Tracking Dynamic Near-Regular Texture Under Occlusion and Rapid Movements

Wen-Chieh Lin[1] and Yanxi Liu[2]

[1] College of Computer Science, National Chiao-Tung University, Taiwan
wclin@cs.nctu.edu.tw
[2] School of Computer Science, Carnegie Mellon University, USA
yanxi@cs.cmu.edu

Abstract. We present a dynamic near-regular texture (NRT) tracking algorithm nested in a lattice-based Markov-Random-Field (MRF) model of a 3D spatiotemporal space. One basic observation used in our work is that the lattice structure of a dynamic NRT remains invariant despite its drastic geometry or appearance variations. On the other hand, dynamic NRT imposes special computational challenges to the state of the art tracking algorithms: including highly ambiguous correspondences, occlusions, and drastic illumination and appearance variations. Our tracking algorithm takes advantage of the topological invariant property of the dynamic NRT by combining a global lattice structure that characterizes the topological constraint among multiple textons and an image observation model that handles local geometry and appearance variations. Without any assumptions on the types of motion, camera model or lighting conditions, our tracking algorithm can effectively capture the varying underlying lattice structure of a dynamic NRT in different real world examples, including moving cloth, underwater patterns and marching crowd.

1 Introduction

Real-world examples of near-regular texture (NRT) are numerous, especially in man-made environment, such as fabric patterns, decorated surface of architectures, floor patterns, and wallpapers. Effective computational algorithms to handle NRTs, however, are scarce. Although texture analysis and synthesis have been studied in computer vision and computer graphics for years, the geometric and photometric regularity of NRT have not been fully exploited in existing algorithms. Liu et al.[1, 2] first utilized the idea of departures from regularity to analyze and manipulate a static NRT.

An NRT $\mathcal{P}$ is defined as a geometric and photometric deformation of a regular texture $\mathcal{P} = d(\mathcal{P}_r)$, where $\mathcal{P}_r$ is a congruent wallpaper pattern formed by 2D translations of a single tile and d is the deformation mapping [1]. Dynamic NRTs are NRTs under motion. Correspondingly, we define the basic unit of a dynamic NRT *texton*, as a geometrically and photometrically varying tile, moving through a 3D spatiotemporal space. Topologically, the structure of an NRT can be modeled as a network of statistically varied springs. Photometrically, the appearance of different textons are similar but not exactly identical. The

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part II, LNCS 3952, pp. 44–55, 2006.

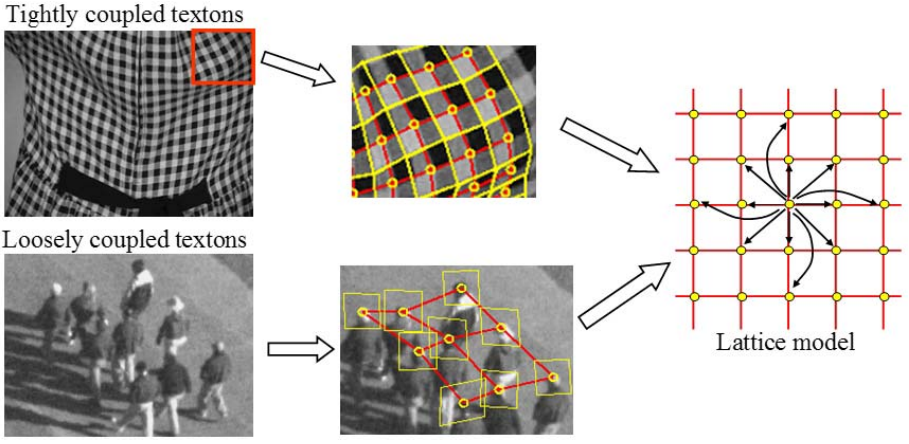


Fig. 1. Lattices (red lines) and textons (yellow quadrilaterals) of different types of NRTs. We model the lattice of an NRT as an MRF where each node represents a texton. The state of a node statistically depends on its twelve neighbors (pointed by arrows).

tracking of dynamic NRT is to treat the deformation field d as a function of time $d(t)$ while maintaining its topological relations. Figure 1 shows the lattice and textons of different types of dynamic NRTs.

In this paper, we further exploit texture regularity to track a dynamic NRT under rapid motion and self-occlusions. One fundamental observation in our work is that the topological structure of a dynamic NRT remains invariant while the NRT going through geometric and photometric variations. Tracking a dynamic NRT, however, is challenging computationally. Because textons of an NRT have similar appearance, a tracking algorithm can easily mistake one texton for another. Furthermore, the tracking problem becomes very difficult when textons move rapidly or occlude each other. Due to these difficulties, tracking textons of a dynamic NRT remains an unsolved problem.

Seeking for an effective computational tool and an in-depth understanding of dynamic NRTs, we propose a lattice-based tracking algorithm based on a spatiotemporal inference formulation. We treat textons of an NRT as multiple targets with a topological constraint while allowing individual textons to vary flexibly in geometry and appearance. Inspired by the physics-based cloth simulation[3], we model the lattice topology as a network of springs and implement it as a Markov Random Field (MRF). We use a Lucas-Kanade-registration-based observation model to handle the appearance and geometry variations of individual textons in the tracking process. Under such a computational modeling of the topology and appearance of a dynamic NRT, we solve the spatiotemporal inference problem using the belief propagation and the particle filtering algorithms. The main contribution of this paper is a framework to track the global structure as well as individual textons of a dynamic NRT, which may undergo rapid motion, occlusion, large geometric and photometric deformations.

2 Related Work

Our work is related to three types of tracking problems: deformable object tracking, cloth motion capture, and multi-target tracking. Image alignment is adopted in many deformable object tracking algorithms where different models are used to confine the deformation space, such as PCA[4, 5], finite element mesh[6], or subdivision surface[7]. These models are not suitable for tracking textons on a folded surface as they assume the surface to be tracked is smooth and non-folded. Recently, Pilet et al. track a non-rigid surface by repeatedly detecting and matching features in an image sequence[8]. They do not handle NRT tracking in which repeated patterns can cause a serious feature correspondence problem.

The goal of cloth motion capture is to capture the 3D motion of cloth. Special calibrated multi-camera systems[9, 10, 11], color-coded patterns[9, 10], or controlled lighting[9] are required to reduce the tracking difficulties due to ambiguous feature correspondences or occlusion problems. Guskov[12] developed an algorithm that can detect and track a black-white chess board pattern on cloth. His algorithm does not work on general NRTs since the black-white chess board pattern is assumed in the detection and image alignment process. Our tracking algorithm can serve as the front end of a cloth motion capture system where no special purpose color-coded pattern or camera calibration is required.

Tracking textons of a dynamic NRT can also be considered as a special case of multi-target tracking. The main difference between NRT tracking and multi-target tracking is that the connection topology among targets does not change in NRT tracking. Modeling the spatial relation among tracked targets using an MRF has been applied to ant tracking[13], sports player tracking[14] and hand tracking[15]. These algorithms may not track dynamic NRTs effectively since topology regularity is not explicitly modeled and utilized.

Existing algorithms for deformable object tracking, cloth motion capture, or multi-target tracking succeed in their respective domains, but none of them deals with general NRT tracking problem under various types of motion and occlusion conditions as treated in this paper. Our approach combines techniques used in multi-target tracking (MRF) and deformable object tracking (image alignment). Therefore, we can track various types of dynamic NRTs under different motion and conditions in a unified framework.

3 Approach

Dynamic NRTs can be categorized into two types based on the spatial connectivity between textons. If the textons of a dynamic NRT are located on a deforming surface where there is no gap between textons, we call this type of texture a *dynamic NRT with tightly coupled textons*. On the other hand, if two neighboring textons are allowed to move with a loosely connected constraint, there might be a gap or overlap between two neighbor textons. We call this type of texture a *dynamic NRT with loosely coupled textons*. Figure 1 illustrates these two types of dynamic NRTs.

Our NRT tracking algorithm consists of four components: 1)**texton detection**, 2)**spatial inference**, 3)**temporal tracking**, and 4)**template update**

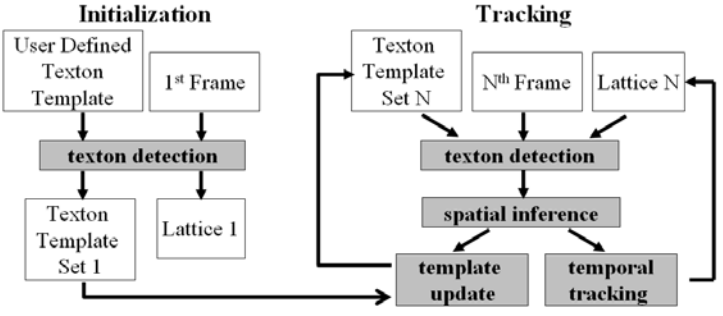


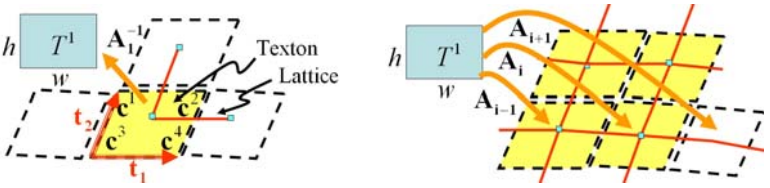
Fig. 2. Tracking approach overview

(Figure 2). In the **initialization stage**, the texton detection algorithm finds all textons in the first frame based on a given template for textons. All detected textons are then geometrically aligned. We call these aligned textons from the first frame *texton templates*. A quadrilateral lattice is constructed by connecting the centers of detected textons. In the **tracking stage**, texton detection is performed at each frame to include any additional texton entering the scene. We handle the texton tracking problem through a statistical inference process consisting of spatial inference and temporal tracking, where the states of a texton (position, shape, and visibility) are sampled and its distribution is modeled by a particle filter in the tracking process. In each frame, a set of sampled states is drawn and a dynamic model generates the predicted states for the next frame. Belief propagation (BP)[16, 17] is then applied to these predicted states to find the most likely lattice configuration based on an MRF-based lattice model and image data. BP also provides the probability of each texton state, which is used to refine the approximation of the distribution of texton states through particle filtering. The above process iterates until the whole image sequence is tracked. In addition, the texton template set is updated to handle the variation of image intensities of textons in the tracking process.

3.1 Tracking Initialization and Texton Detection

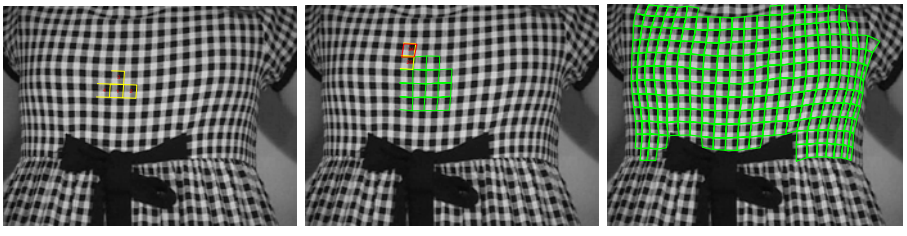
In the initialization stage, the user identifies a texton in the first image frame by specifying two vectors $\mathbf{t_1}$ and $\mathbf{t_2}$ that form a parallelogram (Figure 3(a)). Once the first texton is identified, the second, third, and fourth textons are obtained by translating the first texton by $\mathbf{t_1}$, $-\mathbf{t_1}$ and $\mathbf{t_2}$. A texton template T^1 is constructed by transforming the parallelogram region in the image to a rectangular region $[1, w] \times [1, h]$, where $w = length(\mathbf{t_1})$, $h = length(\mathbf{t_2})$, and the affine transformation matrix $\mathbf{A_1}$ is parameterized by the image coordinates of texton vertices $(c^{1x}, c^{1y}), (c^{2x}, c^{2y}), (c^{3x}, c^{3y}), (c^{4x}, c^{4y})$,

$$\mathbf{A_1} = \begin{bmatrix} c^{1x} & c^{3x} & \frac{c^{2x}+c^{4x}}{2} \\ c^{1y} & c^{3y} & \frac{c^{2y}+c^{4y}}{2} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} w & 1 & w \\ 1 & h & \frac{h}{2} \\ 1 & 1 & 1 \end{bmatrix}^{-1} \tag{1}$$



(a) (b)

Fig. 3. (a)Initial texton (yellow parallelogram formed by $\mathbf{t_1}$ and $\mathbf{t_2}$) and lattice (red lines). The neighboring textons are estimated by translating the first texton by $\mathbf{t_1}$, $-\mathbf{t_1}$ and $\mathbf{t_2}$. (b)Spatial prediction of the position of a new texton.



(a)initial lattice (b)intermediate result (c)final result

Fig. 4. Temporal tracking initialization via spatial tracking

Using the first four textons as the basis for the initial lattice, the lattice grows by repeating the spatial prediction and the validation steps. In the spatial prediction step, the vertices of a texton are estimated from existing textons. Let $\mathbf{A}_i$ be the affine transformation matrix that maps pixels of the texton template T^i to the texton i in the image. Suppose texton $i-1$, i, and $i+1$ are on the same lattice row or column, and $\mathbf{A}_{i-1}$ and $\mathbf{A}_i$ are known, $\mathbf{A}_{i+1}$ is predicted by $\mathbf{A}_{i+1} = \mathbf{A}_i \cdot \mathbf{A}_{i-1}^{-1} \cdot \mathbf{A}_i$[18]. In the validation step, we verify if the texton is valid by checking its associative topology constraints, area and side length difference with the neighboring textons. Additionally, the vertex positions of all valid textons are refined through an image alignment process where a global optimization that involves the whole lattice is performed[18]. The spatial prediction and validation process are repeated until no new texton is detected. A texton template set $\mathcal{T}_1 = \{T_1^i\}_{i=1}^N$ is constructed by collecting all valid texton template T_t^i, where $\mathcal{T}_t$ denotes the template set at frame t. The initial configuration of lattice is obtained by connecting all the centers of textons. Figure 4 shows texton detection results at different stages of tracking initialization.

3.2 Spatial Inference

Let $X_t = (x_t^1, x_t^2, ..., x_t^N)$ be the configuration of the lattice and $Z_t = (z_t^1, z_t^2, ..., z_t^N)$ the image observation at frame t, where N is the number of textons, and x_t^i, z_t^i represent the state and the image intensities of texton i respectively. The spatial inference problem is modeled as a Markov network[17]:

$$p(X_t|Z_t) \propto \prod_{(i,j)\in\mathcal{E}} \varphi(x_t^i, x_t^j) \prod_{i=1}^{N} \phi(x_t^i, z_t^i) \tag{2}$$

where $x_t^i = (c_t^{i1x}, c_t^{i1y}, c_t^{i2x}, c_t^{i2y}, c_t^{i3x}, c_t^{i3y}, c_t^{i4x}, c_t^{i4y}, v_t^i)$ is the state of a texton and $\mathcal{E}$ is the set of all connected edges. The pair (c^{ikx}, c^{iky}) denotes the image coordinates of the kth vertex of the texton $(k = 1, 2, 3, 4)$ and $v_t^i \in (0, 1)$ represents visibility of texton i at frame t. The first product term $\prod \varphi(x_t^i, x_t^j)$ in Equation (2) can be considered as a **lattice model** that models the probabilistic relation among textons, and the second product term $\prod \phi(x_t^i, z_t^i)$ is an **observation model** that evaluates the likelihood of texton states based on image data.

Lattice Model. We model the lattice structure as a pairwise MRF. An MRF is an undirected graph $(\mathcal{V}, \mathcal{E})$ where the joint probability is factored as a product of local potential functions at each node (each node corresponds to a texton), and the interactions are defined on neighborhood cliques. The most common form of MRF is a pairwise MRF in which the cliques are pair of connected nodes in the undirected graph. The potential function in our MRF is defined as follows:

$$\varphi(x_t^i, x_t^j) = e^{-\beta \cdot d_g(x_t^i, x_t^j)} \tag{3}$$

$$d_g(x_t^i, x_t^j) = (\|\mathbf{c}_m^i - \mathbf{c}_m^j\| - l_t^{ij})^2 \cdot v_t^i v_t^j \tag{4}$$

where β is a global weighting scalar that is applied to all springs. β weights the influence of the lattice model versus the observation model in the Markov network. d_g is a function that measures the geometric deformation (spring energy function). $\mathbf{c}_m^i \in \mathbb{R}^{2\times 1}$ is the mean position of four vertices of the texton i. This potential function acts like a spring that adjusts the position of textons based on their mutual distance. The rest length l_t^{ij} of the spring is spatially dependent. To handle occlusion, v_t^i and v_t^j in Equation (4) are used to weigh the influence of a node by their visibility status.

The neighborhood configuration of the MRF in our lattice model is similar to the spring connection used in cloth motion simulation[3]. Figure 1 shows the connection of a node where the state of a node depends on the states of its twelve neighbors. It has been shown in cloth simulation that this kind of configuration provides a good balance between structural constraint and local deformations.

Observation Model. We define the image likelihood as follows:

$$\phi(x_t^i, z_t^i) \propto e^{-\frac{1}{v_t^i} d_a(x_t^i, z_t^i, T_t^i)} \tag{5}$$

where the appearance difference function d_a is weighted by the visibility score v^i of a texton so that visible textons contribute more in the likelihood function. $d_a = \sum_{r=1}^{2} \sum_{\mathbf{p}} \|z_t^{ir}(\mathbf{p}) - T_t^i(\mathbf{p})\|^2$ is the sum of squared differences (SSD) between a texton template T_t^i and the observed texton at frame t. $\mathbf{p}$ denotes a pixel location in the coordinate frame of the template. $z_t^{ir} = I_t(\mathbf{W}(\mathbf{p}; \tilde{\mathbf{a}}_t^{ir}))$ is an aligned texton obtained from the affine warp $\mathbf{W}$ whose parameters $\tilde{\mathbf{a}}_t^{ir}$ are computed by the Lucas-Kanade algorithm using the texton vertex coordinates

(c_t^{ikx}, c_t^{iky}) as the initial values (see appendix in [18]). Note that a quadrilateral texton is divided into two triangles and the vertex coordinates of each triangle are used to parameterize $\mathbf{a}_t^{i1}$ and $\mathbf{a}_t^{i2}$ respectively. If textons are tightly coupled, the textured region is modeled as a piecewise affine warp and the position of each texton vertex is affected by four neighboring textons. This enforces hard connected constraints among textons when computing $\tilde{\mathbf{a}}_t^{ir}$. If the textons are loosely coupled, $\tilde{\mathbf{a}}_t^{ir}$ of each texton is computed independently. This allows the observation model to handle more flexible motion, such as underwater texture, or people in a crowd.

Visibility Computation. The visibility of a texton is determined by constraints and measurements related to geometry and appearance of a texton. The constraints, which include topology, side length and area difference with neighboring textons, are used to decide if a texton is valid and can be included in the tracking process. The measurements define the visibility score v_t^i of a valid texton i at frame t:

$$v_t^i = \frac{1}{1+\rho}\left(\frac{s^i}{s^*} + \frac{\rho}{4}\sum_{k=1}^{4}\frac{|b_k^i - b_k^*|}{b_k^*}\right) \tag{6}$$

where ρ is a constant to weight the influence of area and side length variations in the visibility measurement. s^i and s^* are the area of texton i and the seed texton. b_k^i and b_k^* are the kth side length of texton i and the seed texton. A visibility map V is constructed based on the visibility scores of all textons:

$$V = \{M^i | M^i \in (0,1), i = 1 \ldots N\} \tag{7}$$

where $M^i = 1$ if $v_t^i \geq 0.5$; $M^i = 0$ if $v_t^i < 0.5$.

Belief Propagation. The spatial inference is solved by the belief propagation (BP) algorithm[16, 17]. Since the conventional BP algorithm works on discrete variables while the configuration of a lattice is described by continuous variables, we need to either discretize the state variables or apply continuous BP algorithms[19, 20]. For computational efficiency, we choose to use the discrete BP and adopt the sample-based statistics to represent the continuous state variables for each texton. Particle filtering[21, 22] is applied to update the particle set for each texton in the temporal tracking process.

3.3 Temporal Tracking

We adopt particle filtering to represent and maintain the distribution of the lattice configurations in temporal tracking. The belief distribution computed by the BP is used in importance sampling to draw new samples. The dynamic model is then applied to predict a set of states for each texton, and the discrete BP is applied to infer the most likely configuration based on these predicted states.

We use a second-order dynamic model, i.e., the state of the lattice at current frame depends on the states at previous two frames:

$$p(X_t | X_{t-1}, X_{t-2}) \propto \prod p(x_t^i | x_{t-1}^i, x_{t-2}^i) \tag{8}$$

where a constant velocity model with Gaussian noise is used for each texton:

$$p(x_t^i|x_{t-1}^i, x_{t-2}^i) = \mathcal{N}(x_t^i - 2x_{t-1}^i + x_{t-2}^i; 0, \Lambda_i) \quad (9)$$

Λ_i is a diagonal matrix whose diagonal terms correspond to the variance of the state at different dimensions.

Our approach of combining BP and particle filter is similar to PAMPAS[20] in spirit, however, PAMPAS incorporates particle filter in the message propagation process within BP while we only use particle filter to carry the texton states between image frames. Note that Guskov et al.[10] also used the Markov network to associate color-coded quadrilaterals in an image with the quadrilaterals of the surface model. They did not use the Markov network to infer the position and the shape of the textons.

3.4 Template Update

As the appearance of textons vary during tracking process, it is necessary to update the texton template set. We use the template updating algorithm in [23] where the basic idea is to correct the drift in each frame by additionally aligning the current image with the template at the first frame. After aligning the current image with the previous frame, the computed warping parameters are used as the initial values in the additional alignment process. If the warping parameters obtained from the second image alignment process is close to the first one, the template is updated; otherwise, the template remains unchanged.

4 Results

4.1 Tracking Dynamic NRTs Without Occlusion

We tested our tracking algorithm on several dynamic NRTs under different types of motions[1] (Figure 5). We also compared our results against the robust optical flow algorithm as it is a general purpose tracking algorithm[24]. Although there are more sophisticated tracking algorithms, they are usually designed for certain types of problems (section 2). Due to similar appearance of NRT textons, the optical flow tracking algorithm was distracted by neighboring textons. We also tested our algorithm on a pattern viewed through disturbed water (Figure 5(b)(e)). The appearance of textons vary rapidly in the video because of surface refraction and motion blur. Despite these difficulties, our algorithm is able to track these highly dynamic and varied textons successfully. These two experiments demonstrate that, even without occlusion, general tracking algorithms like optical flow is not suitable to track a dynamic NRT in a video effectively.

The textons of the underwater texture are modeled as a loosely coupled MRF allowing flexible motion of textons. Figure 5(c)(f) show another example of tracking loosely coupled textons. In this example, a texton is defined as a local patch around the head region of a person. The marching motion presents a relatively large global motion and small local deformation of individual textons compared to the motion of tightly coupled textons. Also, the appearance of textons varies

[1] All video results can be seen in *http://www.cs.cmu.edu/∼wclin/dnrtPAMI/dnrt.html.*

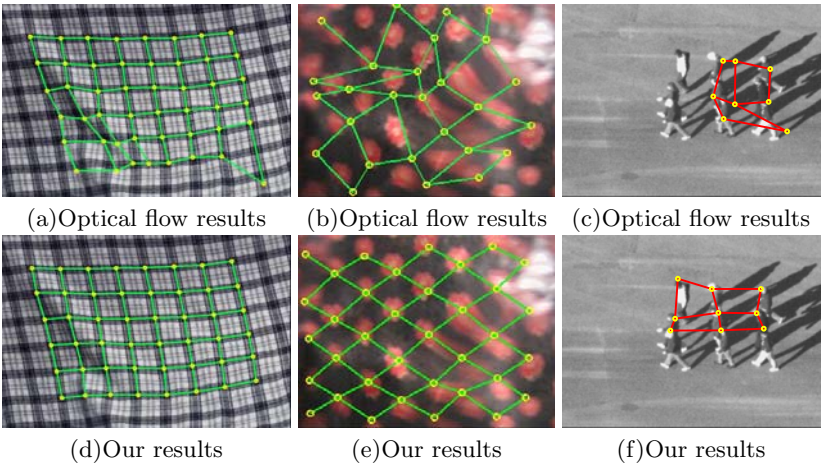


Fig. 5. Comparison of dynamic NRT tracking results without occlusion (robust optical flow[24] vs. ours). (a)(d) is a dynamic NRT on slowly varying cloth. (b)(e) is a pattern seen through disturbed water; there are serious motion blur and reflection highlights in the video. (c)(f) is a crowd motion exhibiting NRT property.

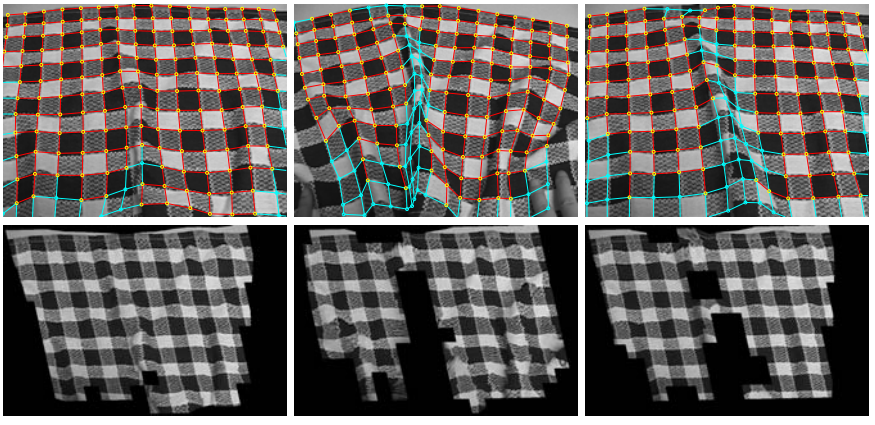
more due to shadows. The underwater texture and crowd marching examples show that our algorithm is able to handle large illumination changes, rapid geometric deformation, and intensity variations in the tracking process.

4.2 Tracking Dynamic NRTs with Occlusion

Occlusion is one of the major challenges in dynamic texture tracking. Textons may leave/enter the scene, or be occluded by other objects or other textons on a folded surface. Figure 6 shows our tracking result of a folding fabric pattern under self and external occlusions. The lattice, visible textons, and occluded textons are shown in red, yellow, and cyan colors respectively. The bottom row in Figure 6 shows visibility maps of textons where black regions correspond to occluded textons and visible textons are geometrically aligned. One can observe that a few textons are occluded in the middle and two occluded by a finger in the bottom-right region in Figure 6(b). When the texton is at the boundary of a lattice, the BP inference result for the texton is less reliable since it receives messages from fewer neighboring nodes. This is the reason why there are some tracking errors in the cyan lattice at boundary, e.g, top-middle in Figure 6(c).

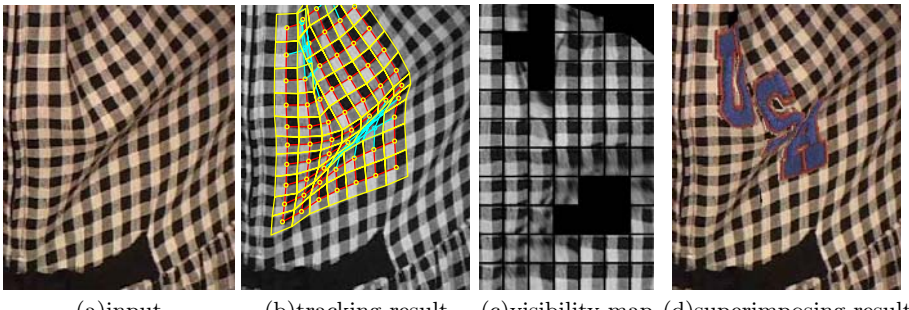
The results of dynamic NRT tracking can be used in many other applications, e.g., video editing, cloth motion capture, and fashion design preview (changing cloth texture). Figure 7 demonstrates a superimposing application as a result of the NRT tracking. For more tracking and video editing results, and comparisons to other tracking algorithms, please see [18].

Although our algorithm can successfully track textons through occlusion, there is another interesting research problem: can we infer the positions of textons when they are occluded? One way to solve this problem is to modify the



(a) frame 25 (b) frame 50 (c) frame 100

Fig. 6. Tracking lattices (top row) and visibility map (bottom row) of a folding fabric pattern. The visible lattice, occluded lattice, visible textons, and occluded textons are shown in red, cyan, yellow, and cyan color. One can observe that there are self-occlusions due to folding in (b)(c) and external occlusion by a finger in (b).



(a)input (b)tracking result (c)visibility map (d)superimposing result

Fig. 7. Tracking and superimposition results of a fabric pattern under occlusion

MRF model such that it can represent a folded topology under occlusion. We would like to explore this problem in the future.

5 Conclusion

We propose a lattice-based dynamic NRT tracking algorithm which combines a lattice structure model to represent the topological constraint of a dynamic NRT and a registration-based image observation model to handle the geometry and appearance variations of individual textons. We demonstrate the effectiveness of our algorithm on tracking dynamic NRTs under rapid movements, folding motion or illumination changes through different mediums. There are several

remaining future research issues. First, given the captured lattice structure of an NRT on a 3D surface, a shape-from-texture algorithm may be applied to obtain the 3D geometry of the textured surface. Results from this will further expand graphics applications of our tracking algorithm. Secondly, we would like to investigate whether it is necessary to use a varying topology formalization for extremely deformed regular textures. For example, the topological structure for a crowd motion may vary drastically. It may or may not be beneficial to allow adaptive topology during the tracking process. Finally, a thorough and more comprehensive evaluation of various tracking algorithms on dynamic NRT tracking is needed to enhance and solidify our understanding of both the state of the art tracking algorithms and the dynamic NRT itself.

Acknowledgement

We thank the reviewers for their constructive comments; Robert T. Collins for providing the crowd motion video and proofreading this paper; Jing Xiao, Jiayong Zhang, Chieh-Chih Wang and Sanjiv Kumar for their suggestions on our tracking algorithm; Jasmine Collins for being the subject in the dress video. This work was supported in part by an NSF grant IIS-0099597.

References

1. Liu, Y., Lin, W.C., Hays, J.: Near-regular texture analysis and manipulation. In: ACM SIGGRAPH. (2004) 368–376
2. Liu, Y., Tsin, Y., Lin, W.C.: The promise and perils of near-regular texture. International Journal of Computer Vision **62**(1-2) (2005) 145–159
3. House, D.H., Breen, D.E., eds.: Cloth Modeling and Animation. A.K. Peters, Ltd., Natick, Massachusetts (2000)
4. Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active appearance models. In: ECCV. (1998) 484–498
5. Matthews, I., Baker, S.: Active appearance models revisited. International Journal of Computer Vision **60**(2) (2004) 135 – 164
6. Sclaroff, S., Isidoro, J.: Active blobs. In: ICCV. (1998) 1146–1153
7. Guskov, I.: Multiscale inverse compositional alignment for subdivision surface maps. In: ECCV (1). (2004) 133–145
8. Pilet, J., Lepetit, V., Fua, P.: Real-time non-rigid surface detection. In: CVPR. (2005) 822–828
9. Scholz, V., Stich, T., Keckeisen, M., Wacker, M., Magnor, M.: Garment motion capture using color-coded patterns. In: Eurographics. (2005) 439–448
10. Guskov, I., Klibanov, S., Bryant, B.: Trackable surfaces. In: ACM Symposium on Computer Animation. (2003) 251–257
11. Pritchard, D., Heidrich, W.: Cloth motion capture. In: Eurographics. (2003)
12. Guskov, I.: Efficient tracking of regular patterns on non-rigid geometry. In: Proceedings of ICPR. (2002)
13. Khan, Z., Balch, T., Dellaert, F.: An mcmc-based particle filter for tracking multiple interacting targets. In: ECCV. (2004) 279–290

14. Yu, T., Wu, Y.: Decentralized multiple target tracking using netted collaborative autonomous trackers. In: CVPR. (2005) 939–946
15. Sudderth, E.B., Mandel, M.I., Freeman, W.T., Willsky, A.S.: Distributed occlusion reasoning for tracking with nonparametric belief propagation. In: Neural Information Processing Systems. (2004) 1369–1376
16. Yedidia, J., Freeman, W., Weiss, Y.: Understanding belief propagation and its generalizations. In: International Joint Conference on Artificial Intelligence. (2001)
17. Freeman, W.T., Pasztor, E.C., Carmichael, O.T.: Learning low-level vision. Int. J. Comput. Vision **40**(1) (2000) 25–47
18. Lin, W.C.: A lattice-based mrf model for dynamic near-regular texture tracking and manipulation. Technical Report CMU-RI-TR-05-58, Ph.D. Thesis, Robotics Institute, Carnegie Mellon University (2005)
19. Sudderth, E., Ihler, A., Freeman, W., Willsky, A.: Nonparametric belief propagation. In: CVPR. (2003) 605–612
20. Isard, M.: Pampas: real-valued graphical models for computer vision. In: CVPR. (2003) 613–620
21. Isard, M., Blake, A.: Condensation – conditional density propagation for visual tracking. Int. J. Comput. Vision **29**(1) (1998) 5–28
22. Doucet, A., Freitas, N.D., Gordon, N.: Sequential Monte Carlo Methods in Practice. Springer-Verlag (2001)
23. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**(6) (2004) 810 – 815
24. Black, M.J., Anandan, P.: The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. Computer Vision and Image Understanding **63**(1) (1996) 75–104