Texture Oriented Image Inpainting Based on Local Statistical Model

Jiří Grim, Petr Somol

Pavel Pudil, Irena Míková and Miroslav Malec

Institute of Information Theory and Automation of the Academy of Sciences of the Czech Republic

and Faculty of Management, Prague University of Economics

IASTED SIP, Hawaii, August 2008

Outline

Synthesis / Reconstruction / Repair of Image Contents

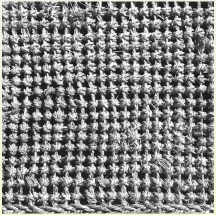
Goal: synthesise missing image data that resemble reality well:

- available training data is analyzed and a model is devised
- missing data is (re)constructed from the most "alike" part of the original image
- many approaches known,
  but only a few can deal with textures well
- local statistical model can be used to capture and replicate texture properties

Texture Modeling by Local Statistical Models

RGB image: $\mathcal{Z} = [\mathbf{z}_{ij}]_{i=0}^{I}\,_{j=0}^{J}, \quad \mathbf{z}_{ij} = (z_{ij1}, z_{ij2}, z_{ij3}) \in \mathcal{R}^3$

Examples of textures and textured images:



Statistical "homogeneity" assumption:

Texture can be described locally based on statistical properties of pixels inside a properly sized shifting window.

$$\mathbf{x}(i,j) = \mathbf{x} = (x_1, x_2, \ldots, x_N) \in \mathcal{X}, \quad \mathcal{X} = R^N$$

$\approx$ pixels inside the shifting window (usually $N = 20 \div 900$)

Remark: vectors $\mathbf{x} \in \mathcal{S}$ are not independent due to window overlaps stemming from shifting.

Texture Modeling by Normal Mixtures

Principle of Modeling (Grim et al. 2001, Haindl et al., 2004):

- estimation of local statistical properties of the texture within a shifting window using normal product mixtures $P(\mathbf{x})$
- pixelwise prediction (synthesis) of texture (of arbitrary size) based on conditional densities derived from $P(\mathbf{x})$ based on known window part
- (optional) replacement of the predicted pixel in each step by the "most similar" piece of original image

Subvector of <u>known</u> pixel variables in window:

$$\mathbf{x}_{\mathcal{C}} = (x_{n_1}, x_{n_2}, \ldots, x_{n_l}) \in \mathcal{X}_{\mathcal{C}} = \mathcal{R}^{|\mathcal{C}|}$$

$$\mathcal{C} = \{n_1, n_2, \ldots, n_l\} \subset \mathcal{N}$$

Local Statistical Model: Normal Product Mixture

Data: $\mathcal{S} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(K)}\}, \quad \mathbf{x}^{(k)} \in \mathcal{X}, \quad (K = |\mathcal{S}|)$
$\approx$ data obtained by shifting a window over image

Approximate by Gaussian mixture:

$$P(\mathbf{x}) = \sum_{m \in \mathcal{M}} w_m F(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\sigma}_m), \quad \mathbf{x} \in \mathcal{X} \tag{1}$$

$$F(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\sigma}_m) = \prod_{n \in \mathcal{N}} f_n(x_n|\mu_{mn}, \sigma_{mn}) \tag{2}$$

$$f_n(x_n|\mu_{mn}, \sigma_{mn}) = \frac{1}{\sqrt{2\pi}\sigma_{mn}} \exp\left\{-\frac{(x_n - \mu_{mn})^2}{2\sigma_{mn}^2}\right\}. \tag{3}$$

Component indexes: $\mathcal{M} = \{1, 2, \ldots, M\}$
Variable indexes: $\mathcal{N} = \{1, 2, \ldots, N\}$

Local Statistical Model: Normal Product Mixture

Mixture parameters optimized to maximize log-likelihood:

$$L = \frac{1}{|\mathcal{S}|} \sum_{x \in \mathcal{S}} \log P(\mathbf{x}) = \frac{1}{|\mathcal{S}|} \sum_{x \in \mathcal{S}} \log \left[\sum_{m \in \mathcal{M}} w_m F(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\sigma}_m) \right] \tag{4}$$

using the well-known EM algorithm:

E-step: $(m \in \mathcal{M}, n \in \mathcal{N}, \mathbf{x} \in \mathcal{S})$

$$q(m|\mathbf{x}) = \frac{w_m F(\mathbf{x}|\boldsymbol{\mu}_m, \boldsymbol{\sigma}_m)}{\sum_{j \in \mathcal{M}} w_j F(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j)}, \quad m \in \mathcal{M} \tag{5}$$

M-step:

$$w_m^{'} = \frac{1}{|\mathcal{S}|} \sum_{x \in \mathcal{S}} q(m|\mathbf{x}), \tag{6}$$

$$\mu_{mn}^{'} = \frac{1}{\sum_{x \in \mathcal{S}} q(m|\mathbf{x})} \sum_{x \in \mathcal{S}} x_n q(m|\mathbf{x}), \tag{7}$$

$$(\sigma_{mn}^{'})^2 = -(\mu_{mn}^{'})^2 + \frac{1}{\sum_{x \in \mathcal{S}} q(m|\mathbf{x})} \sum_{x \in \mathcal{S}} x_n^2 q(m|\mathbf{x}). \tag{8}$$

Local Texture Prediction by Normal Mixture

Estimate the remaining missing pixel values $x_n, n \in (\mathcal{N} \setminus \mathcal{C})$ using:

$$p_{n|\mathcal{C}}(x_n|\mathbf{x}_\mathcal{C}) = \frac{P_{n,\mathcal{C}}(x_n, \mathbf{x}_\mathcal{C})}{P_\mathcal{C}(\mathbf{x}_\mathcal{C})} = \sum_{m \in \mathcal{M}} W_m(\mathbf{x}_\mathcal{C}) f_n(x_n|\mu_{mn}, \sigma_{mn}) \qquad (9)$$

where

$$W_m(\mathbf{x}_\mathcal{C}) = \frac{w_m F(\mathbf{x}_\mathcal{C}|\boldsymbol{\mu}_m, \boldsymbol{\sigma}_m)}{\sum_{j \in \mathcal{M}} w_j F(\mathbf{x}_\mathcal{C}|\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j)}. \qquad (10)$$

$$F(\mathbf{x}_\mathcal{C}|\mu_m, \sigma_m) = \prod_{n \in \mathcal{C}} f_n(x_n|\mu_{mn}, \sigma_{mn}), \quad \mathbf{x}_\mathcal{C} \in \mathcal{X}_\mathcal{C} \qquad (11)$$

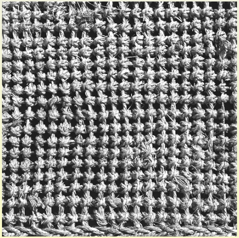
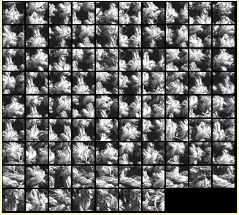
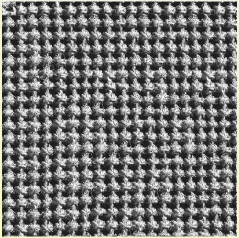
by computing the conditional expectation:

$$E\{x_n\} = \int x_n p_{n|\mathcal{C}}(x_n|\mathbf{x}_\mathcal{C}) dx_n = \sum_{m \in \mathcal{M}} W_m(\mathbf{x}_\mathcal{C}) \mu_{mn}, \qquad (12)$$

Remark: missing pixels can be substituted by component means of a single component selected randomly with respect to component weights

Example 1: Texture Modeling by Normal Mixture

"Ratan" texture synthesis – example of replacement of predicted pixels by the most similar piece of original image:

original texture	optimal patches	synthesis using patches
		

patch utilization: component means $\boldsymbol{\mu}_m$ substituted by most similar pieces (patches) $\boldsymbol{\mu}_m^*$ of the original image. Patches selected optimally so that:

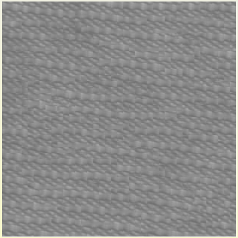
$$\boldsymbol{\mu}_m^* = \arg\min_{\mathbf{x}\in\mathcal{S}} \{\|\mathbf{x} - \boldsymbol{\mu}_m\|^2\}$$

Example 2: Texture Modeling by Normal Mixture

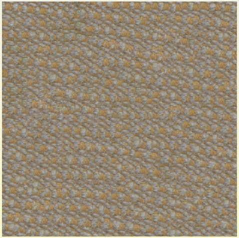
"Coarse linen" texture synthesis:

original texture

component means based synthesis



means substituted by patches



- shifting window size: 30×30 pixels
- mixture dimensionality: $N = 30 \times 30 = 900$, components: $|\mathcal{M}| = 128$
- no. of samples obtained by shifting the window: $|\mathcal{S}| \doteq 232000$
- EM algorithm iterations: $t = 15$
- window shift during synthesis: 13 pixels

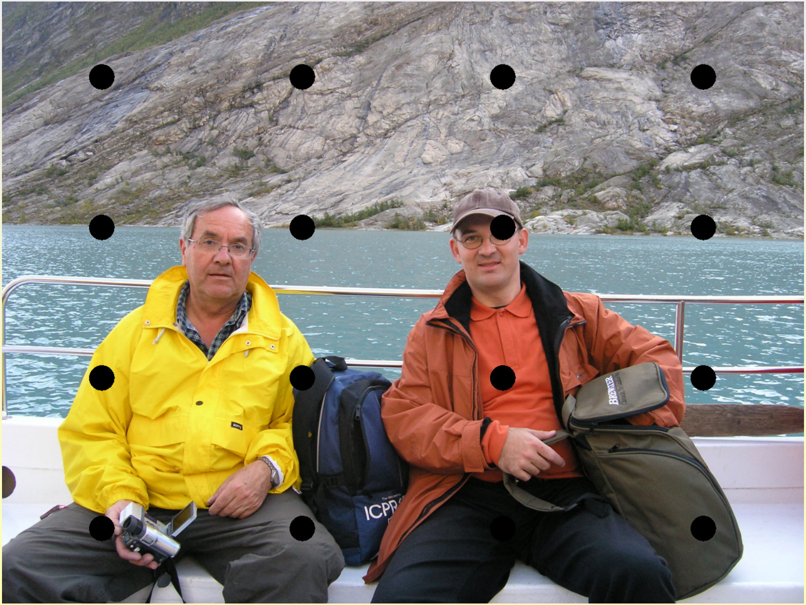
Inpainting Procedure

- location of damaged parts in input image assumed to be known
- local statistical model estimated from the available (undamaged) data, up to a specified distance from the damaged parts
- damaged parts replaced iteratively; in each iteration only those pixels with undamaged (or synthesised in last iteration) neighbors are substituted
- substitution made so that the new pixel "fits best" in the known neighbourhood

Parameters:

- window size
- window shift distance
- no. of components
- pixel substitution mechanism – component means vs. patches

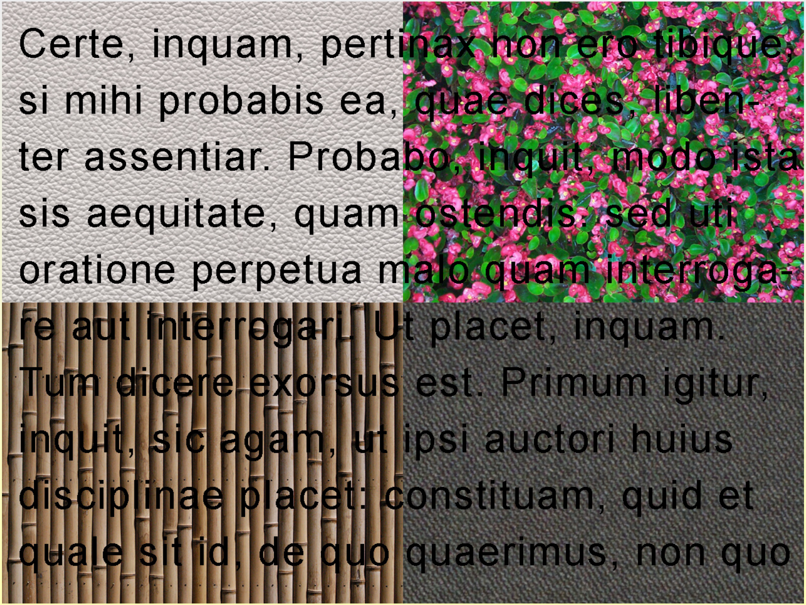
Local Image Prediction – Sample Image 1: Damaged



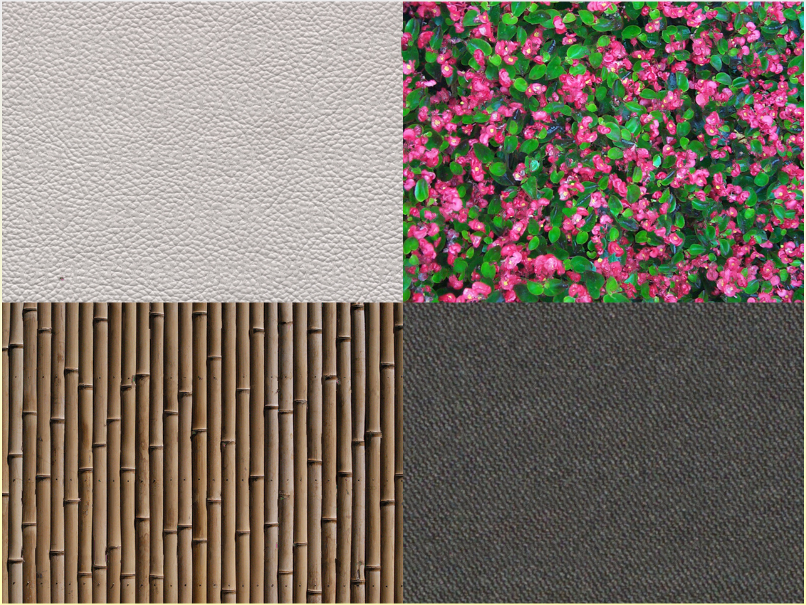
Local Image Prediction – Sample Image 1: Inpainted

Local Image Prediction – Sample Image 1: Components

Local Image Prediction – Sample Image 2: Damaged

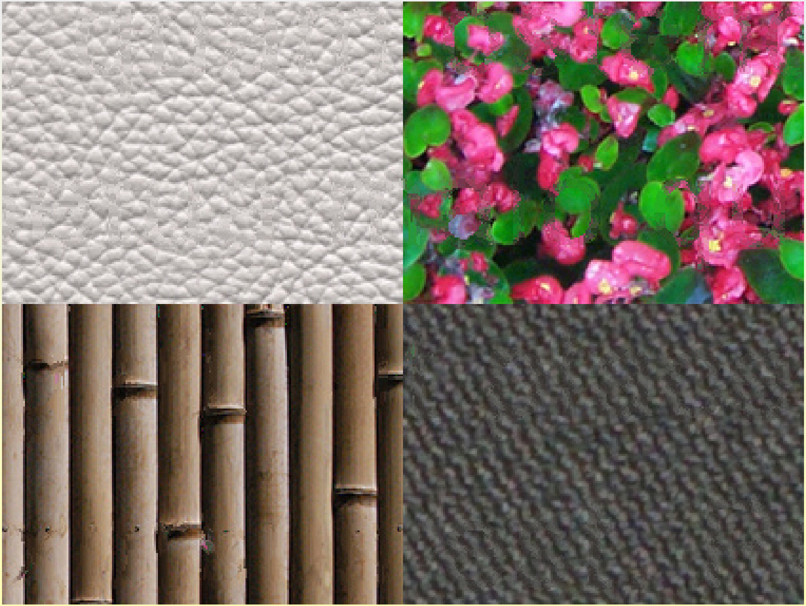


Local Image Prediction – Sample Image 2: Inpainted



Local Image Prediction – ZOOMED Image 2: Damaged

Local Image Prediction – ZOOMED Image 2: Inpainted



Local Image Prediction – Sample Image 2: Components

Local Image Prediction – Sample Image 3: Damaged

Local Image Prediction – Sample Image 3: Inpainted

Local Image Prediction – Sample Image 3: Components

References

- M. Bertalmio, G. Sapiro, V. Caselles & C. Ballester, Image inpainting, *Computer Graphics (SIGGRAPH)*, 2000, 417–424
- M. Bertalmio, L. Vese, G. Sapiro & S. Osher, Simultaneous Structure and Texture Image Inpainting, *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR'03)*, Vol. 2, IEEE Computer Society, 2003
- J. Grim & M. Haindl, Texture Modelling by Discrete Distribution Mixtures, *Computational Statistics and Data Analysis*, 41(3-4), 2003, 603–615
- M. Haindl, J. Grim, P. Somol, P. Pudil & M. Kudo, A Gaussian mixture-based color texture model, *Proc. 17th IAPR Int. Conf. on Pattern Recognition*, IEEE, Los Alamitos, 2004, 177–180
- H. Igehy & L. Pereira, Image replacement through texture synthesis, *Proc. of the Int. Conf. on Image Processing (ICIP '97)*, Vol. 3, IEEE, Washington, 1997