# Dynamic Texture Synthesis in Space with a Spatio-temporal Descriptor

Rocio A. Lizarraga-Morales[1], Yimo Guo[2], Guoying Zhao[2], and Matti Pietikäinen[2]

[1] Universidad de Guanajauto, DICIS. Salamanca, Guanajuato, Mexico
[2] Center for Machine Vision Research, Department of Electrical and Information Engineering, University of Oulu. PO Box 4500, FI-90014, Finland

**Abstract.** Dynamic textures are image sequences recording texture in motion. Given a sample video, the goal of synthesis is to create a new sequence enlarged in spatial and/or temporal domain, which looks perceptually similar to the input. Most synthesis methods are mainly focused on extending sequences only in the temporal domain. In this paper, we propose a dynamic texture synthesis approach for spatial domain, where we aim to enlarge the frame size while preserving the aspect and motion of the original video. For this purpose, we use a patch-based synthesis method based on LBP-TOP features. In our approach, 3D patch regions from the input are selected and copied to an output sequence. Usually, in other patch-based approaches, the selection of the patches is based only in the color, which cannot capture the spatial and temporal information, causing an unnatural look in the output. In contrast, we propose to use the LBP-TOP operator, which implicitly represents information about appearance, dynamics and correlation between frames. The experiments show that the use of the LBP-TOP improves the performance of other methods giving a good description of the structure and motion of dynamic textures without generating visible discontinuities or artifacts.

## 1 Introduction

Texture synthesis (TSyn) has generated considerable research interest in recent years, since it is an essential element in many computer graphics applications. Given a sample texture, the goal is to synthesize a new texture that looks perceptually similar to the input, with an arbitrary size specified by the user. TSyn is a practical alternative way to create textures for a given surface, instead of the more traditional ways like hand drawing or scanning pictures [13]. One primary advantage of TSyn lies on the storage requirements, because it only needs to store a small sample of the texture, regardless of the size of the surface to cover.

Dynamic textures (DTs) are video sequences that are spatially repetitive and temporally stationary [5]. Basically, DTs are textures in motion. Analogously to the definition of TSyn, dynamic texture synthesis (DTSyn) consists in creating an infinite sequence, either in space or time domains, using a video exemplar

J.-I. Park and J. Kim (Eds.): ACCV 2012 Workshops, Part I, LNCS 7728, pp. 38–49, 2013.

as input. The time domain comprises the duration of the video, while the spatial domain consists of enlarging the image size. Both domains must preserve a natural appearance and motion in the outputs.

Different methods for DT synthesis have been proposed. These approaches can be separated into two categories: parametric and non-parametric. Parametric methods are applied to model the behavior of a given phenomenon as a linear dynamic system [3,4,8,15], and typically, they are focused on the two domains at the same time. Even though these methods are able to obtain an output similar to the input, the visual quality is not realistic enough. On the other hand, non-parametric, or exemplar-based, methods are based on taking small parts from an input sample as elements to build the output. Results of non-parametric methods look more natural and realistic than the parametric methods, in view of that these approaches reuse the information of the input.

Non-parametric approaches have been used to synthesize dynamic textures in both time and space domains. Considerable work has been developed for DTSyn along the time domain, for example, in [6,11,7]. The idea behind these techniques for extending the duration of the video, is to find sets of matching frames in the input video and then, jump between these frames during playback. On the other hand, in order to enlarge the frame size, but keeping the duration of the video with non-parametric methods, two main approaches have been followed: pixel-based and patch-based methods. The essential difference between these two methods is in how the information is transferred to the output. As their name says, the pixel-based methods transfer one pixel at a time. The value of each pixel in the output is determined by comparing its spatial neighborhood with all neighborhoods in the input texture. Some pixel based techniques that have been applied for DTSyn are those introduced by Bar-Joseph *et al.* [1] and Wei and Levoy [14]. By contrast, patch-based techniques select and copy whole neighborhoods each time to the output. With these methods, the speed and quality of synthesis can be improved. However, the problem of how to avoid mismatches between adjacent patches arises. The patch is pasted on the output with a portion of overlapped volume with the already synthesized portion. The patch can be just blended, or an optimal cut can be found for seaming the two patches. In these methods, each patch must be carefully selected depending on a given visual feature. Typically, only the color of the pixels is considered. A significant number of patch-based approaches for static texture synthesis has been proposed, while dynamic textures synthesis has not received the same attention. One representative method for DTSyn in space is the proposed by Kwatra *et al.* [7], using dynamic programming to find an optimal path to cut through the overlapped regions considering only the color of pixels. Even though pixel-based and patch-based approaches have obtained good results, the influence of the visual features used for DT description for DTSyn remains unexplored. According to Chetverikov and Péteri [2], fundamental issues regarding the DT description include the combination of appearance with motion features. This issue cannot be achieved by only using the intensity of pixels and must be considered for DTSyn implementations.

In this paper, we propose the use of local binary patterns from three orthogonal planes [16] as a reference feature in a non-parametric patch-based method for DTSyn in space. This operator can capture the structure of local brightness variations in three orthogonal planes, and therefore, describe appearance and motion based on the local spatial and temporal patterns. The use of this operator gives to our approach an advantage in comparison to those based only on the color. In addition, it is not intricate since we do not need an optimization of the boundary zone between adjacent patches. Experiments carried out on different dynamic textures show that the use of LBP-TOP features allows a better description of DT patches and preserves the structure and dynamics without generating visible discontinuities between regions. It is also shown that our method can achieve better or at least similar performance to previously proposed methods.

This paper is organized as follows: in Section 2, the LBP-TOP operator and the synthesis algorithm are presented. Experiments and results are presented in Section 3, and concluding remarks are given in Section 4.

## 2 Dynamic Texture Synthesis in Space Using LBP-TOP Features

The proposed method for DTSyn in spatial domain is carried out by using a spatio-temporal descriptor as visual feature, which allows a better perceptual representation of DT. Details of the implementation are given below.

### 2.1 Spatio-Temporal Descriptor

The local binary pattern from three orthogonal planes (LBP-TOP) [16], is a spatio-temporal descriptor for dynamic textures. The LBP-TOP considers the co-occurrences in three planes XY, XT and YT, capturing information about the space-time transitions, as shown in Fig. 1.

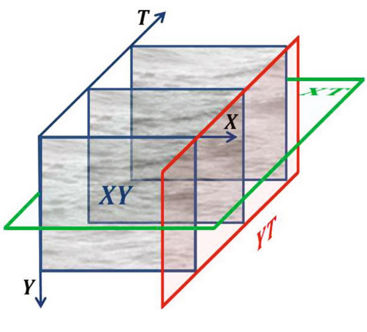



**Fig. 1.** The LBP-TOP feature is obtained by extracting the LBPs from three orthogonal planes

The LBP-TOP is an extension of the Local Binary Patterns (LBP) presented by Ojala *et al.* [9]. As it is known, the LBP is a theoretically simple, yet efficient approach, to characterize the spatial structure of local texture. Basically, the operator labels a given pixel of an image by thresholding its neighbors in function of the pixel intensity and summing the thresholded values weighted by powers of two. According to Ojala, a static texture $T$ in a local neighborhood of a monochrome texture image is defined as the joint distribution of the gray levels of $P(P > 1)$ image pixels $T = t(g_c, g_0, \ldots, g_{P-1})$, where $g_c$ is the gray value of the center pixel and $g_p(p = 0, 1, \ldots, P-1)$ are the gray values of P equally spaced pixels on a circle radius $R(R > 0)$, that form a circularly symmetric neighbor set. If the coordinates of $g_c$ are $(x_c, y_c)$, then the coordinates of $g_p$ are $(x_c - R\sin(2\pi p/P), y_c + R\cos(2\pi p/P))$. The LBP code for the pixel $g_c$ is defined as

$$LBP_{P,R}(g_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{1}$$

where the thresholding function $s(\cdot)$ is defined in equation 2. More details can be further consulted in [9].

$$s(t) = \begin{cases} 1, t \geq 0 \\ 0, \text{otherwise} \end{cases} \tag{2}$$

For the spatio-temporal extension of the LBP, named as LBP-TOP, the local patterns are extracted from the XY, XT and YT planes. Each code is denoted as XY-LBP for the space domain, and XT-LBP and YT-LBP for space-time transitions [16]. In the LBP-TOP approach, the three planes intersect in the center pixel and three different patterns are extracted in function of that central pixel. The local pattern of a pixel from XY plane, contains information about the appearance and, in the local patterns from XT and YT planes, statistics of motion in horizontal and vertical directions are included. In this case, the radii in axes X,Y and T are $R_X$, $R_Y$ and $R_T$ respectively and the number of neighboring points in each plane are defined as $P_{XY}$, $P_{XT}$, $P_{YT}$. Supposing that the coordinates of the center pixel $g_{t_c,c}$ are $(x_c, y_c, t_c)$, the coordinates of the neighbors $g_{XY,p}$ in the plane XY are given by $(x_c - R_X\sin(2\pi p/P_{XY}), y_c + R_Y\cos(2\pi p/P_{XY}, t_c)$. Analogously, the coordinates of $g_{XT,p}$ in the plane XT are $(x_c - R_X\sin(2\pi p/P_{XT}), y_c, t_c - R_T\cos(2\pi p/P_{XT})$, and the coordinates of $g_{YT,p}$ on the plane YT are $(x_c, y_c - R_Y\cos(2\pi p/P_{YT}), t_c - R_T\sin(2\pi p/P_{YT})$. Consequently, every pixel in the input video is represented by 3 codes, one for each orthogonal plane.

For the implementation proposed in this paper, each pixel of the input sequence $V_{in}$ is analyzed with the LBP-TOP operator, in such a way that we obtain an LBP-TOP-coded sequence $V_{LBP-TOP}$. Each pixel in the $V_{LBP-TOP}$ sequence is coded by three values, comprising each of the space-time patterns of the local neighborhood, as can be seen in Fig. 2. As we said before, in patch-based methods each patch must be carefully selected depending on a given visual feature, then, the patch is positioned with some overlapped area with the already synthesized portion. To accomplish this task, we use $V_{LBP-TOP}$ as a temporary sequence for the patch description in the selection process.

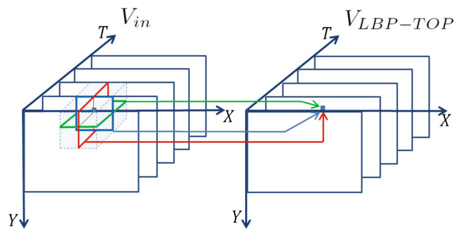


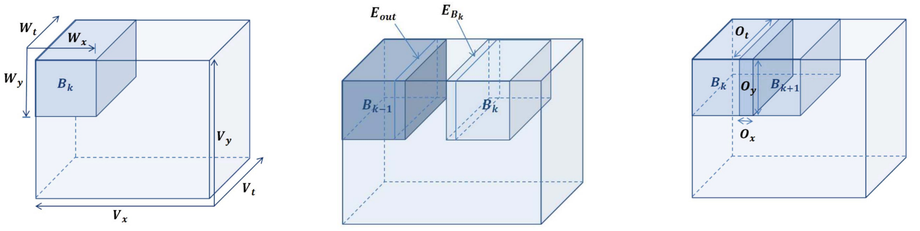

**Fig. 2.** Each pixel in the corresponding LBP-TOP sequence is obtained by extracting the LBPs from the three orthogonal planes in the input sequence

## 2.2 Dynamic Texture Synthesis in Space Domain

In this paper, we propose the use of LBP-TOP features [16] in a non-parametric patch-based method for DTSyn in space. As mentioned, non-parametric algorithms basically select patches, or blocks from the input as elements to build an output. The use of LBP-TOP features, allows us to consider the spatial and temporal relations among pixels and, therefore, obtain more information about a given block and its possible neighbor blocks.

Our method is cyclical, in each step we select a block $B_k$ from the input sample video $V_{in}$ and copy it to the output video $V_{out}$. To avoid discontinuities or artifacts between blocks, we must carefully select $B_k$ based on the blocks already pasted $\{B_0, \ldots, B_{k-1}\}$ in $V_{out}$. At the beginning, a block $B_0$, of $W_x \times W_y \times W_t$ pixels size, is randomly selected from the input $V_{in}$, and copied to the upper left corner of the output $V_{out}$. The following blocks needed to fulfill the output, are positioned in raster scan order in such a way that they are partially overlapped with previously pasted blocks. The overlapped volume between two blocks is of size $O_x \times O_y \times O_t$ pixels. In Fig. 3 an example of a video block, the boundary zone where two blocks should match and an example of the overlapped volume between two blocks are illustrated. In Figure 3(b), the selected block $B_k$ has a boundary zone $E_{B_k}$ and the previously pasted volume in $V_{out}$ has a boundary zone $E_{out}$. According to our method and in order to avoid discontinuities, $E_{B_k}$ and $E_{out}$ should match.

The appropriate description and selection of each block becomes a key issue in our method. In the block selection step, we consider the similarity of the spatio-temporal features on the boundary zones. For this, we first build a set of candidate blocks $A_B$ of $V_{in}$, which are considered to match with the previously pasted volumes in $V_{out}$. Then we select one block randomly from the set. The random selection is performed to keep a good diversity on the blocks selected. Two blocks are considered to match if the distance in the corresponding overlapping volume is lower than a distance tolerance, specified by the user. We




(a) (b) (c)

**Fig. 3.** Examples of (a) a video block, (b) the boundary zone of two different blocks and (c) the overlapped volume between two video blocks. The boundary zones should match.

construct a set of all the potential blocks $B_{(x,y,t)}$ to be considered to match with $E_{out}$. Let $B_{(x,y,t)}$ be the block whose upper left corner is at $(x, y, t)$ in $V_{in}$. We construct

$$A_B = \{B_{(x,y,t)} | d(E_{B(x,y,t)}, E_{out}) < d_{max}, B_{(x,y,t) \in V_{in}}\} \tag{3}$$

where $E_{B_{(x,y,t)}}$ is the boundary zone of $B_{(x,y,t)}$ and $d_{max}$ is the distance tolerance between two boundary zones. Details on how to compute $d(\cdot)$ are given later. When we have determined all the potential blocks, we pick one randomly from $A_B$ to be the $k^{th}$ block $B_k$ to be pasted on $V_{out}$. The size of $A_B$ depends on how many blocks satisfy the similarity constraints. With a high value of $d_{max}$ the output will have a better quality but, few blocks would be considered to be part of $A_B$. By contrast, with a low tolerance a big number of blocks will be part of the set and there will be more options to select, but the quality of the output will be compromised. For a given $d_{max}$, the set $A_B$ could be empty. In such case, we choose $B_k$ to be the block $B_{(x,y,t)}$ in $V_{in}$ with the smallest distance to the boundary zone of the output $E_{out}$. In Fig. 4 the three possible configurations of the overlapping zones between the already pasted zones $E_{out}$ and the new patch $B_k$ are shown. The first possibility, shown in Fig. 4(a), is when $B_k$ is on the first row and goes after $B_0$. The second is when $B_k$ is the first block in the second or subsequent rows (Fig. 4(b)). The third is when $B_k$ is not the first on the second or subsequent rows as shown in Fig. 4(c), here the total distance is the addition of the above and left boundaries distances.

The algorithm to pursue for synthesizing dynamic texture in space can be described as follows:

1. Let $V_{in}$ be an input DT sample of $V_x \times V_y \times V_t$ pixels size. Set the synthesis block size as $W_x \times W_y \times W_t$, and the size of the overlapped volume of two adjacent blocks as $O_x \times O_y \times O_t$. Consider $V_t = W_t = O_t$.
2. Obtain the $V_{LBP-TOP}$ sequence of $V_{in}$.
3. Transfer the first block $B_0$ from $V_{in}$ to the upper left corner of the output $V_{out}$ by random selection. Set $k = 1$.
4. Synthesize next block in raster scan order:

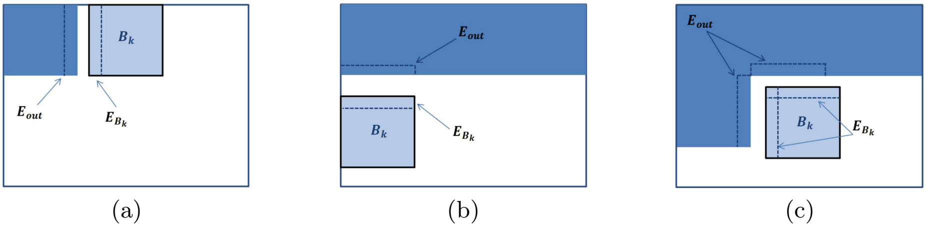



**Fig. 4.** Three possible overlapping zones between the output $E_{out}$ and the new block $E_k$. (a) $B_k$ is in the first row, but after $B_0$. (b) $B_k$ is the first block in the second or subsequent rows and (c) $B_k$ is not the first on the second or subsequent rows.

a) Select a set of candidate blocks $A_B$ from $V_{in}$, such that for each block in $A_B$, the boundary zone satisfies the overlap constraints (above and left) with the previously pasted blocks, with certain tolerance distance between the blocks, computed using the LBP-TOP features.
b) Pick one block randomly from $A_B$ to be $B_k$ and paste it from $V_{in}$ to $V_{out}$. Set $k = k + 1$. Perform blending in the boundary zones.
c) Repeat until $V_{out}$ is completely synthesized.

On the overlapped volume, in order to obtain smooth transitions and minimize artifacts between two adjacent blocks, we blend the volumes using a feathering algorithm [12]. This algorithm set weights to the pixels for attenuating the intensity around the blocks' boundaries using a ramp style transition. As a result, the possible discontinuities are avoided, and uniform transitions are achieved.

The sizes of a given block and the overlapped volume are dependent on the properties of a particular DT, hence, in our algorithm they can be adjusted by the user. This characteristic makes our algorithm flexible and controllable. The boundary zone should be large enough to avoid mismatching features across the borders but at the same time, it should be small to be tolerant to the border constraints. Usually, the overlap volume is a small fraction of the block size, 1/6 of the total volume in our experiments.

In this approach, the overlap distance between the boundary zones of a given block $E_{B_{(x,y,t)}}$ and the output $E_{out}$ is estimated by using the L2 norm among the LBP values of each orthogonal plane. This error is defined as:

$$d(E_{B_{(x,y,t)}}, E_{out}) = \frac{1}{V}\sum_{i=1}^{V}\sum_{j=1}^{3}\left[p^j_{B_{(x,y,t)}}(i) - p^j_{out}(i)\right]^2 \qquad (4)$$

where $V$ is the number of pixels in the overlapped volume. $p^j_{B_k}(i)$ and $p^j_{out}(i)$ represent the LBP values of the $i^{th}$ pixel in the overlapping zones on the $j^{th}$ orthogonal plane, respectively. For color DTs, we compute the LBP-TOP code for each color channel. In this paper, we use the RGB color space, the final overlapping distance is the sum of the errors in each color component. The matching estimation between two blocks is computed based on their LBP values from the $V_{LBP-TOP}$ sequence. As a result, spatial and temporal features are considered simultaneously for the block description.

## 3 Experiments and Results

In this section, we present two series of tests that have been accomplished in order to evaluate the performance of our method. At first, a visual evaluation of performance is made on a variety of dynamic textures. Afterwards, comparisons between the proposed approach with other state-of-the-art methods are made to validate the application of it. All the resulting videos are available on the website: dl.dropbox.com/u/13100121/LBP2012Results.zip

### 3.1 Performance on a Variety of DTs

In the first experiment, a set of videos was selected for evaluating our approach performance on different types of dynamic textures. The videos were selected from the DynTex database [10], which provides a comprehensive range of high-quality DTs and can be used for various research purposes. In Figs. 5(a)-(f), a frame ($176 \times 120$ pixels size) taken from the original videos is shown. The selected sequences correspond to videos that show: spatio-temporal stationarity (a-c), a scene with a variety of textures and colors with different kind of dynamics (d) and a scene composed by structured objects (e-f).

In Figs. 5(g)-(l), the results of the synthesized outputs enlarged to $200 \times 200$ pixels size are presented. Spatial dimensions of the block $W_x \times W_y$ used for synthesis are shown below each image. As we said before, the size of the block is a user-specifiable parameter and should be proportional to the size of the spatial or temporal texture patterns. Here, the size of the overlapped volume $O_x \times O_y$ is 1/6 of $W_x \times W_y$. As we can observe in Figs. 5(g)-(i), our method preserves the spatio-temporal stationarity of the input and the borders between blocks are almost invisible. It is worth mentioning that in our method, we do not need to do an additional optimal seam on the borders to achieve smooth transitions, such as the graph cut used in [7]. This soft transition is achieved because of the selection of the blocks, based on the LBP-TOP features. The corresponding output for the video shown in Fig. 5(d) is presented in Fig. 5(j), where the same variety of colors and the diversity of surfaces is maintained. In this video, the transitions between blocks are also invisible. Sequences shown in Figs. 5(e)-(f) are different in the sense that they are composed of structured objects. Therefore, it is crucial that the structure of these objects can be maintained in the output, where we aim to generate an array of these objects. The synthesized results, seen in Figs. 5 (k)-(l), exhibit such arrangement showing that our method can keep the shape and structure of the given object without adding any discontinuity.

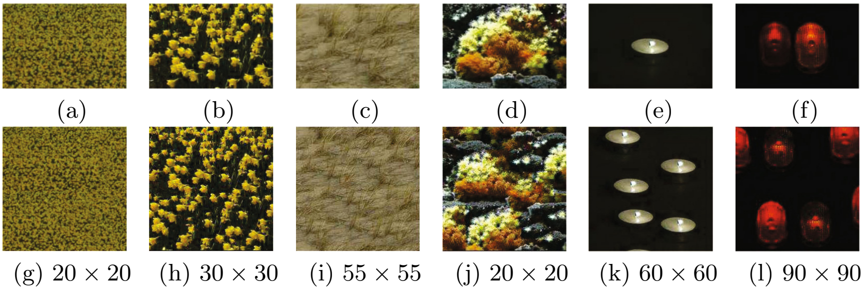



**Fig. 5.** Results of spatial synthesis. (a-f) A frame taken from the original sequence. (g-l) the corresponding synthesis result with the video block size used. The block size is proportional to the size of the spatial or temporal patterns.

### 3.2 Performance Comparison

The second experiment consist of a comparison with other state-of-the-art methods. We have compared our approach with the methods proposed by Wei and Levoy [14], Bar-Joseph *et al.* [1] and Kwatra et al. [7]. The firsts two are pixel-based approaches, while the third is a patch-based method.

We have borrowed the sequences named OCEAN and SMOKE (frame of $150 \times 112$ pixels size) used by Wei and Levoy in their experiments and made a comparison of the quality of the results. In Fig. 6 a frame extracted from the original sample, from the result of Wei and Levoy and from our result are presented. Here, it is observed that the videos obtained by Wei and Levoy (frame of $150 \times 112$ pixels size) are blurred, while the videos generated by our method (frame of $170 \times 170$ pixels size) keeps a natural appearance and motion of the two phenomena.

A second comparison is made with the results obtained by Bar-Joseph *et al.* [1]. We have used the sequences named as CROWD and JELLY FISH (frame of $256 \times 256$ pixels size). In Fig. 7, a frame from each resulting sequence is presented. Here, it is observed that the videos obtained by Bar-Joseph (frame of $256 \times 256$ pixels size) have some artifacts, blurred spots and discontinuities, while the videos generated by our method (frame of $280 \times 280$ pixels size) keep a natural look.

In a third comparison, we synthesized the sequence named RIVER (frame of $176 \times 112$ pixels size), provided by Kwatra *et al.* [7] for spatial extension. As it can be seen from both results (frame of $200 \times 150$ pixels size) shown in Fig. 8, Kwatra has generated good results of DT synthesis, which can be taken as the baseline to compare with. From the experimental results, we found that our method also achieves a good performance. It is observed that the appearance and dynamics of the water are preserved.

We have found that our method obtains very good results with sequences that present some spatial homogeneity, however we have detected limitations of our method on a very specific type of dynamic textures. Our approach does not work very well when a moving object occupies a big portion of the scene and thus,

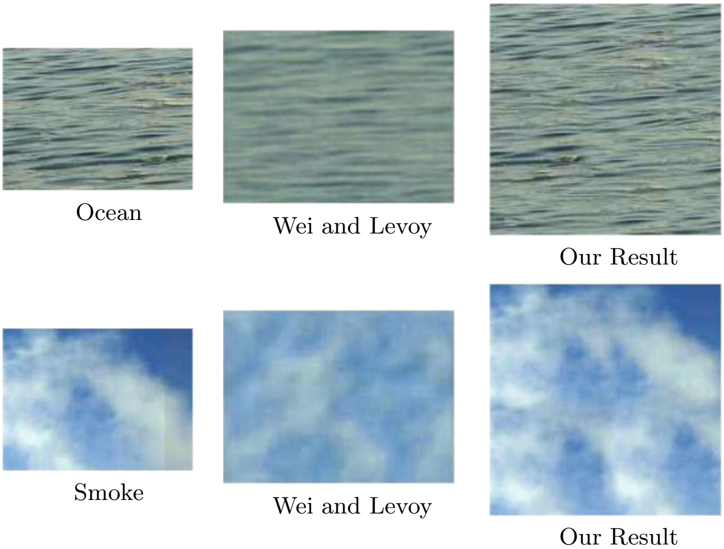



**Fig. 6.** Comparisons between the proposed approach and the method proposed by Wei and Levoy [14]

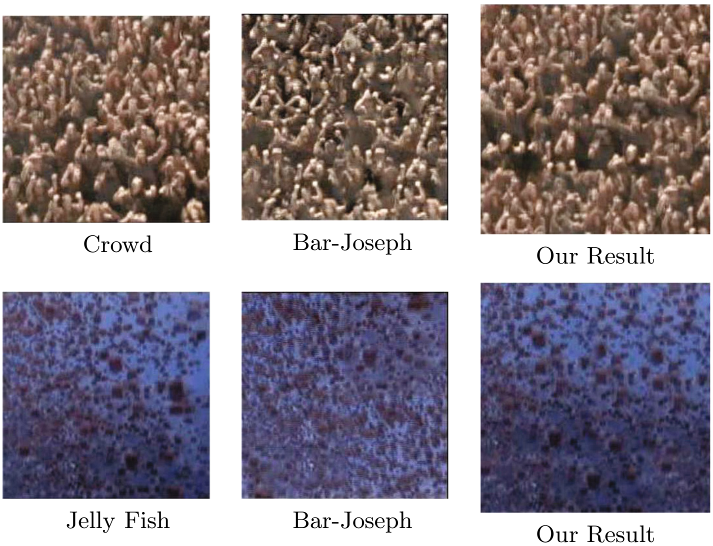



**Fig. 7.** Comparison between the proposed approach and the method proposed by Bar-Joseph *et al.* [1]


**Fig. 8.** Comparison between the proposed approach and the method proposed by Kwatra *et al.* [7]

there is not enough diversity to choose the blocks to be pasted on the output. Examples of this, using sequences form the DynTex database are shown in Fig. 9 where there is certain repeatability between the selected blocks, leading to discontinuities on the resulting videos.

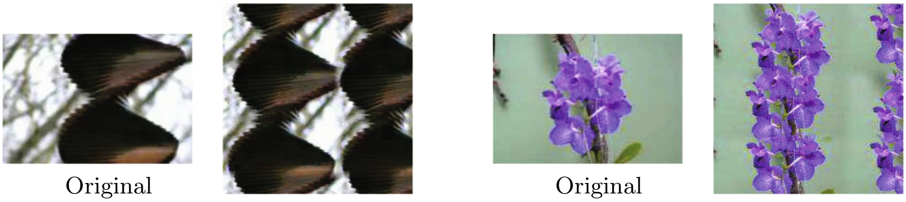



**Fig. 9.** Two examples where our method did not achieve good results as in the general case

## 4 Conclusions

In this paper, the use of spatio-temporal features for dynamic textures synthesis in space has been considered. The proposed approach is centered on synthesis in the spatial domain, unlike previous work that is mostly focused on temporal domain. This method explores a 3D patch-based synthesis, where the patch selection is accomplished by taking LBP-TOP features, instead of just making use of the intensity of pixels. LBP-TOP features can enhance the capability of describing the appearance and dynamics of DTs due to the local spatio-temporal patterns extracted. The main advantage of the presented approach is that it preserves on the output the visual similarity, dynamics and continuity of the input. Furthermore, no additional seam optimization is needed to achieve smooth transitions between blocks. From experimental results, the proposed method produces very good results on a variety of DTs. The performance of the proposed method has shown to be better than, or at least equal to other methods. In future work, the inclusion of a temporal domain synthesis approach will be considered.

**Acknowledgement.** The authors would like to thank the Academy of Finland, the Finnish CIMO and the Universidad de Guanajuato for the financial support.

## References

1. Bar-Joseph, Z., El-Yaniv, R., Lischinski, D., Werman, M.: Texture mixing and texture movie synthesis using statistical learning. IEEE Trans. on Visualization and Computer Graphics 7, 120–135 (2001)
2. Chetverikov, D., Peteri, R.: A brief survey of dynamic texture description and recognition. In: Proc. of the CORES 2005, vol. 30, pp. 17–26 (2005)
3. Constantini, R., Sbaiz, L., Susstrunk, S.: Higher order SVD analysis for dynamic texture synthesis. IEEE Trans. on Image Processing 17, 42–52 (2008)
4. Doretto, G., Jones, E., Soatto, S.: Spatially Homogeneous Dynamic Textures. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3022, pp. 591–602. Springer, Heidelberg (2004)
5. Ghanem, B., Ahuja, N.: Phase PCA for dynamic texture video compression. In: Proc. of the IEEE ICIP 2007, vol. 3, pp. 425–428 (2007)
6. Guo, Y., Zhao, G., Chen, J., Pietikäinen, M., Xu, Z.: Dynamic texture synthesis using a spatial temporal descriptor. In: Proc. of the IEEE ICIP 2009, pp. 2277–2280 (2009)
7. Kwatra, V., Schodl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: Image and video synthesis using graph cuts. ACM Trans. on Graphics 22, 277–286 (2003)
8. Liu, C.B., Lin, R.S., Ahuja, N., Yang, M.H.: Dynamic textures synthesis as nonlinear manifold learning and traversing. In: Proc. of the BMVC 2006, pp. 859–868 (2006)
9. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. on Pattern Analysis and Machine Intelligence 24, 971–987 (2002)
10. Peteri, R., Fazekas, S., Huiskes, M.J.: DynTex: A comprehensive database of dynamic textures. Pattern Recognition Letters 31, 1627–1632 (2010)
11. Schodl, A., Szeliski, R., Salesin, D., Essa, I.: Video textures. In: Proc. of the ACM SIGGRAPH 2000, pp. 489–498 (2000)
12. Szeliski, R., Shum, H.Y.: Creating full view panoramic image mosaics and environment maps. In: Proc. of the ACM SIGGRAPH 1997, pp. 251–258 (1997)
13. Wei, L.Y., Lefebvre, S., Kwatra, V., Turk, G.: State of the art in example-based texture synthesis. In: Eurographics 2009, EG-STAR, pp. 93–117 (2009)
14. Wei, L.Y., Levoy, M.: Fast texture synthesis using tree-structured vector quantization. In: Proc. of the ACM SIGGRAPH 2000, pp. 479–488 (2000)
15. Yuan, L., Wen, F., Liu, C., Shum, H.-Y.: Synthesizing Dynamic Texture with Closed-Loop Linear Dynamic System. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3022, pp. 603–616. Springer, Heidelberg (2004)
16. Zhao, G., Pietikäinen, M.: Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Trans. on Pattern Analysis and Machine Intelligence 29, 915–928 (2007)