 4,622 edge pixels in the target image (320×211) are potential matching candidates. We compare the running time of the proposed scheme with GC and BP. SC-LP, GC, and BP all successfully match the target in four iterations. As Table 6 shows, SC-LP is more efficient than GC and BP for matching with a large number of labels. The searching window for graph cut is set to 120×120 for each site. ICM fails to locate the target, even though it converges in less than one second.

BP can be sped up by embedding displacement vectors into grids and computing function lower envelop with a method similar to distance transform [10]. This scheme has linear complexity over the number of labels (displacement vectors). But, it has to trade time complexity with high spatial complexity. Multiresolution schemes can also be used to reduce the complexity of BP for large label set problems. Instead of directly matching all of the target labels, we can group multiple target points into coarser labels and subdivide the groups iteratively for matching refinement. The multiscale method speeds up matching, but, at the same time it may also degrade the matching result. Our experiments show that, when the target image's resolution is reduced by more than eight times, the result degrades rapidly. Because of this constraint, multiscale BP is still much slower than the proposed method.

4 Conclusion

In this paper, we present a robust linear programming-based matching scheme—linear programming with successive convexification (SC-LP). This method uses a small basis to

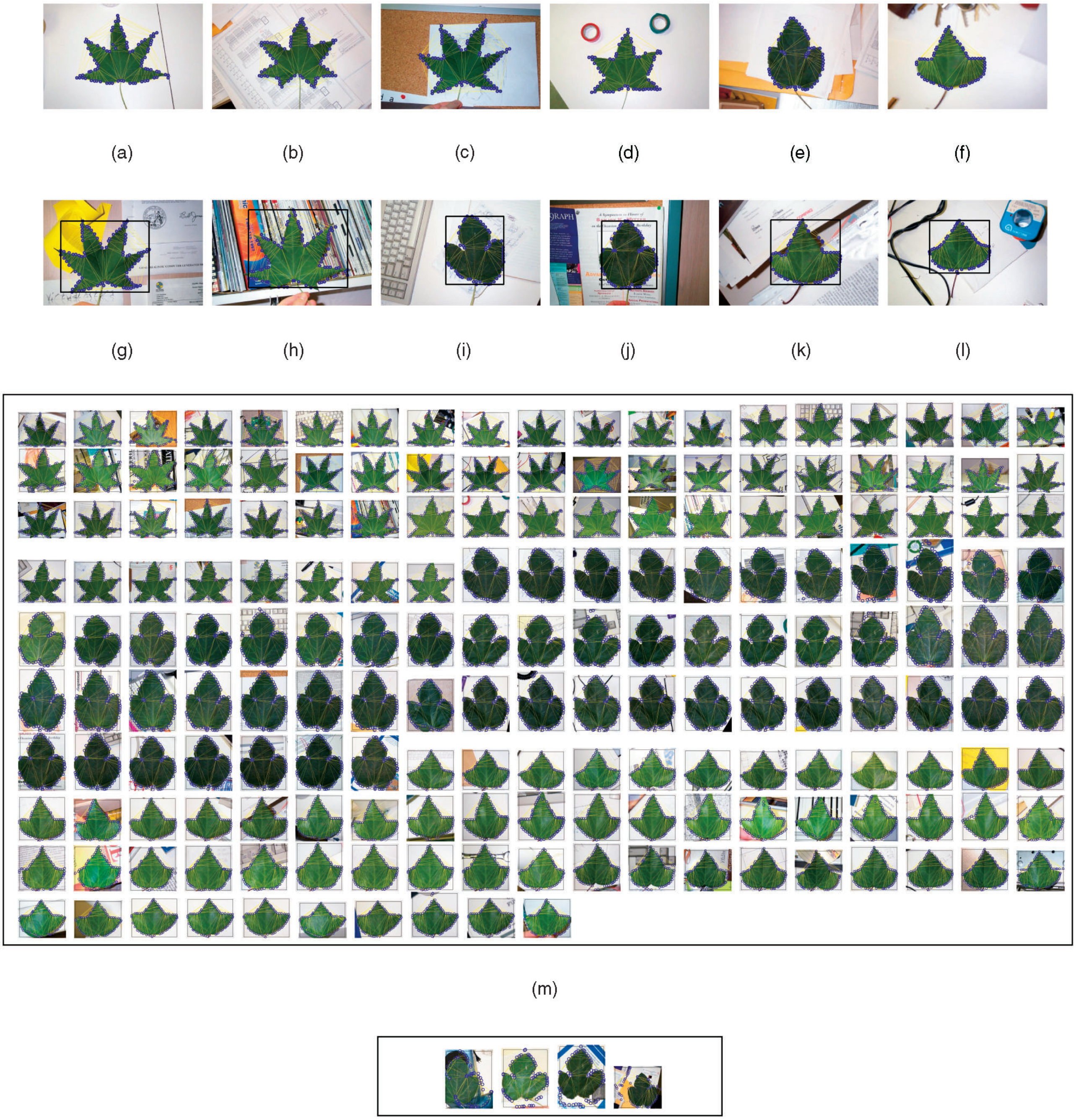


Fig. 15. Matching result of leaves. (a), (b), (c), (d), (e), and (f) Templates. (g), (h), (i), (j), (k), and (l) Matching examples. (m) Correct matching blocks automatically cropped from the target images. (n) Incorrect matching blocks cropped from target images.

represent the matching space for each feature point on the template, which allows the method to be applied to problems with a very large number of target candidates. This linear programming relaxation has many useful properties and is a strong approximation scheme. We further propose a successive relaxation scheme to improve the matching iteratively. We have applied the optimization method to solving metric labeling problems with the L_1 regularization term, and this method is found to be able to find the correct matching with high probability. The successive convexification idea can also be generalized to solve problems with

TABLE 5
Object Detection Using Caltech Face, Carback, and Leaf Database

Database	Faces	Car-backs	Leaves
Success Rate	98.1%	82.4%	97.8%
	(1 template)	(1 template)	(6 templates)

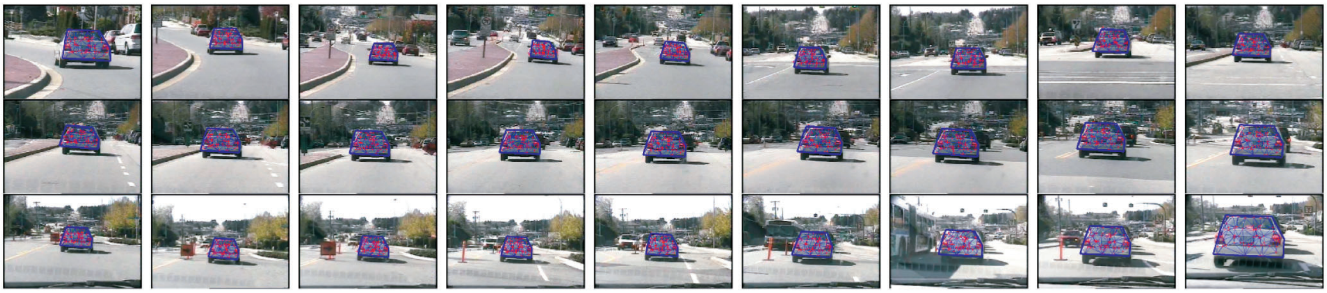


Fig. 16. Tracking result for car sequence. Selected from 1,000 frames.

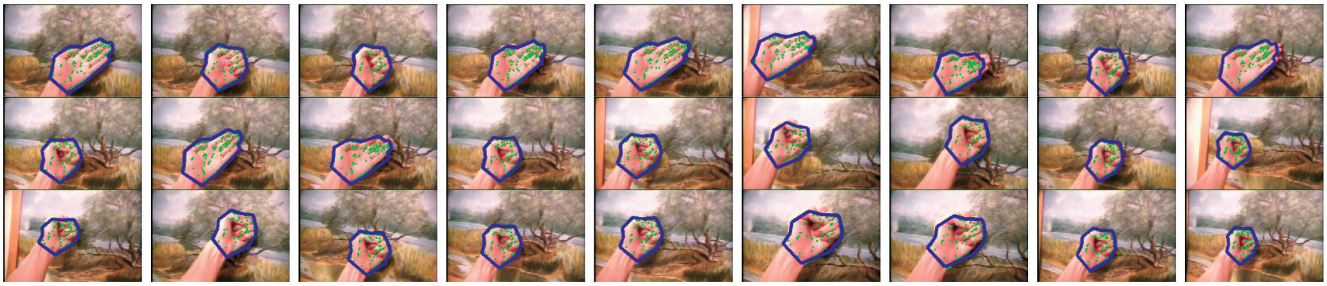


Fig. 17. Tracking result for hand sequence. Selected from 500 frames.

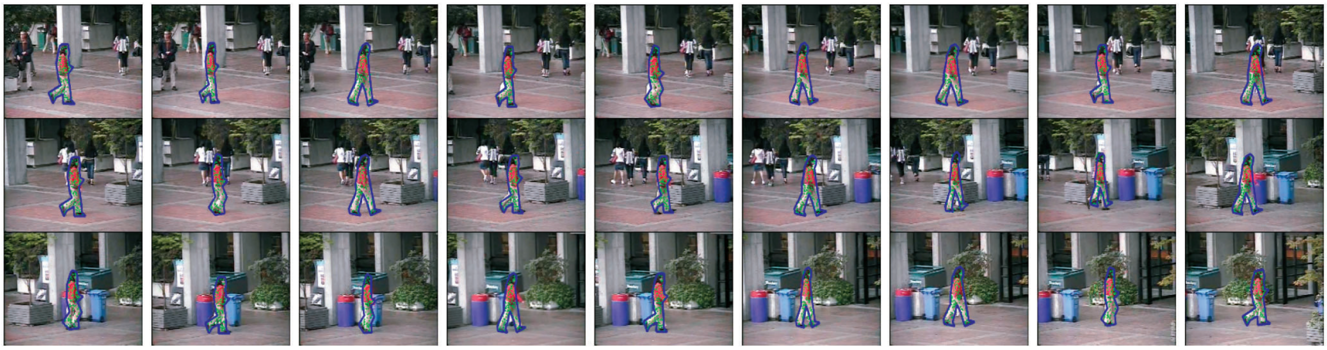


Fig. 18. Tracking result for walking sequence. Selected from 150 frames.

other convex smoothness terms, i.e., an L_2 norm smoothness term. We also propose a method to approximate nonconvex smoothness term problems using successive convexification. As an illustration of the power of the method, we discussed motion estimation, several object location applications, and an object tracking framework in which multiple templates can be used. By choosing the closest template, the proposed scheme can be used to robustly locate objects that change appearance dramatically, using only a few templates. Experiments demonstrate robust object matching results in cluttered environments.

TABLE 6
Comparative Results of Typical Running Times in One Iteration (Matching Leaf Image)

	SC-LP	GC	BP
Running time	1 sec	29 secs	56 mins

References

[1] P.F. Felzenszwalb, "Representation and Detection of Deformable Shapes," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 102-108, 2003.

[2] H. Ishikawa, "Global Optimization Using Embedded Graphs," PhD dissertation, New York Univ., May 2000.

[3] S. Roy and I.J. Cox, "A Maximum Flow Formulation of the N-Camera Stereo Correspondence Problem," *Proc. Int'l Conf. Computer Vision,* pp. 429-499, 1998.

[4] T.M. Breuel, "A Comparison of Search Strategies for Geometric Branch and Bound Algorithms," *Proc. European Conf. Computer Vision,* pp. 837-850, 2002.

[5] A. Rosenfeld, R.A. Hummel, and S.W. Zucker, "Scene Labeling by Relaxation Operations," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 6, no. 6, pp. 420-433, 1976.

[6] J. Besag, "On the Statistical Analysis of Dirty Pictures," *J. Royal Statistical Soc. London, B,* vol. 48, pp. 259-302, 1986.

[7] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, pp. 1222-1239, 2001.

[8] J. Pearl, *Probabilistic Reasoning in Intelligent Systems—Networks of Plausible Inference.* Morgan-Kaufmann, 1988.

[9] 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, pp. 723-735, 2001.

[10] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient Belief Propagation for Early Vision," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 261-268, 2004.

[11] V. Kolmogorov and R. Zabih, "Multi-Camera Scene Reconstruction via Graph Cuts," *Proc. European Conf. Computer Vision,* 2002.

[12] V. Kolmogorov and R. Zabih, "Computing Visual Correspondence with Occlusions Using Graph Cuts," *Proc. Int'l Conf. Computer Vision,* pp. 508-515, 2001.

[13] Y. Boykov and V. Kolmogorov, "Computing Geodesics and Minimal Surfaces via Graph Cuts," *Proc. Int'l Conf. Computer Vision,* p. 26, 2003.

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

[15] J.M. Coughlan and S.J. Ferreira, "Finding Deformable Shapes Using Loopy Belief Propagation," *Proc. European Conf. Computer Vision,* vol. 3, pp. 453-468, 2002.

[16] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *J. Chemical Physics,* vol. 21, no. 6, pp. 1087-1092, 1953.

[17] H. Chui and A. Rangarajan, "A New Algorithm for Non-Rigid Point Matching," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 44-51, 2000.

[18] S. Gold and A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 4, pp. 377-388, Apr. 1996.

[19] P.D. Tao, T.Q. Phong, R. Horaud, and L. Quan, "Stability of Lagrangian Duality for Non Convex Quadratic Programming Solution Methods and Applications to Computer Vision," *Math. Modelling and Numerical Analysis,* vol. 31, no. 1, pp. 57-90, 1997.

[20] X. Bai, H. Yu, and E. Hancock, "Graph Matching Using Embedding and Semidefinite Programming," *Proc. British Machine Vision Conf.,* 2004.

[21] M. Ben-Ezra, S. Peleg, and M. Werman, "Real-Time Motion Analysis with Linear Programming," *Proc. Int'l Conf. Computer Vision,* pp. 703-709, 1999.

[22] J. Kleinberg and E. Tardos, "Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields," *Proc. IEEE Symp. Foundations of Computer Science,* pp. 14-23, 1999.

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

[24] E. Wasserstrom, "Numerical Solutions by the Continuation Method," *SIAM Rev.,* vol. 15, pp. 89-119, 1973.

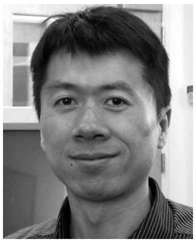
[25] A. Blake and A. Zisserman, *Visual Reconstruction.* MIT Press, 1987.

[26] M.S. Bazaraa, J.J. Jarvis, and H.D. Sherali, *Linear Programming and Network Flows.* Wiley Intersicence, 2005.

[27] H. Jiang, Z.-N. Li, and M.S. Drew, "Optimizing Motion Estimation with Linear Programming and Detail-Preserving Variational Method," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 738-745, 2004.

[28] H. Jiang, M.S. Drew, and Z.-N. Li, "Linear Programming Matching and Appearance-Adaptive Object Tracking," *Proc. Energy Minimization Methods in Computer Vision and Pattern Recognition,* pp. 203-219, 2005.

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



Hao Jiang received the PhD degree in computer science from Simon Fraser University, Vancouver, Canada. He is now a postdoctoral researcher at the University of British Columbia. His research interests include computer vision, image and video processing, AI, and computer graphics. He is a member of the IEEE.

Mark S. Drew is an associate professor in the School of Computing Science at Simon Fraser University in Vancouver, Canada. His background education is in engineering science, mathematics, and physics. His interests lie in the fields of multimedia, computer vision, image processing, color, photorealistic computer graphics, and visualization. He has published more than 100 refereed papers in journals and conference proceedings. Dr. Drew is the holder of a US patent in digital color processing and a US patent application in color computer vision. He is a member of the IEEE.

Ze-Nian Li received the BSc degree in electrical engineering from the University of Science and Technology of China and the MS and PhD degrees in computer science from the University of Wisconsin-Madison. He was an assistant professor at the University of Wisconsin-Milwaukee from 1986 to 1987. In 1988, he joined the School of Computing Science at Simon Fraser University, where he is currently a professor and director of the Vision and Media Lab. His current research interests include computer vision, pattern recognition, multimedia, image processing, and artificial intelligence. He is a member of the IEEE.