Автоматическое создание динамических вставок для спецэффектов видео*

***К. Ватаначоте*[†], *В. Кумара*[†], *Т.К. Ши*[†], *А. Клименко*[‡]**
kanoksak.wattanachote@gmail.com|chinthakawk@gmail.com|timothykshih@gmail.com
andy.klimenko@gmail.com
[†]Факультет информатики и информационных технологий,
Центральный национальный университет, Чжунли, Тайвань
[‡]Институт физико-технической информатики, Протвино, Россия

Изменения в визуализации динамических текстур, приводящие к новому виду наблюдаемого движения и интенсивности определяется как динамическое преобразование текстур. Например, сцену водопада в видеоряде можно проявить в качестве текстуры огня (пожара) или наоборот. Эта статья предлагает новый способ для динамического преобразования текстур между двумя динамическими текстурами в видеоклипах. Целью работы было нахождение методов отображения динамических текстур с различной динамикой и визуализацией в видеоряде. Патч (фрагмент) перехода есть концепция с предположением о том, что две динамические текстуры могут иметь патчи (фрагменты), которые идентичны по внешнему виду, но отличаются по динамике. Патч (фрагмент) перехода состоит из двух модулей, модуль поиска патча (фрагмента) во-первых. Иррегулярной формы патч (фрагмент) используется чтобы найти конгруэнтные участки на исходной и целевой динамической текстуре. Второй модуль отвечает за соответствие патча (фрагмента) перехода результату алгоритма сопоставления шаблонов. Здесь мы предлагаем новый алгоритм сопоставления шаблонов, который допускает соответствия только в Текстуре Интереса (TOI), и два метода сегментации в реальном времени для поиска TOI. Экспериментальные результаты показывают, что каждый патч (фрагмент перехода) может осуществлять алгоритм поиска с эффективно повышенной точностью соответствия.

Ключевые слова: *динамическая текстура, трансформация динамической текстуры, шаблон соответствия, редактирование видео.*

Automatic Creation of Dynamic Patches in Special Effect Videos*

***Kanoksak Wattanachote*[†], *W.G.C.W. Kumara*[†], *Timothy K. Shih*[†], *Andrei Klimenko*[‡]**
[†]Dept. of Computer Science and Information Engineering,
National Central University, Zhongli, Taiwan (R.O.C.);
[‡]Institute of Computing for Physics and Technology, Protvino, Russia

Change in dynamic texture visualization to provide a new look both in dynamic appearance and intensity connectivity is given as the definition for dynamic texture transformation. For instance, a waterfall scene in a video can be appeared as a fire texture or vice versa. This paper proposes a novel method for dynamic texture transformation between two dynamic textures in video clips. The goal was to find out techniques to display dynamic texture with varying dynamics and appearances in a video. Patch transfer is a concept with the assumption of two dynamic textures can have a patch that is identical in appearance but different in dynamics. Patch transfer is composed of two modules, the patch searching module being the first. An irregular shaped patch is designed to find out the congruent regions on source and target dynamic textures. The second module is responsible for patch transfer by adapting to the output of the template matching algorithm. Here, we propose a new algorithm for template matching which allows the matching to be occurred only in the Texture of Interest (TOI), and two methods for real-time segmentation to find the TOI. The experimental results show that each patch can realize the matching algorithm with effectively enhanced matching accuracy.

Keywords: *dynamic texture, dynamic texture transformation, template matching, video editing.*

I. Introduction

Dynamic texture is described as a sequence of images in moving scenes that continuously exhibits movement of intensities in time, such as smoke, fire, water flow, etc. Changes in the intensities and their connectivity derived from other textures were studied to create a new visualization of dynamic textures.

Video editing has become a very important field with the wide adaptation of social media in the Internet and the developments of the film industry allowing users to manipulate their personally captured or other readily available video files in different ways. One aim of this finding was to widen the capabilities of video editing with increasing user experience and satisfaction. With the advances in the field, video editing is now being used in different new dimensions making it further interesting and creative going beyond the traditional

Работа опубликована при финансовой поддержке РФФИ, грант 15-07-20347

uses like fixing or restoring the damaged video scene. Nowadays, users are interested in creating their own video both in the production and the post-production process. Specially, in film industry, video editing can drastically reduce the risk of engaging in very dangerous outdoor acts by the actors themselves like with fire, smoke, water flood, wind storms, etc.

Dynamic texture segmentation methods described in Related Work section are accurate and robust, but not fast enough for real-time segmentation for our purpose. Hence, dynamic texture segmentation methods by color filtering and connected component filtering approach are proposed in this study. This paper studied the appearance pattern of a connectivity set and its motion on dynamic texture, based on a simple model of appearance matching with high accuracy measured by correlation coefficient statistics.

Section II discusses the related research works for this paper. Section III describes the algorithms for patch searching, while dynamic texture segmentation is devoted in section IV. In the section V patch transfer and corresponding algorithms are discussed. The experimental results of our study are demonstrated in section VI. Finally conclusions and future work are discussed in sections VII.

II. Related work

Aforementioned, at present video editing is an attractive research topic both in academics and industry leading to several purposes. For instance, video completion is one which has a powerful capability in fixing and restoring damaged videos to make the completed video natural and visually pleasant for human eyes. In recent years, several methods have been proposed for video editing.

In [1], Tsai *et al* proposed a video editing using motion inpainting which allows users to change the dynamic texture in video background. Dynamic texture such fire, smoke, water etc., could be edited by using patches in irregular shaped blocks to reproduce a realistic dynamic background [1]. In this vein, dynamic texture synthesis is a method to generate the dynamic background in the video. The synthesized background can be used to fill the background hole as proposed by Lin and Cheng [2]. In addition, editable dynamic textures [5] proposed by Doretto and Soatto serve a different purpose of video editing by synthesizing the dynamic texture to create the realistic sequences of images of dynamic scenes. Nonetheless, there is no existing method proposed for changing dynamic texture appearances.

Inspired by the works and experiments of Doretto *et al* [6], the changing of texture was proposed to study in two approaches. The first one is the changing of appearance, and the other one is the changing of dynamics. Consequently, dynamic texture transformation is studied in this paper to find the appropriate methods for changing dynamic textures which can be different in texture appearance and dynamics, between two video clips.

Further, to increase the performance, accuracy of changing and subjective quality assessment by human eyes, dynamic texture segmentation is inescapably considered in this paper. The following researches demonstrate various aspects of dynamic texture segmentation. In [3], Fazekas *et al* proposed a method for dynamic texture segmentation by using a level set scheme which three possible alternatives of gradient constancy, color constancy and brightness conservation were examined which aim to overcome the segmentation both on weak and strong dynamic texture. In [8], Phillips III *et al* combined color and motion information computed from video sequences to locate fire texture. Wang *et al* [9] proposed fire frame segmentation tool based on the Hue, Saturation and Intensity (HSI) color space model. However, the segmentation method in our purpose does not deal only fire frame, but also includes other dynamic textures (e.g., waterfall and smoke). The visual perception of the variation in HSV has been implemented by O'Malley *et al* [10], and Mazzeo *et al* [11], in different purposes, and has been found that the advantages of using HSV for visual perception are faster and easier in filtering both static and dynamic texture in video. Hence, in this study, we adopt color filtering method that is generally implemented based on the HSV color space, for the patch searching.

We propose the patch matching algorithms by adapting template matching whose various matching metrics are implemented to measure the minimal different intensity of patches. The concept of traditional template matching as found in Ouyang *et al* [1] and Delalandre *et al* [13] indicate that most matching algorithms are generally used in classifying an object by comparing portions of images with another image, and generally use Sum of Squared Differences (SSD) as the function for distance measurement.

III. Patch Searching

Patch searching is processed on a frame in the source video. A difficulty is finding out an irregular shaped patch on the dynamic texture. To overcome this difficulty, we applied connected component filtering approach to determine the irregular shaped patch and integrated the brightness difference value parameters to allow the user to arrange the size of the patch.

A. Connected Component Model.

In video processing, we describe the video as a volume of sequence images, represented by Φ^3. A patch is a region of connected pixels within the component, described by Ω^2 which can be found in an image or "frame" in the video denoted as Φ^2, where $\Phi^2 \in \Phi^3$

and $\Omega^2 \in \Phi^2$. A pixel is described by I as a matrix consists of three components of intensity as in (1). Each of whose is the intensity of the independent red, green and blue components. The luminance is weighted sum of three corresponding linear intensity values as in (2), described as a grayscale value (I_y).

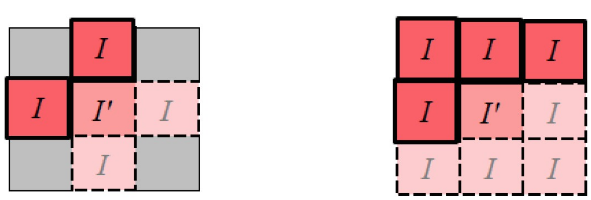
$$I = [I_r, I_g, I_b]^T \tag{1}$$

$$I_y = 0.299 \times I_r + 0.587 \times I_g + 0.114 \times I_b \tag{2}$$

The components in our proposed method are considered in grayscale values range. Hence I and I'_p in the following description mean in grayscale values range calculated by (2). Let I_s represents a given pixel to find out a patch called "seed", obviously $I_s \in \Omega^2$. I'_p represents a component pixel of Ω^2, $I'_p \in \Omega^2, p = 0, 1, 2, \ldots, k$. A given I_s is considered to be the first component of patch, $I_s = I'_0 \in \Omega^2$. Connected component is a concept of finding out a region of connected pixels starting from a given pixel I_s where all pixels within a component set have close values . Let L is a maximal lower brightness difference between the currently observed pixel I and the component I'_p. Let U is a maximal upper brightness difference between the currently observed pixel I and the component I'_p, where $\{L, U\} \in \mathbb{R}$, floating range, and I represents a pixel at (x, y) in Φ^2. The pixel I is considered to belong to the component, $k + 1$, if its value on floating range meets the following conditions for any p of k components.

$$I'_p - L \leqslant I \leqslant I'_p + U \tag{3}$$

According to the connected component labeling algorithm [14], we prefer two connectivity models for finding the component, as shown in Fig. 1, namely 4-connectivity and 8-connectivity. The 4-connectivity represents the path that is carried out by checking the labels of pixels that are located in the top and left side of I', whereas the 8-connectivity represents the path that is carried out by checking the label of pixels on the left, top-left, top and top-right side of I'.



(a) 4-connectivity. (b) 8-connectivity.

Figure 1: The 4-connectivity and 8-connectivity models for two passes connected component labeling algorithm.

The algorithm provides two passes over the frame. The first pass is to record equivalences and assign temporary labels whereas the second pass is to replace each temporary label by the label of its equivalence class [14].

B. Patch Searching Algorithm.

A patch is derived from a given pixel given by user or system. After a seed point was assigned on the texture, all similar value neighbors are considered to be the component.

According to the concept in Section III - A, patch searching algorithm can be processed as described in the following steps

Step 1: Determine the seed point I_s on the dynamic texture in a video frame.

$$I_s \in \Phi_i^2,$$

where i is a frame number in a video, $i \in \mathbb{N}_0$.

Step 2: Arrange the connected component size by adjusting L and U values, and define the connectivity model by choosing an appropriate connectivity, between 4-connectivity and 8-connectivity.

Step 3: Save a satisfied patch for the upcoming process. Since Ω^2 represents the patch, then lets say:

$$I_s \in \Omega^2 \text{ and } \Omega^2 \subset \Phi_i^2,$$

where $\{\Phi_0^2, \Phi_1^2, \Phi_2^2, \ldots, \Phi_i^2, \ldots, \Phi_n^2\} \in \Phi^3, n$ denotes the last frame in the video clip.

IV. Dynamic texture segmentation

It is difficult to specify an appropriate segmentation method for dynamic textures due to its discrepancy nature. We propose two real-time segmentation methods for dynamic texture in the following sections.

A. Segmentation by Color Filtering.

Color filtering uses concept of color tracking which is frequently used in computer vision and robotics related applications for fast object tracking and object chasing. Color tracking is done based on the Hue, Saturation and Value or Intensity (HSV) color space. HSV values describe a different color for each basis RGB space. The HSV with emphasis on the visual perception of the variation in Hue, Saturation and Intensity values of an image pixel has been implemented for segmentation by Sural *et al* [7]. The advantages of color tracking are faster results and easier for tracking both static and dynamic texture from the image or video frame.

Hence, this approach is applied for dynamic texture segmentation for creating TOI in our study which aims for real-time segmentation. We modeled the algorithm as follows:

Step 1: Convert frame Φ_i^2, to the HSV color space.

Step 2: Filter out the result by determining the inclusive lower boundary and exclusive upper boundary in three channels of Hue, Saturate, and Value, in the integer range e.g., 0–255. Let L represents the lower boundary of each channel, whilst U represents the upper boundary of each channel and use these parameters to filter the elements of Φ_i^2, by comparing with I component's values in HSV space. Where I is a pixel basis on HSV color space at $(x, y) \in \Phi_i^2$, pixel I will be considered to be a component of $TOI, I \in TOI$, if and only if:

$$L_H \leqslant I_H < U_H \tag{4}$$

$$L_S \leqslant I_S < U_S \tag{5}$$

$$L_V \leqslant I_V < U_V \tag{6}$$

where H indicates the values considered in Hue space (0-255), S indicates the values considered in Saturate space (0–255), and V indicates the values considered in Value space (0-255).

Step 3: Convert TOI to RGB color space, then we get the final result of TOI denoted as T^2, where $T^2 \subseteq \Phi_i^2$.

To increase the accuracy and performance of patch transfer, there are two purposes of the use of T^2. In one hand, T^2 is used to increase the accuracy of matching on the right position by limiting patch transfer target only in T^2. On the other hand, T^2 is also aimed to reduce the duration time for matching method which a ton of statistical metrics are calculated. With the T^2, the calculation is allowed to be occurred only in the considered region T^2. Fig. 3 illustrates the path of a match searching algorithm. As a result, this idea leads to our template matching algorithm in Section V.

(a) Original frame. (b) Color filtering result. (c) Connected component result.

Figure 2: The segmentation results comparison, the results derived from the different segmentation approaches.

B. Segmentation by Connected Component.

This segmentation approach is proposed to assert the improvement in segmentation for some dynamic textures which failed in color filtering segmentation approach. For instance in Fig. 2(b), some parts of railway sidewalk were detected with smoke detection purpose.

In order to improve the dynamic texture segmentation for creating TOI, a concept as described in the Section III - B for patch searching algorithm is applied. The target dynamic texture segmentation algorithm for TOI is described here in the following.

Step 1: Determine the seed point I_s on the dynamic texture in a video frame.

Step 2: Arrange the connected component size by choosing an appropriate connectivity in Fig. 1 for two passes algorithm, and then determine L and U values until the red mark area is covered all connected part of dynamic texture.

Step 3: Save the satisfied result as T^2 for the upcoming process, and $T^2 \subseteq \Phi_i^2$.

V. Patch transfer

A. Template Matching Approach.

The classical template matching method is generally used to find a match region on an image that match or similar to a template or patch. A traditional template matching algorithm can be described as:

Step 1: Store a source image in the system.

Step 2: Store a template in the system.

Step 3: Find out a best matched result by passing the patch one pixel at a time, line by line, from left to right and top to bottom, see Fig. 3(a). At each point, a convolution of patch over the source image is calculated and a matched metric result on each point is stored. The best matched result is analyzed, based on a statistics method.

Step 4: Indicate the matched region on the source image.

(a) (b)

Figure 3: Describes the match routing, (a) illustrates the classical match routing, and (b) is our match searching model.

B. Target Template Matching Algorithm.

In this section, we focus attention on target dynamic texture detection. A new template matching method is applied for patch transfer algorithm to increase the accuracy and performance of matched result. Although an idea of a template should not include the background information to avoid the influence of the background changes was studied by Sha *et al.* [4].

Indeed, for dynamic texture patch searching, not only excluding the template background, but also should eliminate the background of dynamic texture in source image to get rid of a match occurred in the background location. The routing path of our template matching algorithm is illustrated in Fig. 3(b).

The target template matching algorithm is described as follow:

Step 1: Store the first video Φ_1^3 in the system.
Step 2: Store the second video Φ_2^3 in the system.
Step 3: Determine the Ω^2 on frame Φ_{2j}^2 in Φ_2^3 by using patch searching algorithm in subsection III-B, where $\Phi_{2j}^2 \in \Phi_2^3$ and $j \in \mathbb{N}_0$.
Step 4: Determine the T_2 on the frame Φ_{1i}^2 in Φ_1^3 by using a segmentation algorithm in Section IV, where $\Phi_{1i}^2 \in \Phi_1^3$ and $i \in \mathbb{N}_0$.
Step 5: To find the matched region between Φ_{2j}^2 and Φ_{1i}^2, pass the Ω^2 over Φ_{1i}^2 which the location is verified by T^2. The metric is stored in the result matrix R. Each $R(x,y)$ contains the match matrix which calculated by a statistics method. The Normalized Sum Squared Differences (NSSD), Cross Correlation (XCOR) and Normalized Cross Correlation (NCC) are used for match metric in our experiment. R matrix calculated by NSSD match metric is shown in (7).

$$R(x,y) = \frac{\sum_{xy}(P(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{xy} P(x',y')^2 \cdot \sum_{xy} I(x+x',y+y')^2}} \tag{7}$$

R matrix calculated by XCOR and NCC match metrics is shown in (8) and (9) respectively.

$$R(x,y) = \sum_{x'y'} P(x'y') \cdot I(x+x',y+y')^2 \tag{8}$$

$$R(x,y) = \frac{\sum_{x'y'}(P(x'y') \cdot I(x+x',y+y'))}{\sqrt{\sum_{x'y'} P(x'y')^2 \cdot \sum_{xy} I(x+x',y+y')^2}} \tag{9}$$

where R represents a match metric value at (x,y) which is calculated by $\sum$ operation on Ω^2 and Φ_{1i}^2, when Ω^2 is passed on Φ_{1i}^2, every considered pixels on Ω^2 and Φ_{1i}^2 at the location T^2 are used in the calculation. Let P denotes as a pixel on Ω^2 at (x',y') and I represents a pixel on Φ_{1i}^2.
Step 6: Indicate the matched region on Φ_{1i}^2, where $i,j \in \mathbb{N}_0$ and i,j are independent, and represent a frame number in the video clips.

C. Patch Transfer Algorithm.

Our new algorithm to accomplish patch transfer is discussed in this section. This algorithm combines corresponding algorithms explained in the previous sections. There are two primary components for patch transfer between two video clips as follows.

A source video, where Φ_1^3 represents a source video and Φ_1^2 represents frame in the source video, $\Phi_1^2 \in \Phi_1^3$. A target video, represented by Φ_2^3, and Φ_2^2 represents frame in the target video, where $\Phi_2^2 \in \Phi_2^3$. The dynamic textures in this experiment are assumed to be continuously appeared in every scene in the video clips. The algorithm for patch transfer between two video clips is concluded as below:

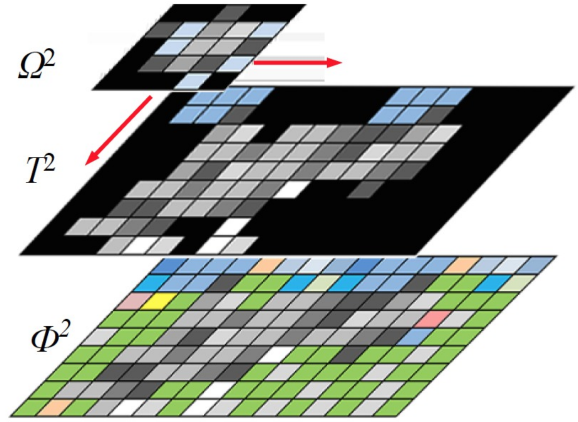


Figure 4: Demonstrates the target template matching components.

Step 1: Store first video Φ_1^3 in the system, where Φ_1^3 contains f_1 frames.
Step 2: Store the second video Φ_2^3 in the system, where the Φ_2^3 contains f_2 frames,f_1 and f_2 are independent.
Step 3: Determine the Ω^2 on frame Φ_{1i}^2 in Φ_1^3 by using patch searching algorithm in subsection III-B, where $\Phi_{1i}^2 \in \Phi_1^3$ and $i \in \mathbb{N}_0$. In this step the system allows user to define more than one patch to cover all area of TOI.
Step 4: Determine the number of output frame for new videos. Let f represents the total number of frames for the output video, and k represents the frame number in the new video, where $k = = 0,1,\dots,f$ and $k \in \mathbb{N}_0$.
Step 5: Determine the TOI or T^2 on frame Φ_{1i}^2 in the Φ_1^3 by using a segmentation algorithm in section IV.
Step 6: Pass the Ω^2 over Φ_{1i}^2 verified by T^2, the metric is stored in the result matrix R. Each $R(x,y)$ contains the match matrix which calculated by a statistics method in (7), (8) or (9).
Step 7: Indicate the matched region on Φ_{1i}^2 verified by T^2. Let M_{1i}^2 stores the best matched region matrix on Φ_{1i}^2.
Step 8: Pass M_{1i}^2 over the Φ_{2j}^2, for each considered location, the metric is stored in the result matrix R. Each R(x, y) contains the match matrix which calculated by a statistics method in Step 6, where $\Phi_{2j}^2 \in \Phi_2^3$, $j \in \mathbb{N}_0, i, j$ and k are independent.
Step 9: Indicate the matched region on Φ_{2j}^2 and supposeM_{2j}^2 stores the best matched region matrix on Φ_{2j}^2.
Step 10: Transform the data between M_{1i}^2 and M_{1j}^2.
Step 11: Here, if $i < f_1$ then increase i by 1 else set $i = 0$ and if $j < f_2$ then increase j by 1 else set $j = 0$, where f_1, f_2 and $f \in \mathbb{N}$, and $f1$, $f2$, $f > 0$. Stop if $k = f$, else increase k by 1, and go to Step

5 if the T^2 is purposed to be dynamic shape, or go to Step 6 if the T^2 is purposed to be fixed shape.

VI. Experimental Result

We developed a tool for this study to manipulate the dynamic texture transformation. The real-time output can be monitored by the tool, as shown in Fig. 5.

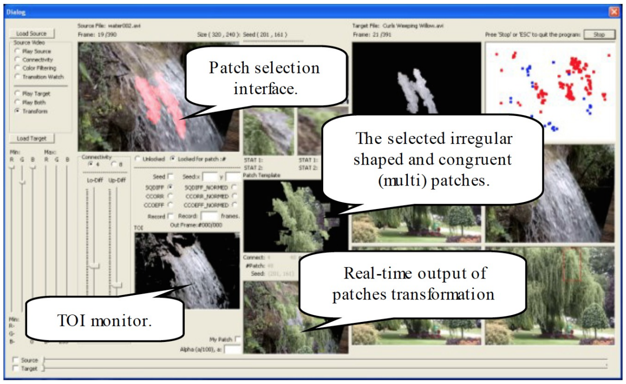
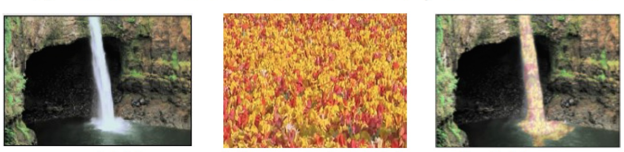


Figure 5: Our tool used to monitor real-time results for patches transformation.

The experiment worked on the video resolution $320 \times \times 240$. More than fifty video clips were tested to study the dynamic texture characteristics and to change their new look. The successful results are shown in Fig. 6.

(a) New waterfall texture derived from fire texture.

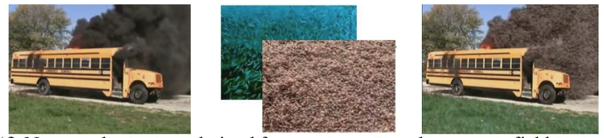
(b) New water flow texture derived from curly leafs tree texture.



(c) New waterfall texture derived from tulip field.

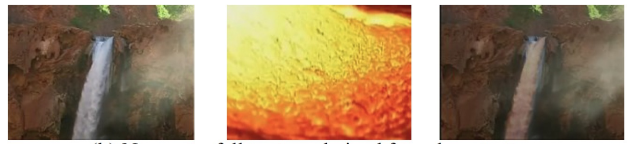
(d) New smoke texture derived from ants nest texture.

(e) New grass field texture derived from baby fish cloud.



(f) New smoke texture derived from ants nest, and new grass field texture derived from baby fish cloud.

(g) New rocket's smoke texture derived from tulip field.



(h) New waterfall texture derived from lava texture.

(i) New lava texture derived from baby fish cloud.

(j) New waterfall texture derived from red lava texture.

(k) New train smoke texture derived from ants nest texture.

(l) New hair color texture derived from red lava texture.

(m) New color smoke texture derived from ants nest texture.

Figure 6: Demonstration of dynamic texture transformation results; the first two columns show the original textures; the last column shows the new look.

VII. Conclusions and future work

In this paper, we presented an idea for transferring a patch of dynamic texture aiming to create a new look of dynamic texture. Several new algorithms were implemented in this study. We propose a simple algorithm for patch searching. Two real-time segmentation techniques were implemented to serve patch transfer on video clips. The target template matching algorithm was implemented to effectively enhance the patch transfer. In the future works, we plan to investigate more elaborate techniques for varying in dynamic texture transformation. Both spatial and temporal information in dynamic texture will be integrated to our study to enhance the transformation. Blending techniques will be investigated to increase higher quality results.

Bibliography

[1] *Tsai J.C., Shih T.K., Wattanachote K., Li K.C.* Video editing using motion inpainting // Information Networking and Applications (AINA), 2012. – pp.649–654.

[2] *Lin C.W. and Cheng N.C.* Video background inpainting using dynamic texture Synthesis // IEEE International Symposium Circuits and Systems (ISCAS), 2010. – pp.1559–1562.

[3] *Fazekas S., Amiaz T., Chetverikov D. and Kiryati N.* Dynamic texture detection based on motion analysis // International Journal of Computer Vision, 2009. – Vol.82 No.1, pp.48–63.

[4] *Sha S., Jianer C. and Sanding L.* A fast matching algorithm based on K-degree template // 4th International Conference on Computer Science & Education (ICCSE), 2009. – pp.1967–1971.

[5] *Doretto G. and Soatto S.* Editable dynamic textures // Computer Vision and Pattern Recognition (CVPR) IEEE Computer Society Conference, 2003. –Vol.2, pp.II — 137–42.

[6] *Doretto G., Cremers D., Favaro P. and Soatto S.* Dynamic texture segmentation // 9th IEEE International Conference on Computer Vision, 2003. – Vol.2, pp.1236–1242.

[7] *Sural S., Qian G. and Pramanik S.* Segmentation and histogram generation using the HSV color space for image retrieval // International Conference on Image Processing, 2002.Vol.2, pp.II – 589 – II–592.

[8] *Phillips III W., Shah M. and Lobo N. de V.* Flame recognition in video // 5th IEEE Workshop on Applications of Computer Vision, 2000. – pp.224–229.

[9] *Wang L.M., Yan X.L. and Bu L.P.* An edge growth segmentation arithmetic based on HSI color space modeling // dentification & Control (ICMIC), 2012. – pp.126–130.

[10] *O'Malley R., Jones E. and Glavin* Rear-lamp vehicle detection and tracking in low-exposure color video for night conditions // Intelligent Transportation Systems,2010. Vol.11(2), pp.453–62.

[11] *Mazzeo P.L., Giove L., Moramarco G.M., Spagnolo P. and Leo M.* HSV and RGB color histograms comparing for objects tracking among non overlapping FOVs, using CBT F // Advanced Video and Signal-Based Surveillance (AVSS), 2011. – pp.498–503.

[12] *Ouyang W.L., Zhang R.Q. and Cham W.K.* Fast pattern matching using orthogonal Haar transform // Computer Vision and Pattern Recognition (CVPR)2010. – pp.3050–3057.

[13] *Delalandre M., Iwata M. and Kise K.* Fast and optimal binary template matching application to manga copyright protection document analysis systems (DAS) // IAPR, 2014. – pp.298–303.

[14] *Shapiro L.G. and Stockman G.C.* Computer Vision // Prentice Hall, 2002. – pp.69–73.