Time-Varying Textures: Definition, Acquisition, and Synthesis

Sebastian Enrique[*]   Melissa Koudelka[†]   Peter Belhumeur[*]   Julie Dorsey[†]   Shree Nayar[*]   Ravi Ramamoorthi[*]

[*]Columbia University   [†]Yale University

{ senrique | belhumeur | nayar | ravir } @ cs.columbia.edu   mkoudelka@msn.com, dorsey@cs.yale.edu

Figure 1. Sample images of a time sequence with synthesized time-varying textures for cracking paint and growing grass.

1 Introduction and Definition

Textures have usually been considered static—the surface itself remains constant through time. Many real-world textures are of interest, however, expressly because of the way their appearance changes or evolves with time. Consider the fracture of mud drying in a riverbed, the growth of grass on a hillside, or the formation of oxides on copper. Each of these natural processes forms a pattern over time, often producing striking effects. In this work we developed an example-based approach to model these processes. We use time-lapse images to capture them and we then synthesize larger spatial patterns of these processes using a new time-varying texture synthesis algorithm (Figure 1).

The Time-Varying Texture Function (TTF) concept is a simple extension of static texture, and we can denote the time-varying texture p as $TTF = p(x, y, t)$, where (x, y) is the corresponding pixel in the image, and t is the time. Time-varying extensions for the BRDF and BTF ([Dana et al. 1999]) functions are also possible.

2 Acquisition and Synthesis

We acquire images using time-lapse photography, with controlled lighting and view conditions, recording photographs at given time intervals. These ranged from five seconds (paint drying and cracking, snow accumulating on slate) to between 1 and 10 minutes (grass growth, oxidating copper, ripening banana), and the durations over which the images were captured ranged from twenty minutes to several days (Figure 2). In all, we obtained between 200 and 2500 images for each time-varying process.

We developed a simple algorithm that enables image quilting [Efros and Freeman 2001] to be adapted to time-varying textures. We choose to quilt blocks that have a small spatial extent as in the original algorithm, but *extend over the entire time sequence*. Consider the input time-varying texture $p(x, y, t)$. The error metric that we use in the quilting comparison phase is

$$E = \sum_t \sum_{x,y} (p_1(x,y,t) - p_2(x,y,t))^2 ,$$ where $p1$ and $p2$ are small

blocks. When we paste a block into the synthesized texture, we paste the same spatial block for the entire time sequence, preserving the original time variation.

If we have acquired one thousand frames, TTF synthesis will be one thousand times slower than synthesizing a single static texture. Memory requirements will also grow linearly with the number of frames. We use a linear SVD method applied to the problem of representing variation with time, to
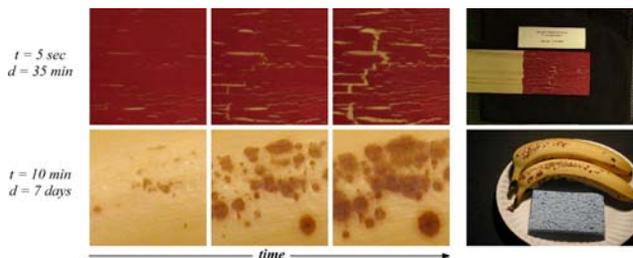

Figure 2. Left: two examples from our database of time-varying textures, cracking paint and ripening banana. t is the elapsed time and d is the duration of the entire capture process. Right: images from the controlled acquisition phase of both textures.

speed up the comparison phase of the algorithm using a rank k for the SVD output. It should be emphasized that this compression only optimizes the comparisons but the final output always copies blocks directly from the original input without compression.

In some cases, the nature of time-varying textures leads to specific visual artifacts not usually found in static texture synthesis. The simple approaches that we have used to improve synthesis quality are: elongated rectangular blocks (especially useful for time-varying textures like grass growing); jittering the time sequence (we jitter according to a smooth function for each block to avoid simultaneous repetitions of texture features); image processing of input data (to avoid low-frequency gradients that cause blocky appearances in the output images).

To control the time progression of a TTF, we resample the data based on some property of the surface or environment. The desired resample rate is represented by a map $M(x, y)$, which is a function at each point on the target surface. For each pixel in each frame of the output sequence, the source frame from the original sequence, $f_t(x, y)$, for that pixel is computed to implement the resampling: $f_t(x, y) = t\, M(x, y)$, where t is the frame number in the output sequence, and $M(x, y)$ is the resample factor for that particular pixel. The map M can be a function of any surface or environmental parameter.

References

DANA K., VAN GINNEKEN B., NAYAR S., KOENDERINK J.: Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics 18*, 1 (January 1999), 1–34.

EFROS A., FREEMAN W.: Image quilting for texture synthesis and transfer. In *SIGGRAPH* (Los Angeles, CA, August 2001), pp. 341–346.