Weather Forecasting for Dummies™

Let’s predict weather:
- Given today’s weather only, we want to know tomorrow’s
- Suppose weather can only be {Sunny, Cloudy, Raining}

The Weather Channel algorithm:
- Over a long period of time, record:
  - How often S followed by R
  - How often S followed by S
  - Etc.
- Compute percentages for each state:
  - $P(R|S)$, $P(S|S)$, etc.
- Predict the state with highest probability!
- It’s a Markov Chain
Markov Chain

What if we know today and yesterday’s weather?

Text Synthesis

[Shannon,’48] proposed a way to generate English-looking text using N-grams:
• Assume a generalized Markov model
• Use a large text to compute prob. distributions of each letter given N-1 previous letters
• Starting from a seed repeatedly sample this Markov chain to generate new letters
• Also works for whole words

WE NEED TO EAT CAKE
Mark V. Shaney (Bell Labs)

Results (using alt.singles corpus):

• “As I’ve commented before, really relating to someone involves standing next to impossible.”
• “One morning I shot an elephant in my arms and kissed him.”
• “I spent an interesting evening recently with a grain of salt”
Still photos

Video clips
Video textures

Problem statement

video clip → video texture
Our approach

• How do we find good transitions?

Finding good transitions

• Compute $L_2$ distance $D_{i,j}$ between all frames $i$ vs. $j$.

Similar frames make good transitions
Markov chain representation

Similar frames make good transitions

Transition costs

- Transition from i to j if successor of i is similar to j
  - Cost function: $C_{i\to j} = D_{i+1, j}$

\[ \begin{array}{c c c c}
\text{i} & \text{i+1} & \text{j-1} & \text{j} \\
\text{i} & \cdot & \cdot & \cdot \\
\text{j-1} & \cdot & \cdot & \cdot \\
\text{j} & \cdot & \cdot & \cdot \\
\end{array} \]
Transition probabilities

• Probability for transition $P_{i \rightarrow j}$ inversely related to cost:
  
  $$P_{i \rightarrow j} \sim \exp \left( - \frac{C_{i \rightarrow j}}{\sigma^2} \right)$$

Preserving dynamics
Preserving dynamics

\[ C_{i \rightarrow j} = \sum_{k = -N}^{N-1} w_k D_{i+k+1, j+k} \]
Preserving dynamics – effect

• Cost for transition $i \rightarrow j$
  \[ C_{i \rightarrow j} = \sum_{k = -N}^{N-1} w_k D_{i+k+1, j+k} \]

Dead ends

• No good transition at the end of sequence
Future cost

• Propagate future transition costs backward
• Iteratively compute new cost
  • $F_{i\rightarrow j} = C_{i\rightarrow j} + \alpha \min_k F_{j\rightarrow k}$
**Future cost**

- Propagate future transition costs backward
- Iteratively compute new cost
  
  \[ F_{i \rightarrow j} = C_{i \rightarrow j} + \alpha \min_k F_{j \rightarrow k} \]
**Future cost**

- Propagate future transition costs backward
- Iteratively compute new cost
  
  \[ F_{i\rightarrow j} = C_{i\rightarrow j} + \alpha \min_k F_{j\rightarrow k} \]

- Q-learning

**Future cost – effect**
Finding good loops

• Alternative to random transitions
• Precompute set of loops up front

Visual discontinuities

• Problem: Visible “Jumps”
Crossfading

- Solution: Crossfade from one sequence to the other.

Morphing

- Interpolation task:
Morphing

• Interpolation task:
  \[
  \frac{2}{5} A + \frac{2}{5} B + \frac{1}{5} C
  \]

• Compute correspondence between pixels of all frames

Morphing

• Interpolation task:
  \[
  \frac{2}{5} A + \frac{2}{5} B + \frac{1}{5} C
  \]

• Compute correspondence between pixels of all frames
• Interpolate pixel position and color in morphed frame
• based on [Shum 2000]
Results – crossfading/morphing

Jump Cut  Crossfade  Morph
Video portrait

• Useful for web pages

Region-based analysis

• Divide video up into regions

• Generate a video texture for each region
Automatic region analysis

Video-based animation

• Like sprites computer games
• Extract sprites from real video
• Interactively control desired motion
**Video sprite extraction**

- blue screen matting
- and velocity estimation

**Video sprite control**

- Augmented transition cost:

\[
C_{i\rightarrow j}^{\text{Animation}} = \alpha C_{i\rightarrow j} + \beta \angle
\]

- Similarity term
- Control term

- vector to mouse pointer
- velocity vector
Interactive fish

Lord of the Flies
Summary

• Video clips → video textures
  • define Markov process
  • preserve dynamics
  • avoid dead-ends
  • disguise visual discontinuities

Texture

• Texture depicts spatially repeating patterns
• Many natural phenomena are textures

radishes  rocks  yogurt
Texture Synthesis

• Goal of Texture Synthesis: create new samples of a given texture
• Many applications: virtual environments, hole-filling, texturing surfaces

The Challenge

• Need to model the whole spectrum: from repeated to stochastic texture
Efros & Leung Algorithm

- Assuming Markov property, compute $P(p|N(p))$
  - Building explicit probability tables infeasible
  - Instead, we search the input image for all similar neighborhoods — that’s our pdf for $p$
  - To sample from this pdf, just pick one match at random

Some Details

- Growing is in “onion skin” order
  - Within each “layer”, pixels with most neighbors are synthesized first
  - If no close match can be found, the pixel is not synthesized until the end

- Using Gaussian-weighted SSD is very important
  - to make sure the new pixel agrees with its closest neighbors
  - Approximates reduction to a smaller neighborhood window if data is too sparse
Neighborhood Window

Varying Window Size

Increasing window size
Homage to Shannon

"Dick Gephart was furious riff on the learning only asked. "What went on there?" A heartfelt sigh story about the emergence against Clinton. Seeing people about consciousness began, patiently one that the legal system is nothing this least known.

Hole Filling
Extrapolation

Summary

• The Efros & Leung algorithm
  – Very simple
  – Surprisingly good results
  – Synthesis is easier than analysis!
  – …but very slow
Image Quilting [Efros & Freeman]

- **Observation:** neighbor pixels are highly correlated

**Idea:** unit of synthesis = block
- Exactly the same but now we want $P(B|N(B))$
- Much faster: synthesize all pixels in a block at once
- Not the same as multi-scale!
Minimal error boundary

overlapping blocks

vertical boundary

overlap error

min. error boundary

Our Philosophy

• The “Corrupt Professor’s Algorithm”:  
  – Plagiarize as much of the source image as you can  
  – Then try to cover up the evidence  

• Rationale:  
  – Texture blocks are by definition correct samples of texture so problem only connecting them together
Failures
(Chernobyl Harvest)

input image

Portilla & Simoncelli
Xu, Guo & Shum

Wei & Levoy
Our algorithm
Political Texture Synthesis!

Application: Texture Transfer

- Try to explain one object with bits and pieces of another object:

Same as texture synthesis, except an additional constraint:
1. Consistency of texture
2. Similarity to the image being “explained”
Image Analogies

Aaron Hertzmann\textsuperscript{1,2}
Chuck Jacobs\textsuperscript{2}
Nuria Oliver\textsuperscript{2}
Brian Curless\textsuperscript{3}
David Salesin\textsuperscript{2,3}

\textsuperscript{1}New York University
\textsuperscript{2}Microsoft Research
\textsuperscript{3}University of Washington
Image Analogies

• Solution:

Image Analogies Implementation

unfiltered target image unfiltered training image

B

A
Image Analogies Implementation

unfiltered target image

unfiltered training image

filtered training image

Image Analogies Implementation

unfiltered target image

unfiltered training image

filtered target image

filtered training image
Training

Unfiltered source (A)  Filtered source (A')

B  B'

Unfiltered source (B)  Filtered source (B')
Learn to Blur

Unfiltered source ($A$)
Filtered source ($A'$)

Unfiltered target ($B$)
Filtered target ($B'$)
Texture by Numbers

Colorization