Markov Random Fields & Texture Analysis

Vincent Cheung
Probabilistic and Statistical Inference Group
Electrical & Computer Engineering
University of Toronto

Jan. 21, 2005

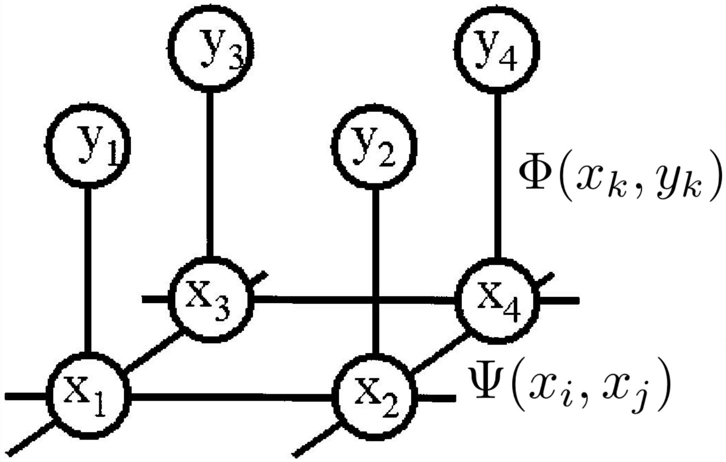
Outline

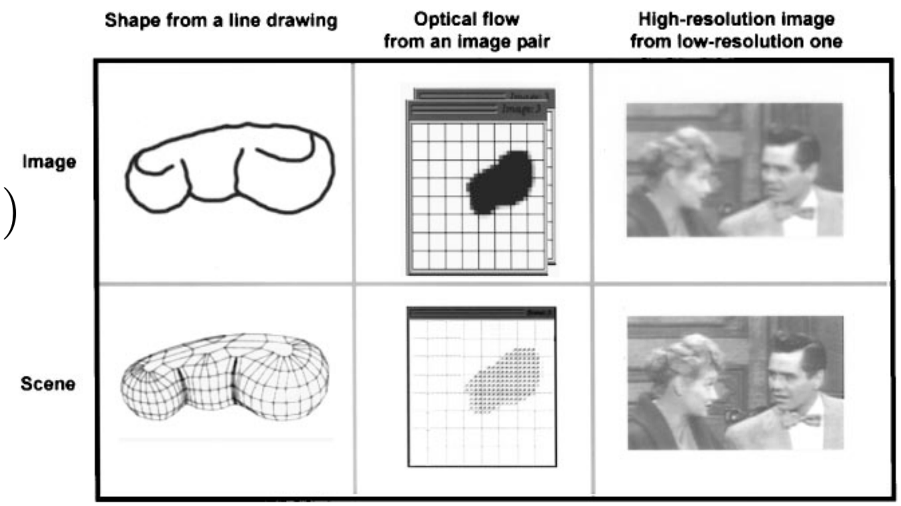
- W. Freeman, E. Pasztor, and O. Carmichael, "*Learning Low-Level Vision*"
  - Markov random fields
  - Belief propagation
- S. Zhu, Y. Wu, and D. Mumford, "*Minimax Entropy Principle and its Application to Texture Modeling*"
  - Minimax entropy principle
  - Feature pursuit

Markov Random Field

- An undirected graph
  - Nodes <=> Variables
  - Edges <=> Functions
- Model the joint probability
- Perform inference

Markov Network for Vision

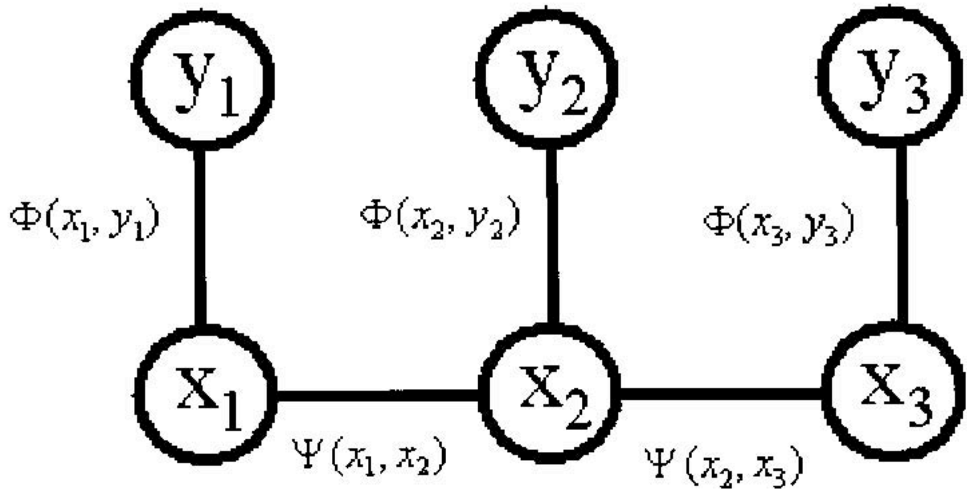




$$P(x_1, x_2, ..., x_N, y_1, y_2, ..., y_N) = \prod_{(i,j)} \Psi(x_i, x_j) \prod_k \Phi(x_k, y_k)$$

$$\hat{x}_j = \underset{x_j}{arg\,max} \; \underset{x_i \neq x_j}{max} \; P(x_1, x_2, ..., x_N, y_1, y_2, ..., y_N)$$

A Simple Markov Network



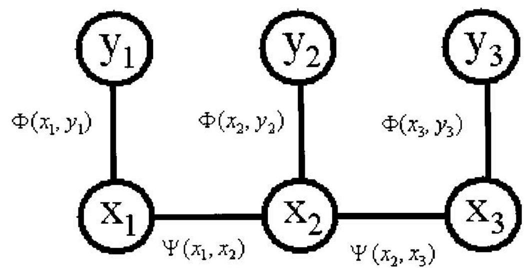
$$P(x_1, x_2, x_3, y_1, y_2, y_3) = \Phi(x_1, y_1)\Phi(x_2, y_2)\Phi(x_3, y_3)\Psi(x_1, x_2)\Psi(x_2, x_3)$$

$$\begin{aligned}
\hat{x}_1 &= \underset{x_1}{arg\ max}\ \underset{x_2}{max}\ \underset{x_3}{max}\ P(x_1, x_2, x_3, y_1, y_2, y_3) \\
&= \underset{x_1}{arg\ max}\ \underset{x_2}{max}\ \underset{x_3}{max}\ \Phi(x_1, y_1)\Phi(x_2, y_2)\Phi(x_3, y_3)\Psi(x_1, x_2)\Psi(x_2, x_3) \\
&= \underset{x_1}{arg\ max}\ \Phi(x_1, y_1)\ \underset{x_2}{max}\ \Psi(x_1, x_2)\Phi(x_2, y_2)\ \underset{x_3}{max}\ \Psi(x_2, x_3)\Phi(x_3, y_3)
\end{aligned}$$

Belief Propagation

- MAP estimates can be computed for all the x_i simultaneously using “message-passing”
- Factorization structure makes this possible

Belief Propagation Example



$$\hat{x}_1 = \underset{x_1}{arg\ max}\, \Phi(x_1, y_1) * \underset{x_2}{max}\, \Psi(x_1, x_2)\Phi(x_2, y_2) * \underset{x_3}{max}\, \Psi(x_2, x_3)\Phi(x_3, y_3)$$

$$\tilde{M}_2^1 = \underset{x_1}{max}\, \Psi(x_2, x_1)\Phi(x_1, y_1) \qquad M_2^1 = \underset{x_1}{max}\, \Psi(x_2, x_1)\Phi(x_1, y_1)$$

$$\tilde{M}_1^2 = \underset{x_2}{max}\, \Psi(x_1, x_2)\Phi(x_2, y_2) \qquad M_1^2 = \underset{x_2}{max}\, \Psi(x_1, x_2)\Phi(x_2, y_2)\tilde{M}_2^3$$

$$\tilde{M}_3^2 = \underset{x_2}{max}\, \Psi(x_3, x_2)\Phi(x_2, y_2) \qquad M_3^2 = \underset{x_2}{max}\, \Psi(x_3, x_2)\Phi(x_2, y_2)\tilde{M}_2^1$$

$$\tilde{M}_2^3 = \underset{x_3}{max}\, \Psi(x_2, x_3)\Phi(x_3, y_3) \qquad M_2^3 = \underset{x_3}{max}\, \Psi(x_2, x_3)\Phi(x_3, y_3)$$

$$\hat{x}_1 = \underset{x_1}{arg\ max}\, \Phi(x_1, y_1) M_1^2$$

Belief Propagation Summary

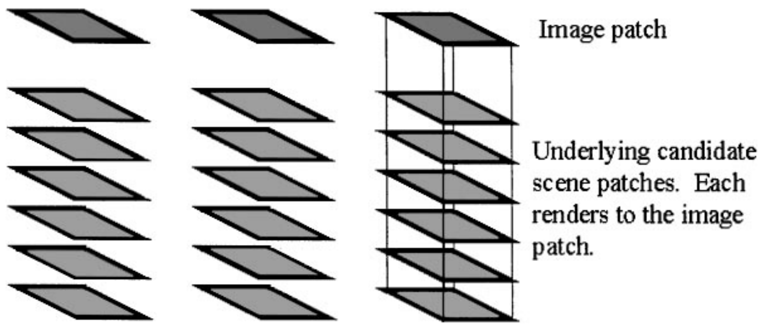
- What a node thinks another should be

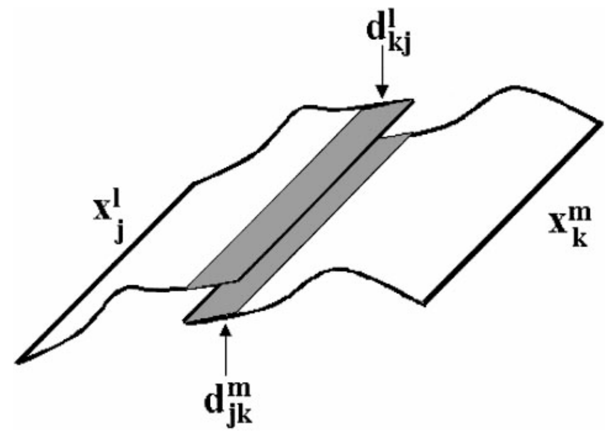
$$M_j^k = \underset{[x_k]}{max} \Psi(x_j, x_k)\Phi(x_k, y_k)\prod_{l\neq j}\tilde{M}_k^l$$

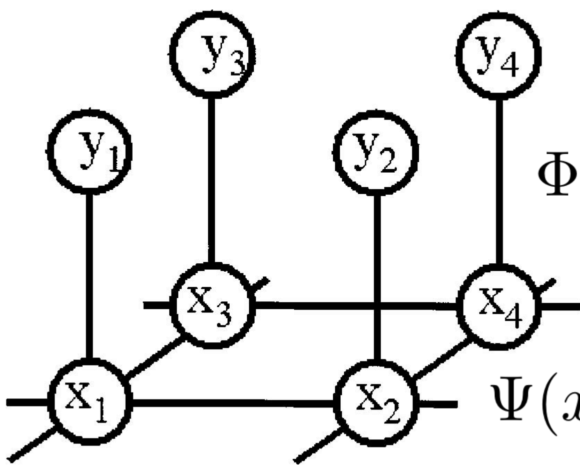
- What a node thinks itself is

$$\hat{x}_j = \underset{x_j}{arg\ max}\ \Phi(x_j, y_j)\prod_k M_j^k$$

Scene Candidate Patches





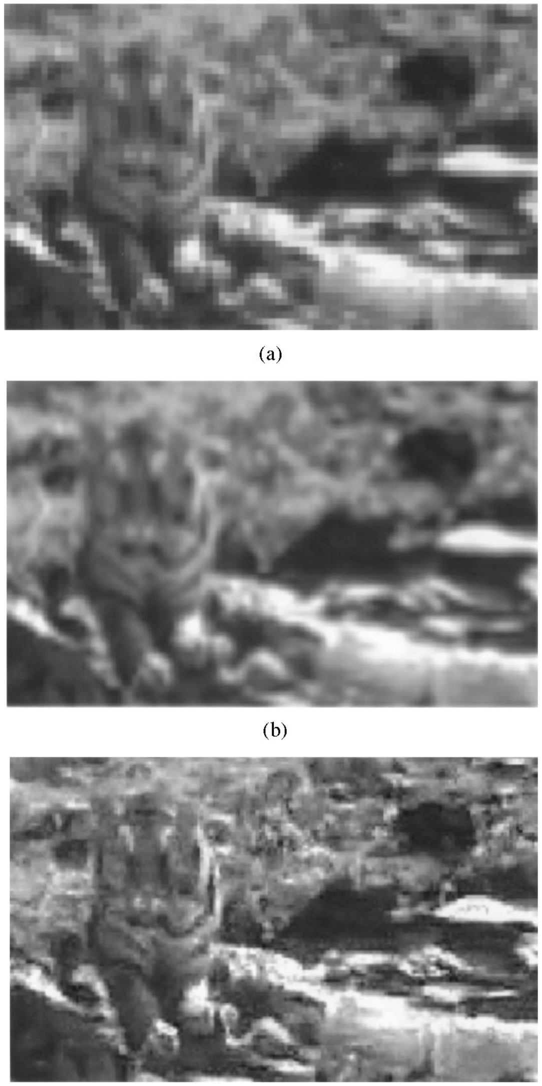


$$\Phi(x_k^l, y_k) = \exp(-\|x_k^l - y_k\|^2/2\sigma_i^2)$$

$$\Psi(x_k^l, x_j^m) = \exp(-\|d_{jk}^l - d_{kj}^m\|^2/2\sigma_s^2)$$

Applications (1)

- Super-Resolution
- Shading and reflectance
- Motion estimation



(a) (b) (c)

Applications (2)

- Transparency (A. Levin, A. Zomet and Y. Weiss)

Minimax Entropy Principle

- Let $\mathbf{I}$ be an image
- Assume that observed images, $\mathbf{I}_i$, $i=1,..,m$, are random samples from a probability distribution $f(\mathbf{I})$
- Goal is to estimate $f(\mathbf{I})$ based on the observed images

Maximum Entropy Principle (1)

- Maximize entropy to obtain the purest and simplest fusion of the observed features and their statistics
- Simplest model as possible

Maximum Entropy Principle (2)

$$\mu_{obs}^{(\alpha)} = \frac{1}{M}\sum_{i=1}^{M}\phi^{(\alpha)}(\boldsymbol{I}_i^{obs}), \quad for\ \alpha = 1, ..., K$$

$$\Omega = \{p(\boldsymbol{I}) \ : \ E_p[\phi^{(\alpha)}(\boldsymbol{I})] = \mu_{obs}^{(\alpha)}\}$$

$$p(\boldsymbol{I}) = \arg\max\{-\int p(\boldsymbol{I})\log p(\boldsymbol{I})d\boldsymbol{I}\}$$

s.t.

$$E_p[\phi^{(\alpha)}(\boldsymbol{I})] = \int \phi^{(\alpha)}(\boldsymbol{I})p(\boldsymbol{I})d\boldsymbol{I} = \mu_{obs}^{(\alpha)}$$

$$\int p(\boldsymbol{I})d\boldsymbol{I} = 1$$

$$p(\boldsymbol{I};\ \Lambda) = \frac{1}{Z(\Lambda)}\exp\{-\sum_{\alpha=1}^{K} < \lambda^{(\alpha)}, \phi^{(\alpha)}(\boldsymbol{I}) >\}$$

Minimum Entropy Principle

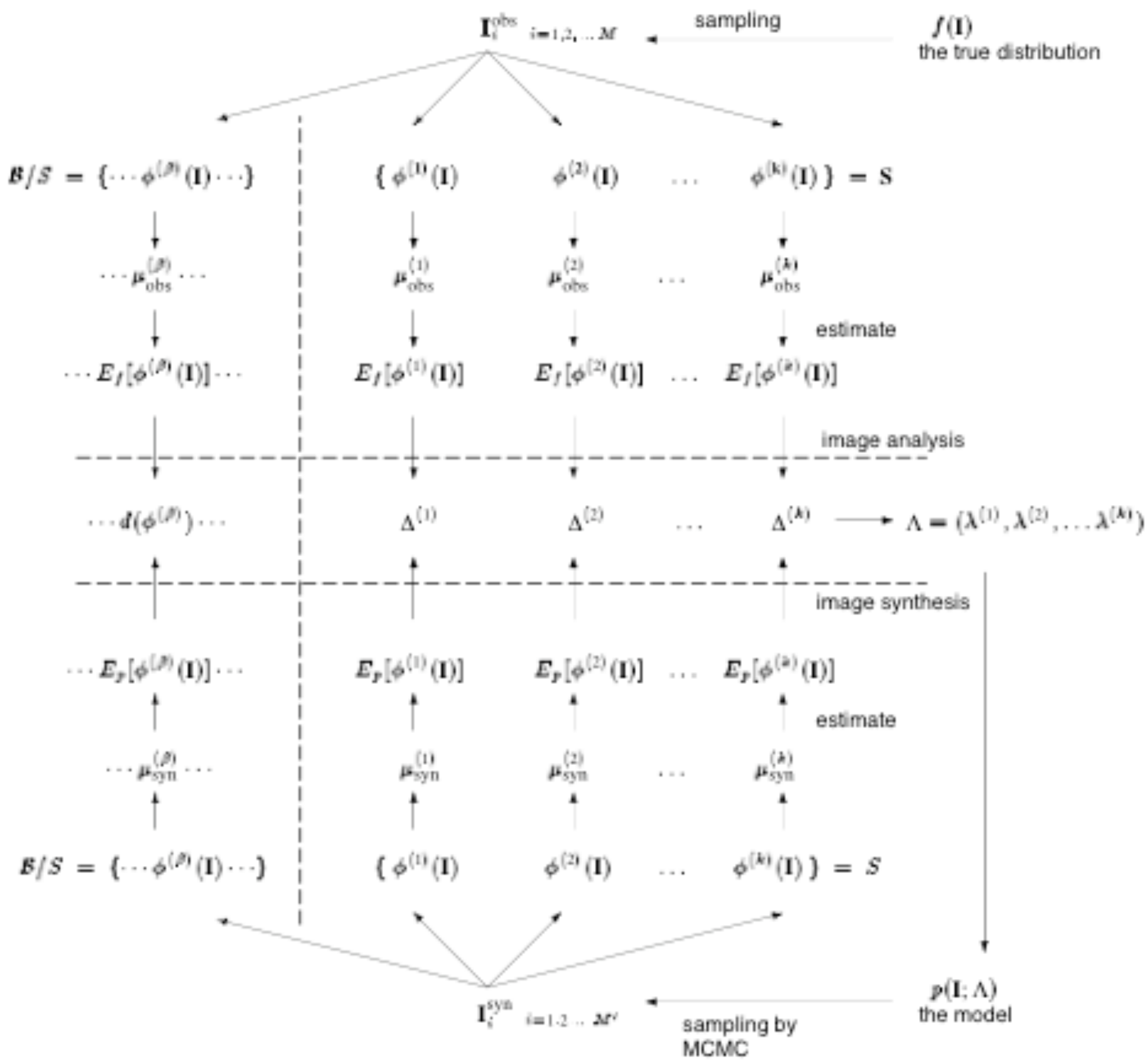
- Minimize entropy to increase model complexity by choosing a set of features

$$\begin{aligned} \min KL(f, p(\boldsymbol{I} \quad \Lambda^*)) &= \min \int f(\boldsymbol{I}) \log \frac{f(\boldsymbol{I})}{p(\boldsymbol{I};\ \Lambda^*)} d\boldsymbol{I} \\ &= \min entropy(p(\boldsymbol{I};\ \Lambda^*)) \end{aligned}$$

$$S^* = \underset{|S| = K}{arg\ min}\ entropy(p_S(\boldsymbol{I};\ \Lambda^*))$$

Feature Pursuit

- Inefficient to just try all possible set of K features
- Use greedy approach
  - Add one feature to the model at a time
  - Choose feature that maximally decreases the entropy
  - Find feature that is most poorly modeled by the current model and add it to it’s repertoire so that the model can better represent this feature.

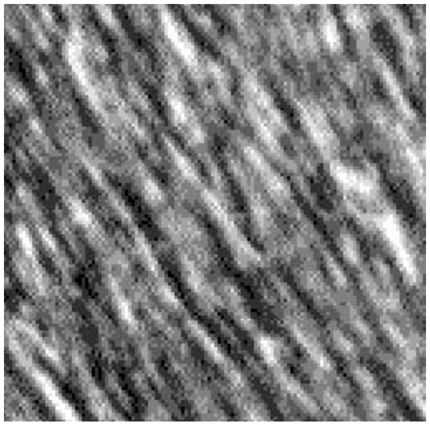


sampling
f(I)
the true distribution
estimate
image analysis
image synthesis
estimate
sampling by MCMC
p(I; Λ)
the model

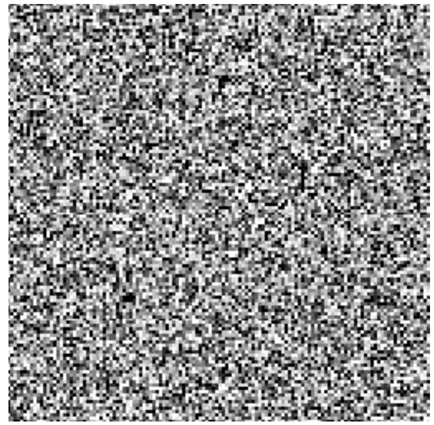
Texture Modeling

- Histogram of filters as features
- FRAME - Filter, Random field, And Minimax Entropy
- Learn the filters
- Synthesize texture from the histogram of the filter response

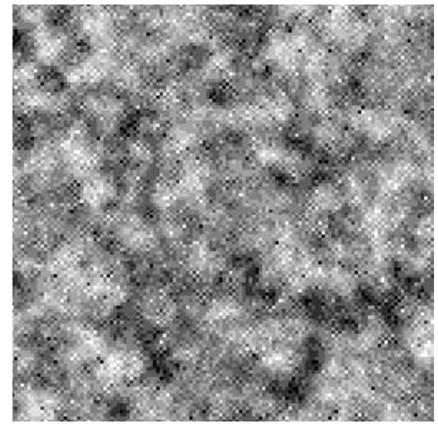
Texture Modeling Example (1)



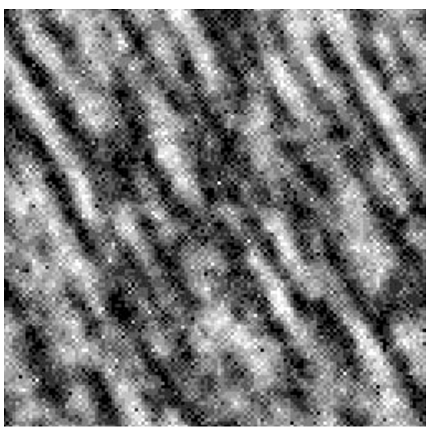
Obs



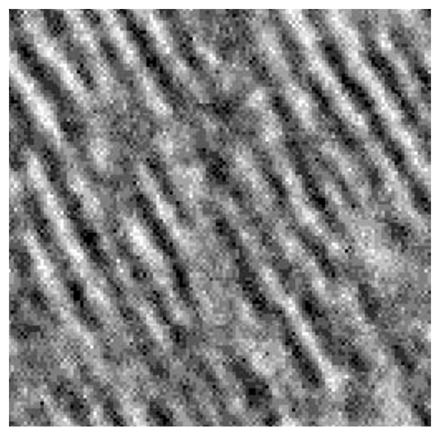
0



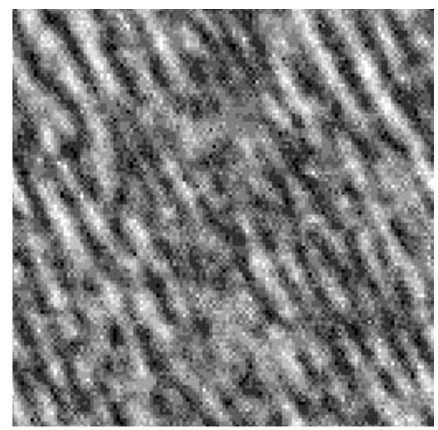
1



2



3



6

Texture Modeling Example (2)