Editable Dynamic Textures

Gianfranco Doretto*
UCLA Computer Science Department

Stefano Soatto†
UCLA Computer Science Department

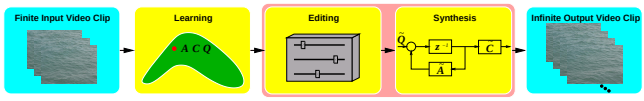
Introduction

This technical sketch presents a simple and efficient algorithm for editing realistic sequences of images of dynamic scenes that exhibit some form of temporal regularity. Such scenes include flowing water, steam, smoke, flames, foliage of trees in wind, crowds, dense traffic flow etc. We call this kind of scenes *dynamic textures*.

Traditionally the problem is approached either by *physics-based* (PHB) techniques [Popović et al. 2000] or by *image-based* (IMB) techniques [Schödl et al. 2000]. Here we propose an IMB technique, based on the work of [Soatto et al. 2001], that relies on a statistical model of the visual signal. We say that a sequence of images $\{I(t)\}_{t=1\ldots\tau} \in \mathbb{R}^m$ represents a *linear dynamic texture* if it is a realization of a second-order stationary stochastic process. If we call $y(t) = I(t) + w(t)$ a sequence of images corrupted by white, zero-mean Gaussian noise $\{w(t)\} \in \mathbb{R}^m; w(t) \sim \mathcal{N}(0, Q)$, the assumption of second-order stationarity of $\{y(t)\}$ corresponds to the existence of a positive integer n, a process $\{x(t)\} \in \mathbb{R}^n$ with initial condition $x_0 \in \mathbb{R}^n$ and symmetric positive-definite matrices $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ such that

$$\begin{cases} x(t+1) = Ax(t) + v(t) & x(0) = x_0 \;;\; v(t) \sim \mathcal{N}(0, Q) \\ y(t) = Cx(t) + w(t) & w(t) \sim \mathcal{N}(0, R) \end{cases} \quad (1)$$

with $I(t) = Cx(t)$, for some matrices $A \in \mathbb{R}^{n\times n}$ and $C \in \mathbb{R}^{m\times n}$.



System Overview

Referring to the figure above, the overall system takes two types of input and produces one output. The inputs are (a) a finite video clip and (b) a set of editing parameters. The output of the system is a potentially infinitely long video of a dynamic texture similar to the original one, that can be interactively modified by editing the parameters (b).

The system is composed of several "modules." The first module, *learning*, takes as input a finite, noisy sequence of images $\{y(1), \ldots, y(\tau)\}$ and returns an estimate of the model parameters $\hat{A}$, $\hat{C}$, $\hat{Q}$, $\hat{R}$. This is borrowed from the prior work of [Soatto et al. 2001].

Once a model has been learned, new sequences can be trivially generated by the *synthesis* module which simulates the model (1). This entails choosing an initial condition $\hat{x}_0$ (for instance one of the original images), drawing a sample of an IID Gaussian process with covariance $\hat{Q}$, performing one step of the iteration $\hat{x}(t+1) = \hat{A}\hat{x}(t) + \hat{v}(t)$ and computing the new synthesized image as $\hat{I}(t) = \hat{C}\hat{x}(t)$. However, one would like to insert an *editing* module before the synthesis. This is described in the next section.

Editing

This module allows the editor to (1) simulate a novel sequence that has the same spatio-temporal statistics as the input video clip, and to (2) interactively modify the spatio-temporal characteristics of the simulation in order to achieve the desired perceptual effect. In principle, any modification of the system parameters, for instance $\tilde{A}, \tilde{C}, \tilde{Q}$, results in a novel synthetic sequence whose spatio-temporal statistics is altered with respect to the original sequence. Unfortunately, naive manipulation of the model parameters rarely results in sequences that have any resemblance with reality. Therefore, this module allows manipulating groups of parameters while enforcing causality, stability, and other constraints that are necessary to yield a realistic outcome.

The sketch shows how to change the spatial scales by deforming the actual sub-space spanned by the columns of C. This is done by substituting the matrix $\hat{C}$ with the matrix $\tilde{C} \doteq \hat{C}W$, where $W \in \mathbb{R}^{n\times n}$ is a diagonal matrix with the non-negative real numbers $w_1, \ldots, w_n$ as its diagonal entries. To alter the speed of a synthesized video clip, we proved that the frequencies associated with the poles of the system matrix A have to be multiplied by the same constant factor Ω. Therefore, each normalized frequency ψ_i, $i = 1, \ldots, n$, is replaced by $\tilde{\psi}_i \doteq \Omega\psi_i$. In order to reverse the time axis, so as to invert the visual flow of a given dynamic texture, while maintaining the same temporal dynamics (speed), all we need to do is to substitute $\hat{A} = V\Lambda V^{-1}$ with $\tilde{A} = V^*\Lambda V^{*-1}$, where Λ and V respectively are the diagonal matrix of eigenvalues and the matrix whose columns are the corresponding eigenvectors of $\hat{A}$, and * denotes the complex conjugate. Finally, notice that for a non-zero Q, the input $v(t)$ excites the modes of the state $x(t)$ and causes it to evolve as a discrete-time Brownian motion. The intensity of the driving noise determines how far away from the initial condition the Brownian motion travels. We show experiments where we rescale the noise components by a positive constant α, i.e. we substitute $\hat{Q}$ with $\tilde{Q} \doteq \alpha\hat{Q}$.

The sketch shows examples of dynamic textures like "Ocean," "Fountain," "Fire," and "Smoke," and their sped up, slowed down, and time reversed versions, incorporated with effects induced by spatial scale and intensity changes.

References

Popović, J., Seitz, S. M., Erdmann, M., Popović, Z., and Witkin, A. 2000. Interactive manipulation of rigid body simulations. In *Proceedings of SIGGRAPH 2000*, 209–218.

Schödl, A., Szeliski, R., Salesin, D. H., and Essa, I. 2000. Video textures. In *Proceedings of SIGGRAPH 2000*, 489–498.

Soatto, S., Doretto, G., and Wu, Y. 2001. Dynamic textures. In *Proceedings of IEEE International Conference on Computer Vision*, vol. 2, 439–446.

*e-mail: doretto@cs.ucla.edu
†e-mail: soatto@ucla.edu