Deformed Lattice Discovery Via Efficient Mean-Shift Belief Propagation

Minwoo Park[1], Robert T. Collins[1], and Yanxi Liu[1,2]

[1] Department of Computer Science and Engineering
[2] Department of Electrical Engineering
The Pennsylvania State University, University Park, PA 16802
{mipark,rcollins,yanxi}@cse.psu.edu

Abstract. We introduce a novel framework for automatic detection of repeated patterns in real images. The novelty of our work is to formulate the extraction of an underlying deformed lattice as a spatial, multi-target tracking problem using a new and efficient Mean-Shift Belief Propagation (MSBP) method. Compared to existing work, our approach has multiple advantages, including: 1) incorporating higher order constraints early-on to propose highly plausible lattice points; 2) growing a lattice in multiple directions simultaneously instead of one at a time sequentially; and 3) achieving more efficient and more accurate performance than state-of-the-art algorithms. These advantages are demonstrated by quantitative experimental results on a diverse set of real world photos.

1 Introduction

Repeated patterns, presented as imagery or otherwise, provide fundamental cues for both human and machine perception [1,2]. At one extreme, mathematicians have generalized these concepts into perfectly periodic patterns generated by a single fundamental region (lattice unit) under a finite number of crystallographic groups [3,4]. In the computer vision and computer graphics communities, on the other hand, these patterns are usually regarded as *textures* with a random nature, composed of deformed versions of one or more basic *texture elements* [5,6,7,8].

Ample evidence can be found that repeated textures are not merely random collections of texture elements (Figures 4 and 5). Instead, these textures often exhibit geometric, topological and statistical regularities [7], especially those texture patterns that originate from human designs or from nature and biology. Using wallpaper texture patterns [9] as an example, all texture elements are related to each other by a pair of linearly independent vectors t_1, t_2, or composition of them, that are the shortest basis vectors [3] forming the boundary of the respective *lattice units*. We fully acknowledge the importance of a robust lattice detection tool for automatic discovery and quantification of texture regularity in real images.

Instead of treating texture elements as isolated individuals, the underlying topological lattice structure of a near-regular texture (NRT) was first introduced by Liu et al for texture analysis and manipulation [10,11,7]. Subsequently, Hays et al [12] developed the first deformed lattice detection algorithm for real images without pre-segmentation; and Lin and Liu [13,14] developed the first deformed lattice tracking algorithm for dynamic NRTs.

The idea behind [12] is simply to look for the t_1, t_2 neighbors of a randomly selected interest point in the image. If a sufficient number of such points look like their respective t_1, t_2 neighbors (lower order similarity) and also share their t_1, t_2 neighbors' directions/orientations (higher order correspondences) with other interest points in the image, those points and their neighboring relations are confirmed to be part of the final lattice. With the found correspondence, the slightly deformed lattice can be straightened out and a new round of lattice discovery begins, so the extracted lattice grows bigger and bigger. Formulating the lattice detection problem as a higher order correspondence problem adds computational robustness against geometric distortions and photometric artifacts in real images. However, although Hays et al [12] produces impressive results, there are several serious drawbacks preventing its wider applicability. First, local correlation-based peak finding is used as a last resort for finding points of interest, which is both time consuming and sensitive to noise, occlusion and transform discontinuity in the image (Figures 4 and 5). Second, the method is based on finding the eigenvalues of a $n^2 \times n^2$ sparse matrix (n is the number of potential texture elements), which is cumbersome computationally. Third, the algorithm only examines one of the t_1 and t_2 vectors at a time, and is thus less robust against misleading repetitions and prone to wasting time on interest points that do not lead to legitimate t_1, t_2 neighbors. Our proposed method overcomes these weaknesses.

Our work is partially inspired by Lin and Liu [13,14], who treat lattice detection as a spatial tracking problem. Different from [13,14], we do not require an initial texton to be given (by the user) nor do we use affine template matching for the spiral, outward growth of the deformed lattice. Instead, we propose to formulate the detection of the underlying deformed lattice in an unsegmented image as a spatial, multi-target tracking problem, using a recently published, fast, Mean-shift Belief Propagation (MSBP) method [15]. The MSBP algorithm only examines the values of the belief surface within its local kernel window, thus there is no need to generate the entire belief surface, yielding great computational savings over non-parametric belief propagation and discrete belief propagation [15]. Inferencing on a Markov Random Field (MRF) for this type of texture (NRTs) is a natural and effective choice given the topological degree-4 graph structure of the underlying lattices of deformed wallpaper patterns [12,14]. In addition, there is a perfect conceptual match between the iterative nature of seeking the underlying lattice of a repeated pattern in an image and performing mean-shift on the implicit belief surface (marginal density) generated by the belief propagation algorithm on a graphical model.

Compared with [12], our proposed approach offers significant improvements in accuracy, robustness and efficiency for automatic lattice detection by: 1. incorporating higher order constraints early-on to propose highly plausible lattice points; 2. growing the lattice along multiple directions simultaneously instead of one at a time; 3. achieving a deterministic algorithm linearly dependent on the size of the lattice structure, instead of quadratic or higher order on the number of texels found, as in [12]. Quantified experimental results on an extensive set of diverse real images demonstrate these advantages (Figures 4 and 5).

2 Lattice Fitting Via Mean-Shift Belief Propagation

The main idea underlying lattice finding using belief propagation is that identifying the location of texture elements in a repeated pattern is made easier and more robust if multiple elements are searched for jointly, rather than one at a time. This is so because the location of each element is constrained by its neighboring elements, such that finding some of them provides the knowledge constraining where the others may be. In an extreme case, if you know where each of the four neighbors of a texture element is, you have a good idea about where the central texture element should be, even if it is occluded or otherwise hard to find. The key to leveraging this insight is to encode the topological structure of texture elements explicitly so that it can be used effectively to perform inference over the joint space of spatial constraints.

A Markov Random Field (MRF) specifies a factorization of the joint distribution of a set $\mathbf{X}$ of random variables. An MRF can be represented as an undirected graph $\mathbf{G} = (\mathbf{N}, \mathbf{E})$, where each node in $\mathbf{N}$ represents a random variable in set $\mathbf{X}$ and each edge in $\mathbf{E}$ represents a statistical dependency between random variables in $\mathbf{X}$. In the present context, the value of a random variable is the location of the center of a texture element, and the spatial dependency between variables represents how one element's position constrains the positions of other elements in the pattern.

Another piece of information to be represented for each texture element is the appearance similarity of the texture elements at given image locations. Since the lattice indicates a repeated pattern, the appearance of each of the texture elements can be described by a single reference appearance model. Determining the location of an element thus requires combining these two sources of information: the compatibility of the texture elements at various image locations with what we think a texture element should look like, and the compatibility of an element location with the spatial constraints provided by other elements.

Determining the location of one particular element given estimated positions of all other elements is infeasible if it requires brute force evaluation of the marginal distribution of that single random variable's value, since that will lead to $O(n \times n^{k-1})$ computation time where k is the number of nodes in the graph and n is the size of the hidden variable space (for simplicity we can think of it as $2D$ location in the current example). Thus determining where each of the elements is in the pattern requires $O(kn^k)$ computation time.

Fortunately, the joint probability over the pattern state x and image measurement z in an MRF can be factored as

$$p(x_1, ..., x_N, z_1, ..., z_N) = \prod_{(i,j)} \psi(x_i, x_j) \prod_{s} \phi(x_s, z_s) \tag{1}$$

where ψ and ϕ are functions specifying pairwise compatibility (spatial constraints between elements) and joint compatibility (appearance similarity of elements at given image locations), respectively. The belief propagation algorithm takes advantage of this factorization to perform inference on the graph efficiently. Cost of computation for estimating the state of all texture elements is reduced from

$O(kn^k)$ to $O(kn^2)$. However, if the hidden variable state space is large, BP can still be very expensive. BP also is not feasible for hidden variable spaces with continuous values.

We have developed an efficient and effective inference method called Mean Shift Belief Propagation (MSBP) [15] that works iteratively with local samples and weights. Since mean-shift is equivalent to finding a local mode within a Parzen window estimate of a density function, we can use mean-shift as a non-parametric mode-seeking mechanism operating on weighted samples generated within the belief propagation framework. Geometrically, the MSBP process can be visualized as performing mean-shift on the implicit belief surface (marginal density) generated by the belief propagation algorithm. Since MSBP only examines the values of the belief surface within its local kernel window, there is no need to generate the entire belief surface, yielding computational savings over non-parametric belief propagation and discrete belief propagation.

3 Our Approach

Our approach for deformed lattice detection is composed of three phases (Figure 1). Phase I has two steps: interest point generation and lattice-unit proposal. Phase II is lattice inference via spatial tracking using MSBP, followed by verification of the found lattice. Phase III is regularized thin-plate spline warping, where the deformed lattice is rectified into a regular lattice. Phase III is a transition phase, from which a new round of lattice inference starts again (Figure 1).

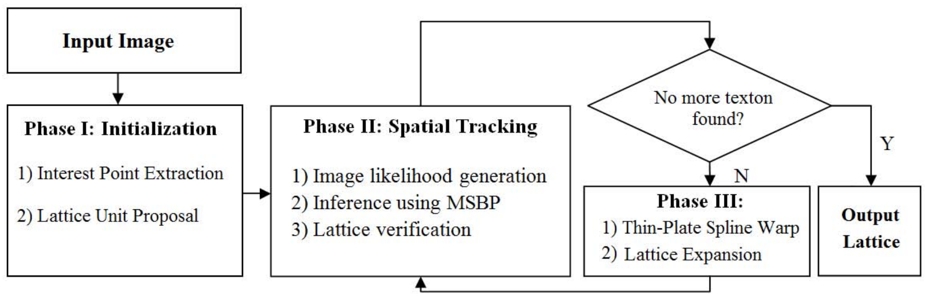

Fig. 1. Flowchart of our algorithm. There are three phases: initialization, spatial tracking and incremental warping (transition phase).

3.1 Phase I: Initialization

Interest point extraction. Our approach starts by extracting a set of interest points in the image. Any interest point detector can be used in this stage. The goal is to extract enough feature points to expose the repeated image substructures reliably without an overwhelming number of false positives for the subsequent lattice finder. For this initial implementation, we use KLT corner

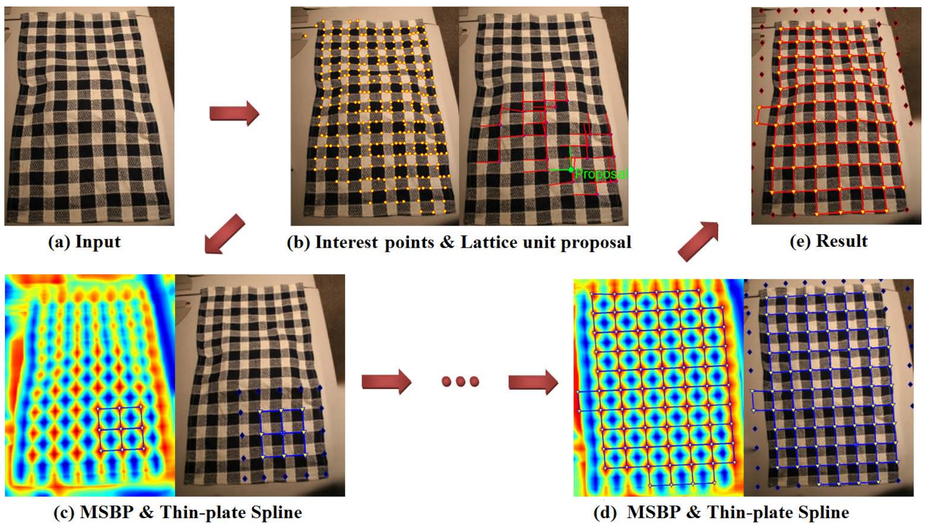
(a) Input
(b) Interest points & Lattice unit proposal
Proposal
(e) Result
(c) MSBP & Thin-plate Spline
(d) MSBP & Thin-plate Spline

Fig. 2. Our proposed algorithm proceeds from (a) to (e): (a) is input, (b) is the output of phase I, (c) and (d) are results from phase II and III, and (e) is the final result (refer also to Figure 1).

features [16]. Since KLT generates a sorted list based on corner strength, the top N features are passed to the next phase as candidate interest points. We set $N = 300$ for all images in our experiments.

Lattice-unit proposal. Our lattice-unit proposal step differs significantly from [12] in two ways. First, each lattice-unit found is composed of the current point under consideration and its *two* nearest matched neighbors, forming an L-shaped (t_1, t_2)-vector pair (Figure 3), as opposed to the previous practice of considering t_1 and t_2 sequentially [12]. Second, the final lattice-unit proposal is generated by a consensus vote of all potential t_1, t_2-vector pairs. This is equivalent to bringing higher-order constraints into the proposal stage so that unnecessary computation of infeasible lattice points is avoided early on, rather than waiting to prune hopeless lattice points at a later stage [12].

A "candidate neighbor" list L_c^i is kept for each interest point p_i, denoting good potential matches within its neighborhood. The goodness is measured by normalized cross correlation (NCC) between the candidate match and the reference intensity template centered at interest point p_i. Let the list of candidate neighbors of point p_i be $L_c^i = \{p_k | k = 1, \cdots, n, k \neq i\}$ and suppose points p_a and p_b are chosen from this list. Then we have $t_1 = p_a - p_i$ and $t_2 = p_b - p_i$ as a potential (t_1, t_2) candidate pair for the i^{th} point p_i. We collect all the possible (t_1, t_2)-vector pairs from the lists, $L_c^i, i = 1, \cdots, N$ to form L_{tv}, a list of all candidate (t_1, t_2)-vector pairs. If there is regularity in the image, a set of spatially and appearance-wise consistent (t_1, t_2)-vector pairs should exist.

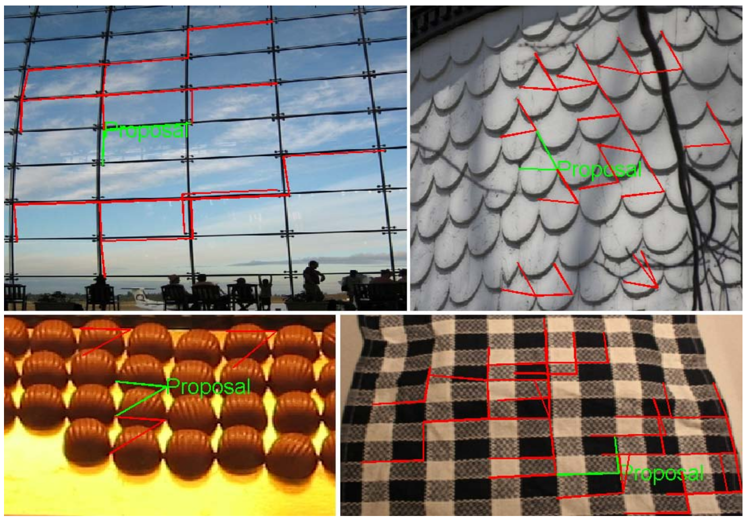

Fig. 3. The results of t_1, t_2-vector proposal : The green L-shape is the t_1, t_2-vector proposal, and the red ones are its supporting members (inlier votes). The images are cropped to emphasize the area of interest.

We measure the spatial consistency of two (t_1, t_2)-vector pairs, (t_1^i, t_2^i) and (t_1^j, t_2^j), using the normalized error term defined below:

$$E(t_1^i, t_2^i, t_1^j, t_2^j) = max(\left\| t_1^i - t_1^j \right\|_2 / \left\| t_1^i \right\|_2, \left\| t_2^i - t_2^j \right\|_2 / \left\| t_2^i \right\|_2) \quad (2)$$

where $\|\|_2$ is L_2 vector norm.

We regard a (t_1, t_2)-vector pair as an inlier if E is smaller than ε. This ε is a tunable knob that can be used to gauge the algorithm's tolerance to geometric irregularity in the found lattice (larger values allow more irregularity). We set $\varepsilon = 0.2$ throughout our experiments. The candidate (t_1, t_2)-vector pair with the largest number of inliers is selected as the best t_1, t_2 proposal.

3.2 Phase II: Lattice Expansion - Spatial Tracking with MSBP

We treat discovery of the underlying lattice of unknown texture elements in an image as a multitarget tracking problem. Without knowing the target to start with, Phase I of our algorithm proposes a lattice unit as a potential target. Now we can state our multi-target tracking problem as: given a single lattice unit **u** and a proposed estimate of the (t_1, t_2)-vectors, a lattice of elements can be extracted by translating **u** spatially via $t1, t2$ in multiple directions to predict its potential locations and then "tracking" those new locations by combining the predictions with image data to produce refined location estimates. Since the lattice may be geometrically distorted in the image, it is dangerous to try to predict the whole lattice of elements at once. Instead, an initial seed lattice is predicted and refined, and then gradually grown outwards into a larger and larger lattice, while the image is progressively unwarped to “straighten out” geometric deformations discovered in the gradually growing lattice structure.

Given its efficiency, we have used the MSBP algorithm [15] as our inference engine for refining predicted texture element locations (Section 2). An initial lattice of size 3 by 3 is built from the (t_1, t_2) proposal generated in Phase I, and an initial texture element template T_0 centered at the origin of the proposed (t_1, t_2)-vectors with size $min(\|t_1\|_2, \|t_2\|_2)$ by $min(\|t_1\|_2, \|t_2\|_2)$ is used to generate an image likelihood map via NCC. This image likelihood map is then taken as a prior density function on the image location of texture elements, and the joint compatibility function (observation model) in the lattice MRF is given by

$$\phi(\mathbf{x}_{[i,j]}, z_{[i,j]}) = \exp(-\alpha(1 - z_{[i,j]})), z_{[i,j]} = NCC(T_0, I(\mathbf{x}_{[i,j]})) \tag{3}$$

where $\mathbf{x}_{[i,j]}$ is the 2D location of node $[i, j]$ at the i^{th} row and j^{th} column in the lattice, $I(\mathbf{x}_{[i,j]})$ is an image patch centered at the location of node $[i, j]$ and T_0 is the initial texture element's appearance template. Equation (3) is of a form typical for data compatibility functions that measure likelihood by appearance similarity. Parameter α is a fixed constant that can be set empirically.

The second kind of function for the MRF is the pairwise compatibility function that specifies the spatial constraints between neighboring pairs of texture elements. In the context of lattice tracking, the pairwise compatibility function governs the geometric characteristics of (t_1, t_2)-vector pairs in the lattice. We define our pairwise compatibility function as

$$\begin{aligned}&\psi(\mathbf{x}_{[i,j]}, \mathbf{x}_{[i,j\pm1]})\\ &= \exp(-\beta \times E(\overrightarrow{\mathbf{x}^{(it)}_{[i,j]}\mathbf{x}^{(it)}_{[i,j\pm1]}}, \overrightarrow{\mathbf{x}^{(it)}_{[i,j]}\mathbf{x}^{(0)}_{[i+1,j]}}, \overrightarrow{\mathbf{x}^{(0)}_{[i,j]}\mathbf{x}^{(0)}_{[i,j\pm1]}}, \overrightarrow{\mathbf{x}^{(0)}_{[i,j]}\mathbf{x}^{(0)}_{[i+1,j]}})^2)\end{aligned} \tag{4}$$

$$\begin{aligned}&\psi(\mathbf{x}_{[i,j]}, \mathbf{x}_{[i\pm1,j]})\\ &= \exp(-\beta \times E(\overrightarrow{\mathbf{x}^{(it)}_{[i,j]}\mathbf{x}^{(it)}_{[i\pm1,j]}}, \overrightarrow{\mathbf{x}^{(it)}_{[i,j]}\mathbf{x}^{(0)}_{[i,j+1]}}, \overrightarrow{\mathbf{x}^{(0)}_{[i,j]}\mathbf{x}^{(0)}_{[i\pm1,j]}}, \overrightarrow{\mathbf{x}^{(0)}_{[i,j]}\mathbf{x}^{(0)}_{[i,j+1]}})^2)\end{aligned} \tag{5}$$

where E is given by equation (2) and measures the consistency of a hypothetical pair of lattice element vectors $(t_1^{(it)}, t_2^{(it)})$ at iteration it with the original proposed vectors $(t_1^{(0)}, t_2^{(0)})$. Equation (4) with subscripts $[-]$ and $[+]$ is used for left-right and right-left message passing respectively. Equation (5) with subscripts $[-]$ and $[+]$ is used for up-down and down-up message passing respectively. The β parameter is a fixed parameter that can be set empirically. Because the error term is normalized, a fixed β parameter can be used for all images regardless of spatial scale of the lattice elements. We used $\alpha = \beta = 5$ in all our experiments. Using the compatibility equations defined above, MSBP [15] is performed. The use of MSBP is critical for speeding up the inference process in real applications, as otherwise the inference process is very slow.

Once the optimization via MSBP converges for this intermediate stage of lattice growth, verification of the converged texture element positions is performed.

This is necessary because propagation of incorrect information to other nodes in the graph may corrupt the optimization process. Verification gives us a safety measure, telling us if we can trust the correspondences between the deformed image lattice and a hypothetical regular lattice, which will be used in Phase III. The general idea behind verification is that we would like to require a converged location to correspond to a significant local maximum or "peak" in the likelihood image. Rather than a hard-coded threshold, we use the region of dominance idea introduced by Liu. et al [10] to determine if an estimated texture element position can be trusted. If the current estimated location is a dominant peak within its neighboring region (that is, if it is a local maximum with a significantly high likelihood score, but is not located close to another local maximum with an even higher score) we select it as a peak location.

Possible mis-alignment at a certain iteration of our algorithm is acceptable because it can be fixed as the iterative fitting procedure repeats. After one iteration of Phase II and Phase III, the lattice structure is expanded from 3 by 3 to 5 by 5, and the growing process continues until each side of the lattice hits a boundary of the image or no more textons are found. A flow diagram of the overall algorithm and the results of subsequent iterations are shown in Figures 1 and 2 respectively.

3.3 Phase III: Regularized Thin-Plate Spline Warping

Once a partial lattice is found in an image, it is natural and useful to relate this possibly deformed lattice to its regular origin: the wallpaper structures. It is natural since the detected degree-4 lattice has the same topological structure as the regular wallpaper patterns [3]. It is useful since straightening out (rectifying) the deformed lattice and its neighborhood helps the iterative algorithm to expand its search for larger and larger lattice structures in the image reliably (Figure 2). We achieve this un-warping step using regularized thin-plate spline (TPS) warping, similar to what is done in both [12,17]. The practical benefits of this phase include: 1) it allows us to deal with deformation discontinuity in the scene; 2) the initial texton proposal can be re-used throughout the procedure, hence preventing the template drift problem; and 3) it enables us to keep the same initial regularized lattice model at all times.

Warping of the found lattice is performed sequentially with template matching on the rectified image, and the results fed back into Phase II for spatial tracking. This process repeats until the growing lattice reaches the edge of the image or there is no more lattice to track (Fig. 2). Although good results can be achieved without warping on patterns where the local distortion is changing gradually, the unwarping process produces improved results when there is an abrupt discontinuity in the distorted pattern, for example, at a fold or crease in the surface.

The main advantage comes from the coupling of BP with MRF and regularized TPS warping. As BP converges, the inference engine provides the deformation correspondences explicitly to the regularized TPS procedure. As the regularized TPS warping rectifies the image, it provides better observation and compatibility constraints in the MRF, resulting in enhanced correspondences on deformed patterns.

4 Experimental Results

We have tested our proposed algorithm on 32 real world images (Figures 4 and 5). Our approach is successful in finding lattice structures in real images, even when

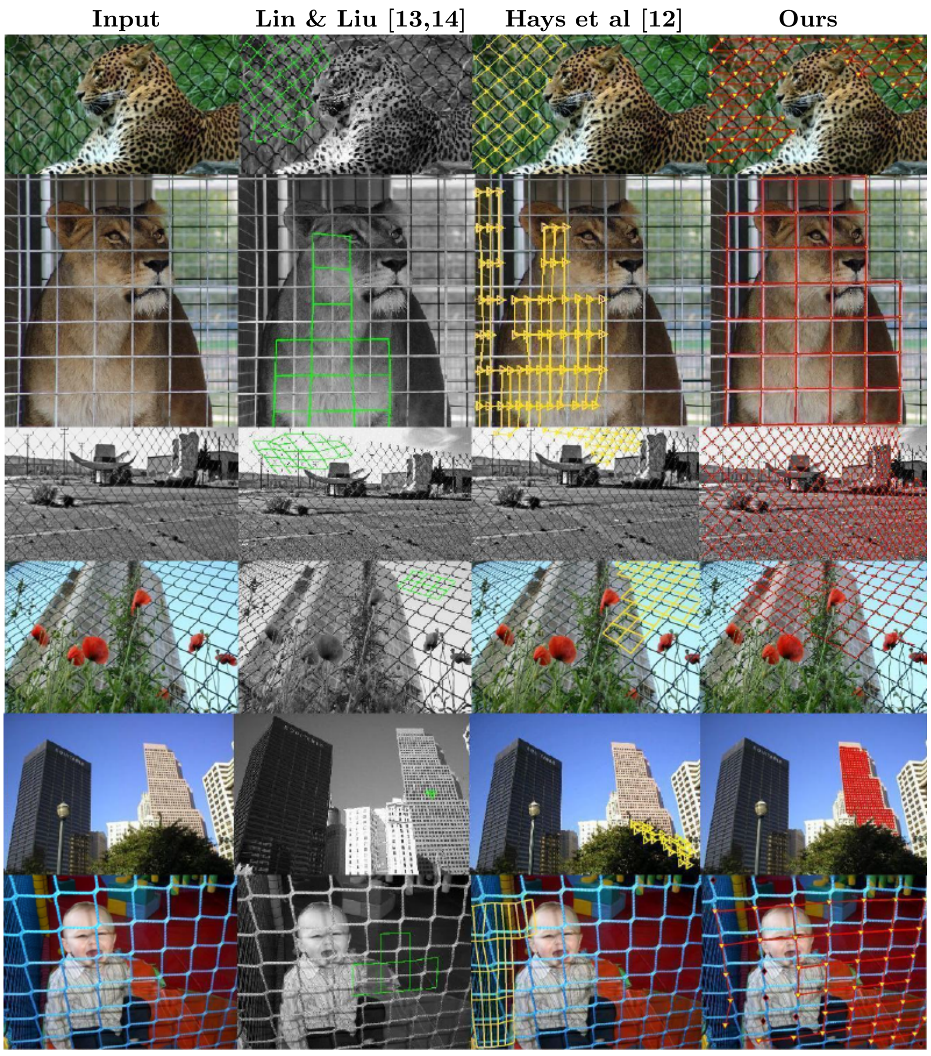

Fig. 4. Sample lattice detection Results: The input images (leftmost), the results of [13,14] (second left), the results of [12] (second right), and our results (rightmost). Note that the comparison with [13,14] is not a fair comparison since they are designed to start interactively. For a complete set of images tested, see our website `http://vision.cse.psu.edu/MSBPLattice.htm`

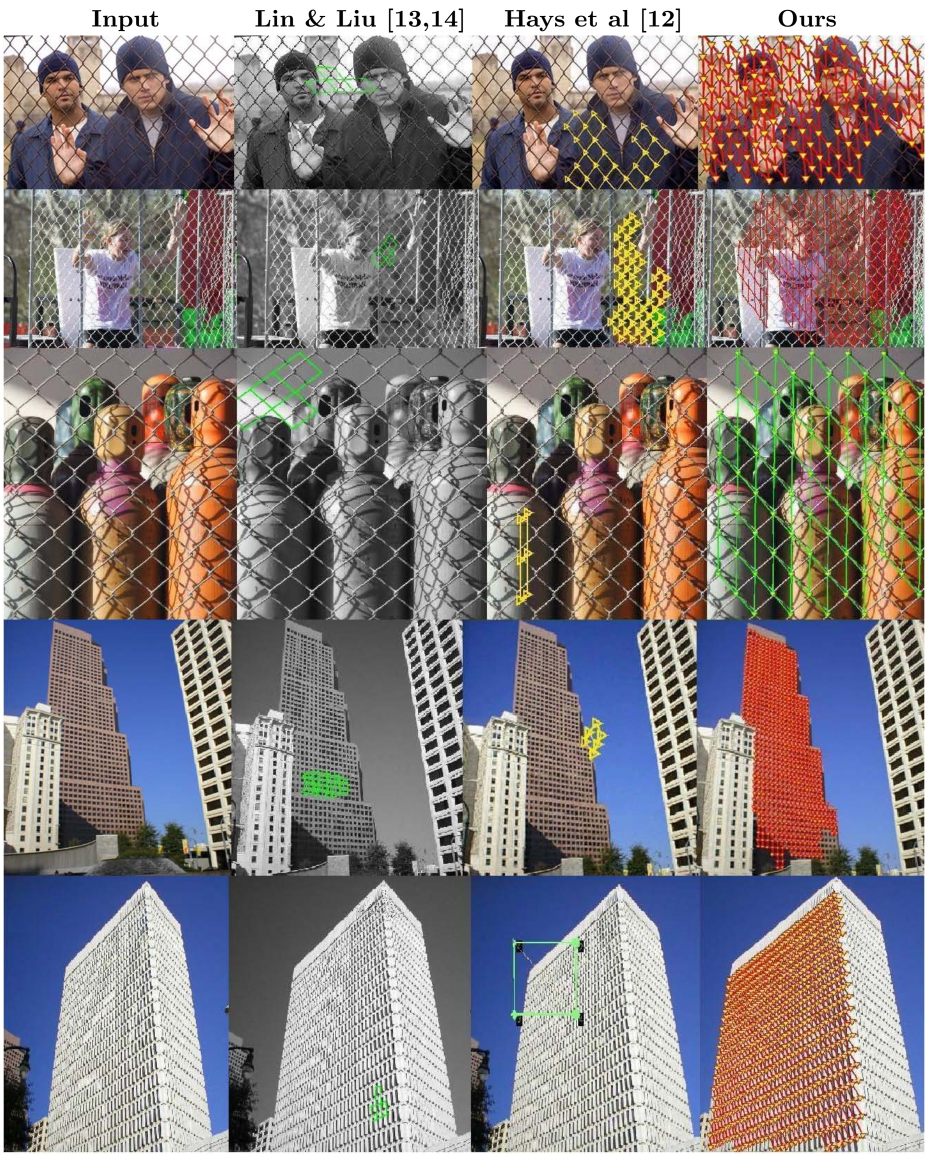

Fig. 5. Sample lattice detection Results: The input images (leftmost), the results of [13,14] (second left), the results of [12] (second right), and our results (rightmost). Note that the comparison with [13,14] is not a fair comparison since they are designed to start interactively. For a complete set of images tested, see our website `http://vision.cse.psu.edu/MSBPLattice.htm`

the scene contains textons with irregular appearances, such as chain link fences where each texture element is dominated by the scattered background. Quantitative evaluations of lattice detection rate and running time on this dataset are as follows: detection rate of Lin and Liu [13,14] is $20 \pm 21\%$, of Hays et al[1] is $47 \pm 38\%$, and of our algorithm is $81 \pm 19\%$. The detection rate is computed by the ratio of the number of detected textons over the number of ground truth textons. The ground truth is manually obtained by two human coders. The average ratio of run times of the Hays' algorithm versus ours is 10.66 ± 9.6. Since [13,14] is a semi-automatic detector, we report the average running time ratio of [12] and ours only. In addition, [12] failed earlier on 5 images which are excluded when computing average time ratio. Running time ratio is defined by the ratio of the time used by [12] over the time used by our algorithm to detect the lattice on each of the 32 test images.

5 Conclusions

We formulate the detection of an underlying deformed lattice from real images as a spatial, multi-target tracking problem using a recently published, efficient MSBP method [15]. The main contributions of our work are: 1) incorporating higher order constraints early-on to propose highly plausible potential interest points as lattice vertices; 2) coupling an MRF optimization procedure with regularized thin-plate spline warping, where MRF with MSBP and regularized TPS warping correct and support each other to yield reliable results in challenging situations; and 3) providing a new lattice detection algorithm with a deterministic time complexity linear in the number of textons in the scene. The quantitative results show that our algorithm is more accurate and significantly faster than other state-of-the-art lattice detection algorithms on diverse real world images.

Acknowledgments

This work was partially funded under NSF grant IIS-0535324 and NSF grant IIS-0729363. We thank J. Hays and W.C. Lin for sharing their code, and K. Brocklehurst for helping with the ground truth labeling.

References

1. Gibson, J.J.: The Perception of the Visual World. Houghton Mifflin, Boston (1950)
2. Julesz, B.: Visual pattern discrimination. IRE Transactions on Information Theory 8(2), 84–92 (1962)
3. Coxeter, H.: Introduction to Geometry, 2nd edn. Wiley, New York (1980)
4. Weyl, H.: Symmetry. Princeton University Press, Princeton (1952)

[1] The algorithm of [12] includes an element of randomness; it runs for several iterations, and takes the best result according to a modified A-score. A-score, originally introduced in [7], is the average per-pixel standard deviation among the final, aligned texels. The modification is the inclusion of $\sqrt{n}$ in the divisor in order to bias the A-score toward more complete lattices [12].

5. Malik, J., Belongie, S., Shi, J., Leung, T.: Textons, contours and regions: cue integration in image segmentation. In: The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 2, pp. 918–925 (1999)
6. Forsyth, D.: Shape from texture without boundaries. In: 7th European Conference on Computer Vision, pp. 43–66 (2002)
7. Liu, Y., Lin, W.C., Hays, J.: Near-regular texture analysis and manipulation. ACM Transactions on Graphics 23(3), 368–376 (2004)
8. Ghanem, B., Ahuja, N.: Phase based modelling of dynamic textures. In: IEEE 11th International Conference on Computer Vision, pp. 1–8 (2007)
9. Schattschneider, D.: The plane symmetry groups: their recognition and notation. American Mathematical Monthly 85, 439–450 (1978)
10. Liu, Y., Collins, R.T., Tsin, Y.: A computational model for periodic pattern perception based on frieze and wallpaper groups. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(3), 354–371 (2004)
11. Liu, Y., Tsin, Y., Lin, W.C.: The promise and perils of near-regular texture. International Journal of Computer Vision 62(1-2), 145–159 (2005)
12. Hays, J., Leordeanu, M., Efros, A., Liu, Y.: Discovering texture regularity as a higher-order correspondence problem. In: 9th European Conference on Computer Vision (2006)
13. Lin, W.C., Liu, Y.: Tracking Dynamic Near-regular Textures under Occlusions and Rapid Movements. In: 9th European Conference on Computer Vision (2006)
14. Lin, W.-C., Liu, Y.: A lattice-based MRF model for dynamic near-regular texture tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(5), 777–792 (2007)
15. Park, M., Liu, Y., Collins, R.T.: Efficient Mean Shift Belief Propagation for Vision Tracking. In: Computer Vision and Pattern Recognition, Anchorage, Alaska (2008)
16. Shi, J., Tomasi, C.: Good features to track. In: Computer Vision and Pattern Recognition, 1994, pp. 593–600 (1994)
17. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(4), 509–522 (2002)