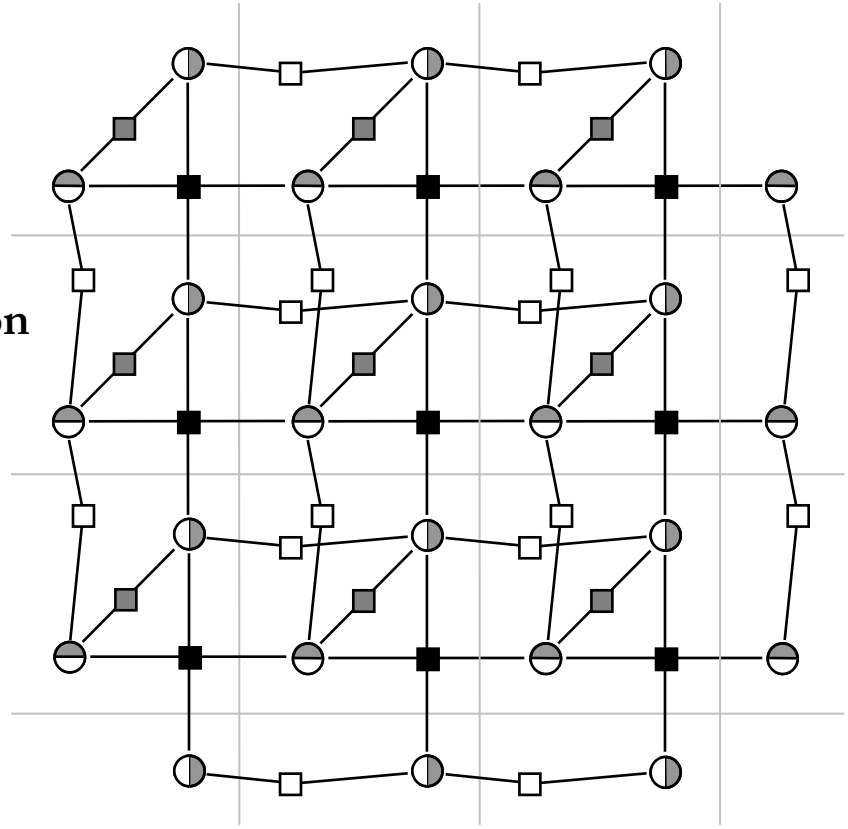
Efficient Belief Propagation for Vision Using Linear Constraint Nodes

Brian Potetz

Dept. of Computer Science &
Center for the Neural Basis of Cognition
Carnegie Mellon University

CVPR 2007



Belief Propagation

- Computes marginals of factorized distribution
  ex: $P(a,b,c,d,e,g) \propto f_1(a,b,c,d) f_2(b,e) f_3(c,g) f_4(e,g)$
- Has been used successfully in many applications
  - Error-correcting codes, Image super-resolution, stereo, photometric stereo, etc.
- Slow for large cliques
  - Each message takes $\mathcal{O}(M^N)$ time
  - Especially slow for continuous variables
  - Loopy Belief Propagation over continuous variables is historically limited to pairwise MRFs

Belief Propagation

- Computes marginals of factorized distribution

  ex: $P(a,b,c,d,e,g) \propto f_1(a,b,c,d) f_2(b,e) f_3(c,g) f_4(e,g)$

- Has been used successfully in many applications
  - Error-correcting codes, Image super-resolution, stereo, photometric stereo, etc.
- Slow for large cliques
  - Each message takes $\mathcal{O}(M^N)$ time
  - Especially slow for continuous variables
  - Loopy Belief Propagation over continuous variables is historically limited to pairwise MRFs

M = number of states per variable

N = number of variables in clique

Efficient Belief Propagation for Vision Using Linear Constraint Nodes

Today's Talk:

1) A method to reduce complexity from $\mathcal{O}(M^N)$ to $\mathcal{O}(NM^2)$ for certain potential functions.
2) Application: Higher Order Spatial Priors
3) Application: Shape from Shading

Efficient Belief Propagation for Large Cliques

Goal of Belief Propagation:

Given $P(\vec{x}) \propto f_1(x_1, x_2, x_3, x_4) f_2(x_2, x_5) f_3(x_3, x_6) f_4(x_5, x_6)$

compute marginals for each variable.

- Each factor sends a message to each associated variable. Requires $\mathcal{O}(M^N)$ operations.

  N = number of variables in the factor. M = number of states per variable.

Efficient Belief Propagation for Large Cliques

Goal of Belief Propagation:

Given $P(\vec{x}) \propto f_1(x_1, x_2, x_3, x_4) f_2(x_2, x_5) f_3(x_3, x_6) f_4(x_5, x_6)$
compute marginals for each variable.

- Each factor sends a message to each associated variable. Requires $\mathcal{O}(M^N)$ operations.

  N = number of variables in the factor. M = number of states per variable.

- Suppose $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$ (Linear Constraint Node)
  Using a change of variables, each message can be computed in $\mathcal{O}(NM^2)$ operations

  – This is an exact method.

- Method also works if $f_1(\vec{x}) = g\left(\sum_{i=1}^{N} g_i(x_i)\right)$

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)dx_2dx_3dx_4$$

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)dx_2dx_3dx_4$$

$$= \int\int\int J\, g(v_1x_1+y_1)m_2(\frac{y_1-y_2}{v_2})m_3(\frac{y_2-y_3}{v_3})m_4(\frac{y_3}{v_4})dy_1dy_2dy_3$$

(Jacobian is constant)

Apply a change of variables:

$$y_3 = v_4x_4$$
$$y_2 = v_3x_3 + y_3$$
$$y_1 = v_2x_2 + y_2$$

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)\,dx_2\,dx_3\,dx_4$$

$$= \int\int\int J\,g(v_1x_1+y_1)\,m_2(\frac{y_1-y_2}{v_2})\,m_3(\frac{y_2-y_3}{v_3})\,m_4(\frac{y_3}{v_4})\,dy_1\,dy_2\,dy_3$$

$$\propto \int g(v_1x_1+y_1)\Big(\int m_2(\frac{y_1-y_2}{v_2})\,\underbrace{\Big(\int m_3(\frac{y_2-y_3}{v_3})m_4(\frac{y_3}{v_4})dy_3\Big)}\,dy_2\Big)\,dy_1$$

Perform each integrand seperately

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)\,dx_2\,dx_3\,dx_4$$

$$= \int\int\int J\,g(v_1x_1+y_1)\,m_2(\frac{y_1-y_2}{v_2})\,m_3(\frac{y_2-y_3}{v_3})\,m_4(\frac{y_3}{v_4})\,dy_1\,dy_2\,dy_3$$

$$\propto \int g(v_1x_1+y_1)\Big(\int m_2(\frac{y_1-y_2}{v_2}) \qquad S_1(y_2) \qquad dy_2\Big)\,dy_1$$

Perform each integrand seperately

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)\,dx_2\,dx_3\,dx_4$$

$$= \int\int\int J\,g(v_1x_1+y_1)m_2(\frac{y_1-y_2}{v_2})m_3(\frac{y_2-y_3}{v_3})m_4(\frac{y_3}{v_4})\,dy_1\,dy_2\,dy_3$$

$$\propto \int g(v_1x_1+y_1)\underbrace{\Big(\int m_2(\frac{y_1-y_2}{v_2})\;S_1(y_2)\;\;dy_2\Big)}\;dy_1$$

Perform each integrand seperately

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)\,dx_2\,dx_3\,dx_4$$

$$= \int\int\int J\,g(v_1x_1+y_1)\,m_2(\frac{y_1-y_2}{v_2})\,m_3(\frac{y_2-y_3}{v_3})\,m_4(\frac{y_3}{v_4})\,dy_1\,dy_2\,dy_3$$

$$\propto \int g(v_1x_1+y_1) \qquad S_2(y_1) \qquad dy_1$$

Perform each integrand seperately

Linear Constraint Nodes

If the factor $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v})$, then I can compute its messages in $\mathcal{O}(NM^2)$ time (instead of $\mathcal{O}(M^N)$ time)

$m_{f \to i}^{t+1}(x_i) =$ (Message from Factor node to Variable node)

$$\int\int\int g(v_1x_1+v_2x_2+v_3x_3+v_4x_4)m_2(x_2)m_3(x_3)m_4(x_4)\,dx_2\,dx_3\,dx_4$$

$$= \int\int\int J\,g(v_1x_1+y_1)\,m_2(\frac{y_1-y_2}{v_2})\,m_3(\frac{y_2-y_3}{v_3})\,m_4(\frac{y_3}{v_4})\,dy_1\,dy_2\,dy_3$$

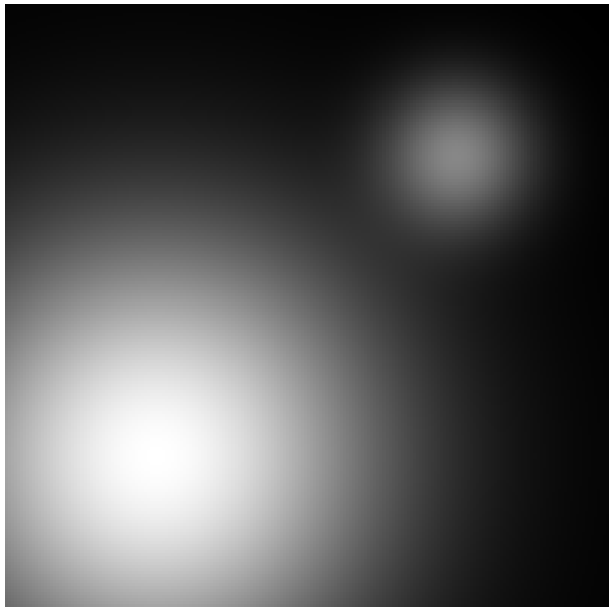
$$\propto \underbrace{\int g(v_1x_1+y_1)\,S_2(y_1)\,dy_1}$$

Perform each integrand seperately

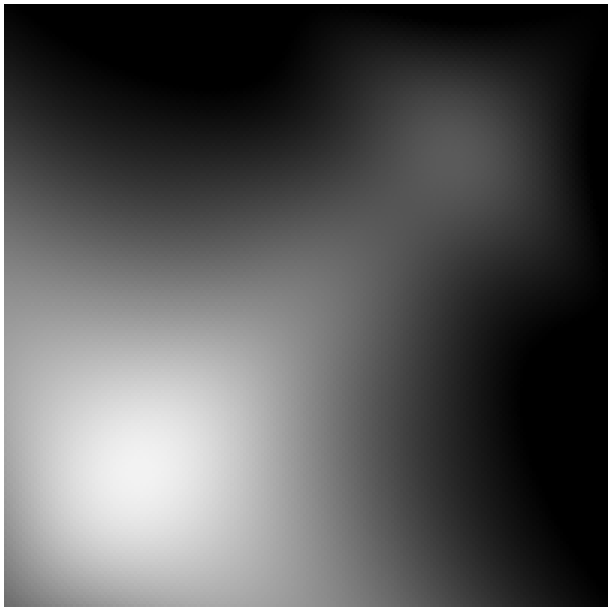
Linear Constraint Nodes

A sufficient number of linear constraint nodes can approximate any potential function

$$f(\vec{x}) = \prod_{i=1}^{2} g_i(\vec{x} \cdot \vec{v_i})$$



Original Potential Function

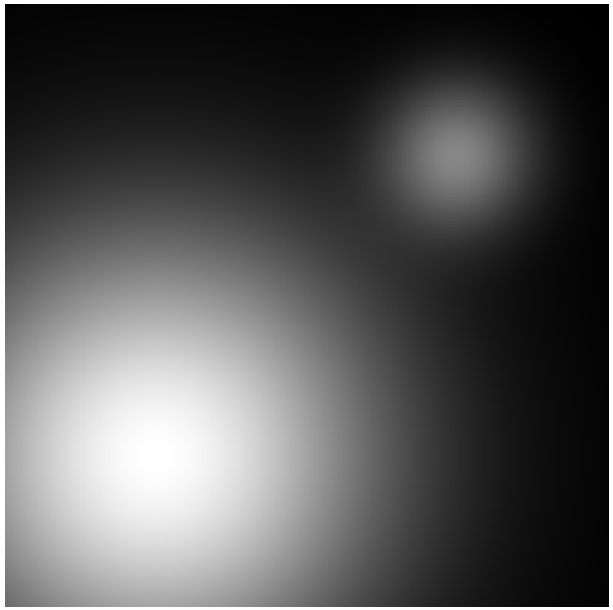


Approximation using 2 constraint nodes

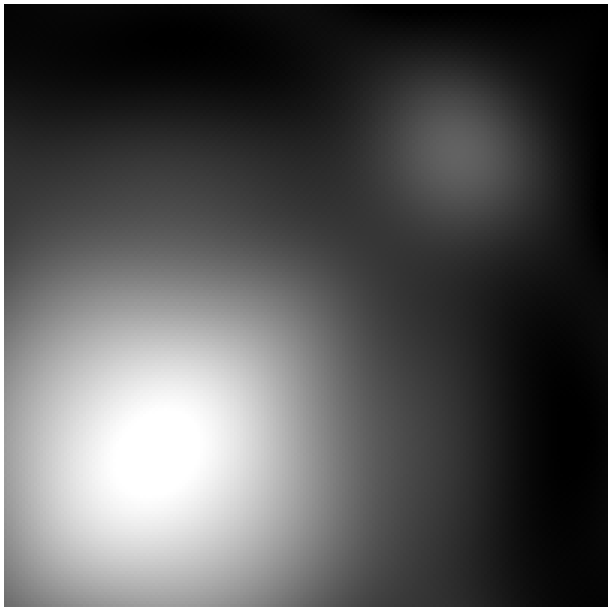
Linear Constraint Nodes

A sufficient number of linear constraint nodes can approximate any potential function

$$f(\vec{x}) = \prod_{i=1}^{4} g_i(\vec{x} \cdot \vec{v_i})$$



Original Potential Function

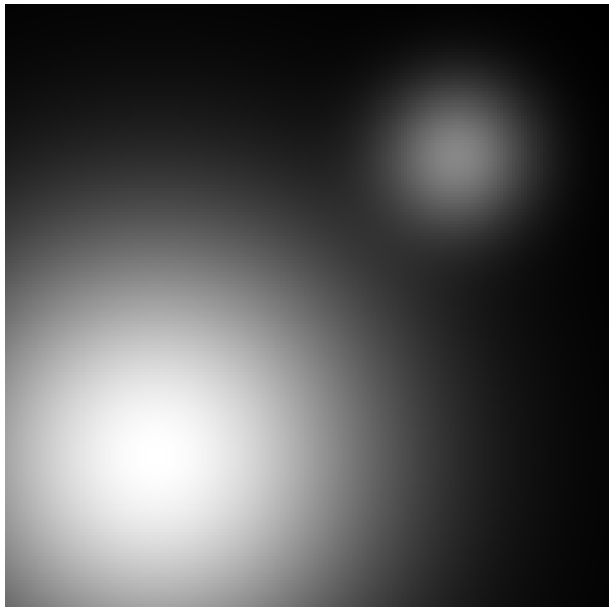


Approximation using 4 constraint nodes

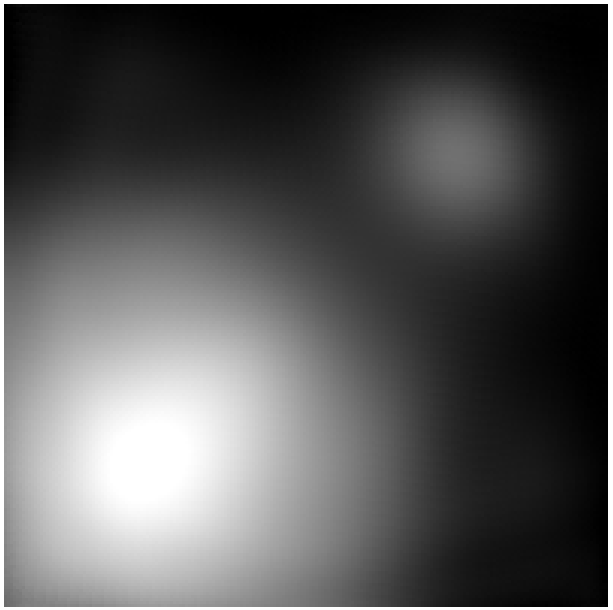
Linear Constraint Nodes

A sufficient number of linear constraint nodes can approximate any potential function

$$f(\vec{x}) = \prod_{i=1}^{6} g_i(\vec{x} \cdot \vec{v_i})$$



Original Potential Function

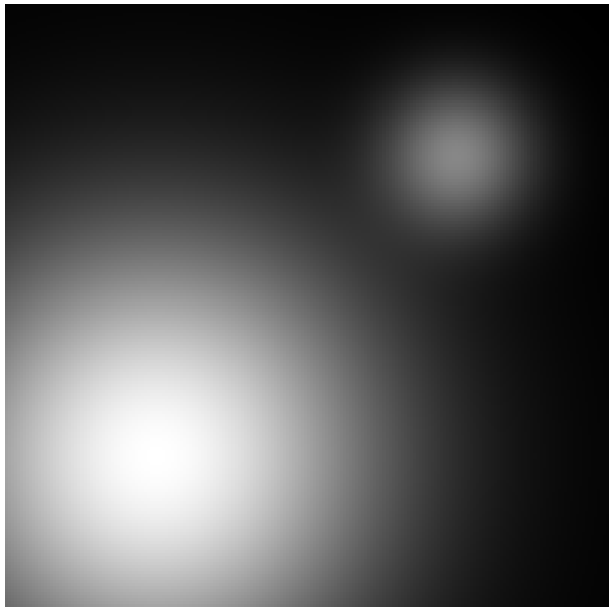


Approximation using 6 constraint nodes

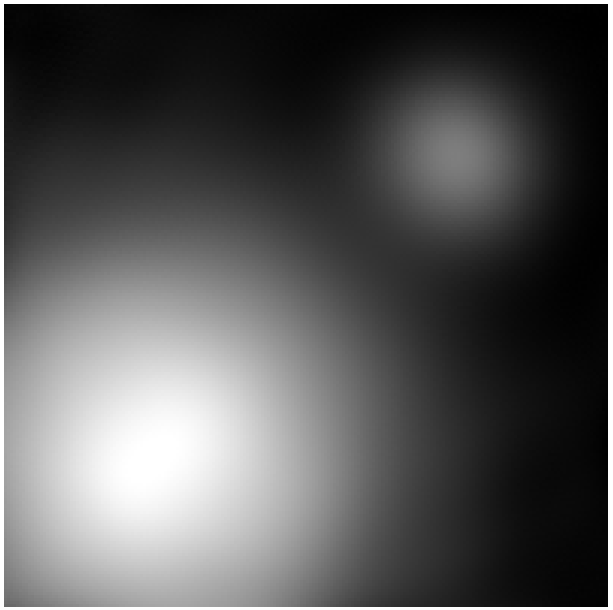
Linear Constraint Nodes

A sufficient number of linear constraint nodes can approximate any potential function

$$f(\vec{x}) = \prod_{i=1}^{8} g_i(\vec{x} \cdot \vec{v_i})$$



Original Potential Function

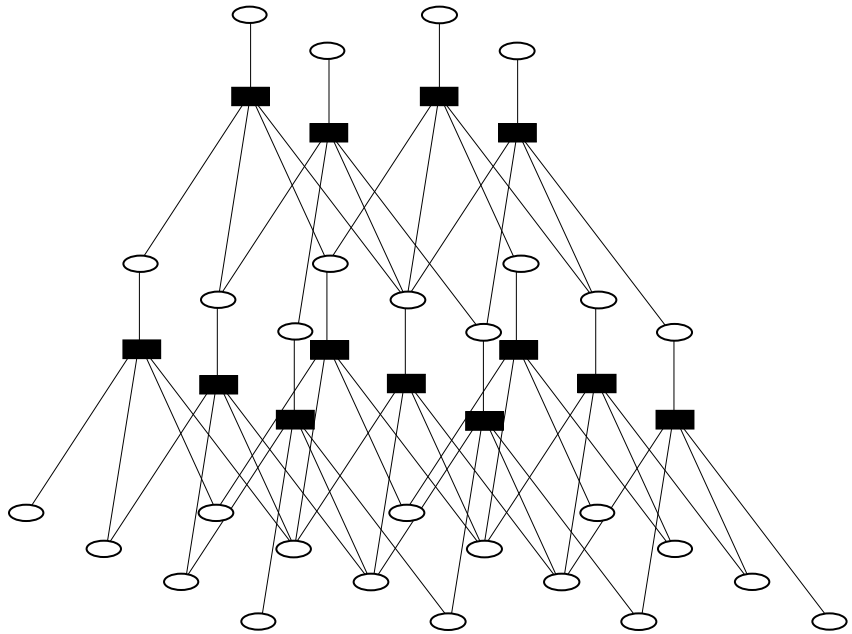


Approximation using 8 constraint nodes

Hard Linear Constraint Nodes

Hard constraints: $f_1(\vec{x}) = g(\vec{x} \cdot \vec{v}) = \begin{cases} 1 & \text{if } \vec{x} \cdot \vec{v} = c \\ 0 & \text{otherwise} \end{cases}$

- Hard constraints allow overcomplete representations.
- Good choice of representation makes LBP more efficient and more effective.



Application 1: Inference with Higher-Order Spatial Priors

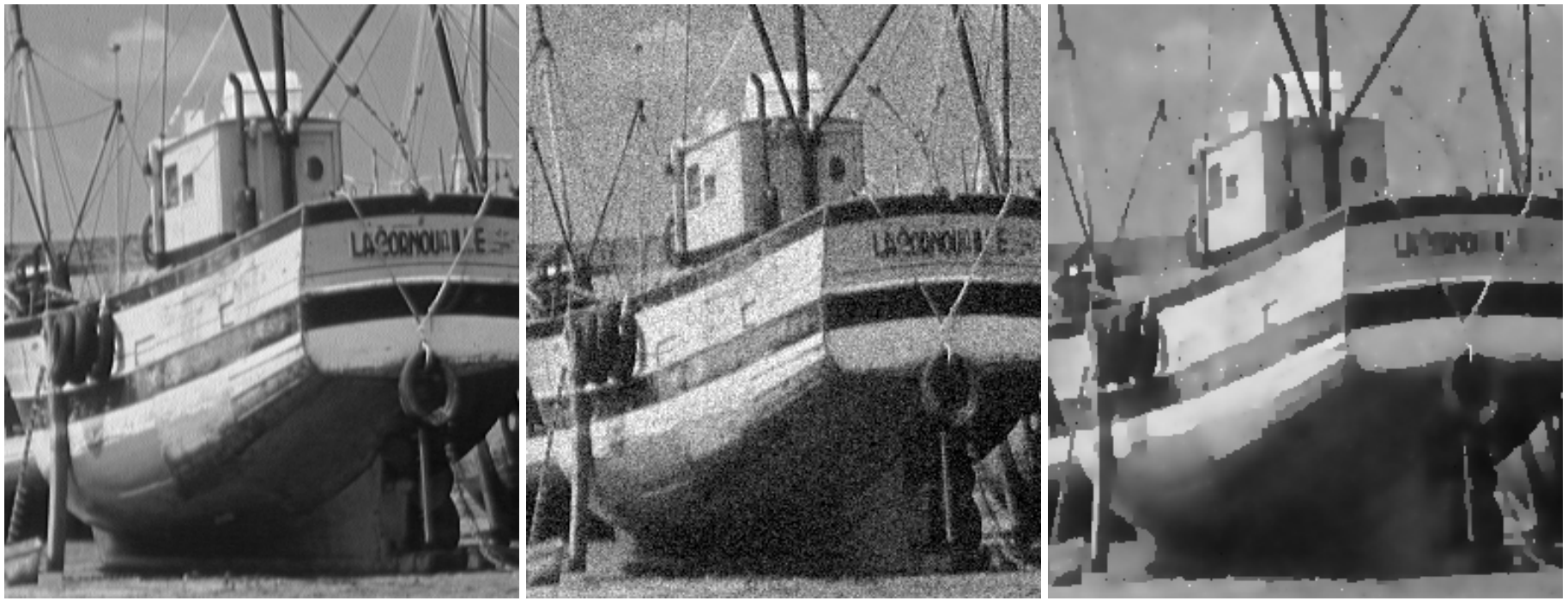
- Many applications of belief propagation infer images or intrinsic images
  - image-based rendering, denoising, inpainting
  - stereo, shape-from-X, reflectance estimation
- These problems are highly ambiguous & require strong spatial priors for success
- Pairwise MRFs cannot capture rich image statistics.

Pairwise MRFs for Denoising

Original Image

Image with Gaussian additive noise ($\sigma = 20$)

Pairwise MRF



P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. In *CVPR*, volume 1, pages 261–268, 2004.

Fields of Experts

- Learn the prior probability of images as a product of student-T distributions on the outputs of *K* linear filters:

$$p(\vec{I}) \propto \prod_{C} \prod_{i=1}^{K} \left(1 + \frac{1}{2}(\vec{I}_C \cdot \vec{J}_i)^2\right)^{-\alpha_i}$$

S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *CVPR*, pages 860–867, 2005.

Fields of Experts

- Learn the prior probability of images as a product of student-T distributions on the outputs of *K* linear filters:

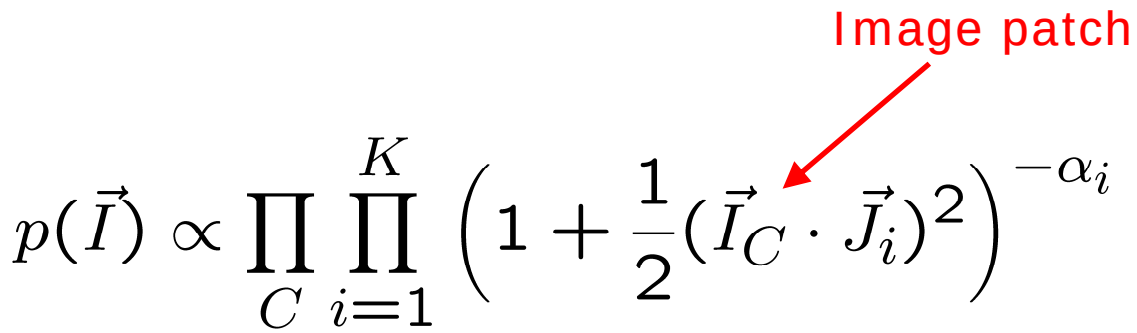
Image patch

$$p(\vec{I}) \propto \prod_{C} \prod_{i=1}^{K} \left(1 + \frac{1}{2}(\vec{I}_C \cdot \vec{J}_i)^2\right)^{-\alpha_i}$$

S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *CVPR*, pages 860–867, 2005.

Fields of Experts

- Learn the prior probability of images as a product of student-T distributions on the outputs of *K* linear filters:



S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *CVPR*, pages 860–867, 2005.

Fields of Experts

- Denoising with FoE is competitive with the current state-of-the-art using 5x5 filters & gradient descent.
- FoE could make a great spatial prior for other applications (stereo, SFS, etc).
- Is efficient belief propagation possible for FoE?

Fields of Experts

- Denoising with FoE is competitive with the current state-of-the-art using 5x5 filters & gradient descent.
- FoE could make a great spatial prior for other applications (stereo, SFS, etc).
- Is efficient belief propagation possible for FoE?
- Lan, Roth, Huttenlocher, & Black, ECCV 06
  - Uses three 2x2 “experts” (clique size = 4)
  - Uses computational short-cuts particular to denoising.
- Using Linear Contraint Nodes reduces computational complexity from $\mathcal{O}(M^4)$ to $\mathcal{O}(4M^2)$

X. Lan, S. Roth, D. P. Huttenlocher, and M. J. Black. Efficient belief propagation with learned higher-order markov random fields. In *ECCV*, pages 269–282, 2006

Original Image

Noisy Image (σ = 20)

Pairwise MRF

2x2 Linear Constraint Nodes

Denoising Results

1) Performance is slightly better than Lan et al:

PSNR:	σ =10	σ =20
Pairwise	30.73	26.66
Lan et al	30.89	27.29
Ours	31.62	27.40

2) Speed is much faster:

Lan et al: 16min / iteration

Ours: <2min / iteration

Both require ~30 iterations.

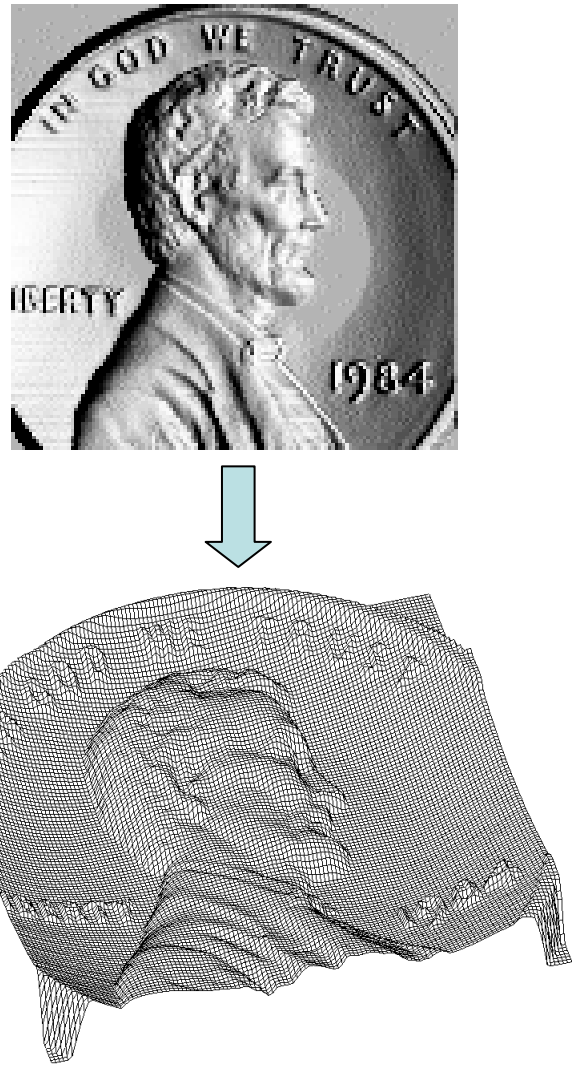
3) Design generalizes to other applications.

Application 2: Shape from Shading

- Goal: Recover 3D surface shape from a single image
  - Assume a single, infinitely-distant light source from a known direction.
  - Assume a Lambertian surface with constant albedo.

$$I = \max(0, L \cdot N)$$

$$= \max(0, \frac{\frac{\partial z}{\partial x}L_x + \frac{\partial z}{\partial y}L_y + L_z}{\sqrt{1 + \frac{\partial z}{\partial x}^2 + \frac{\partial z}{\partial y}^2}})$$

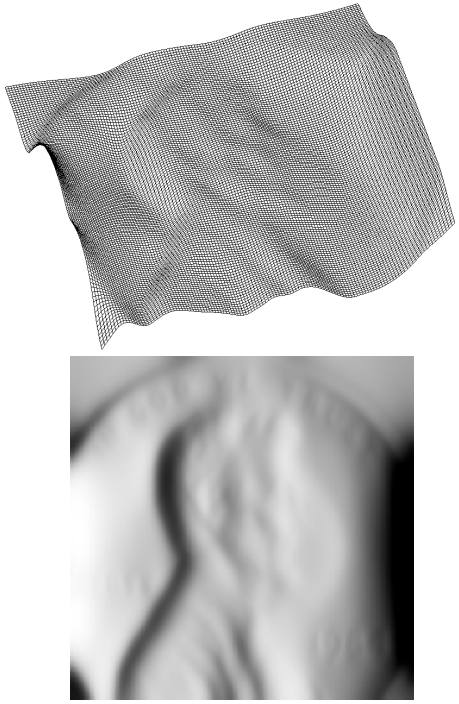


Shape from Shading

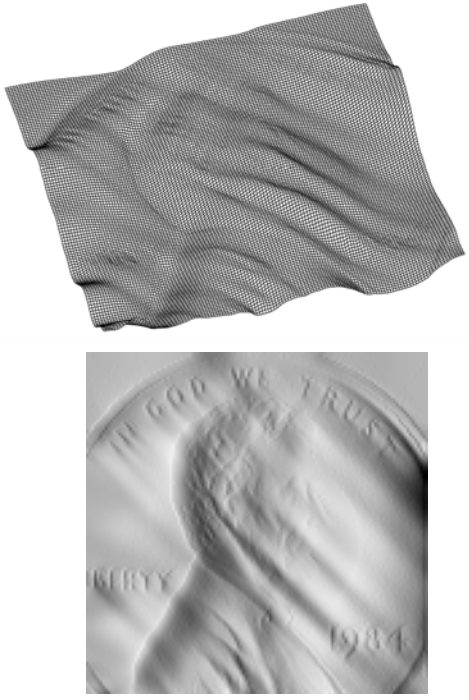
- SFS is highly *nonlinear* and *underconstrained*
- Previous approaches often struggle with local minima.

Ground Truth

Lee & Kuo

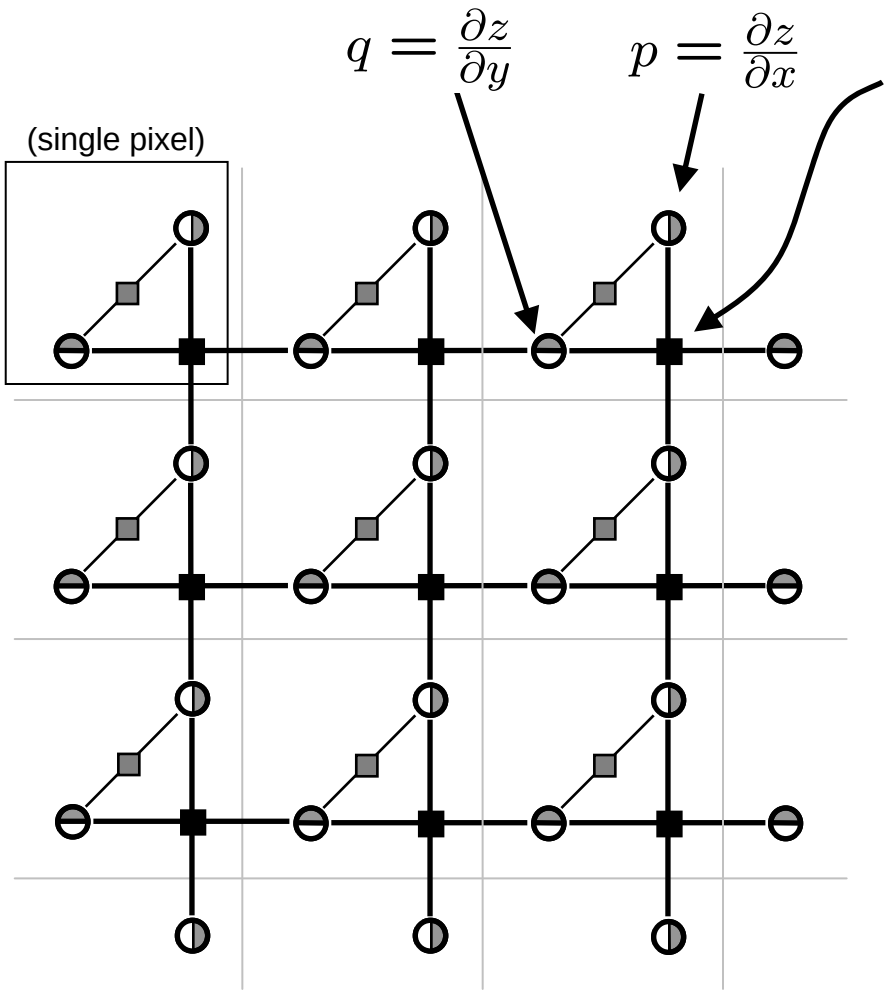


Zheng & Chellappa



R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(8):690–706, 1999

Shape from Shading Using LBP



Integrability Constraint Nodes

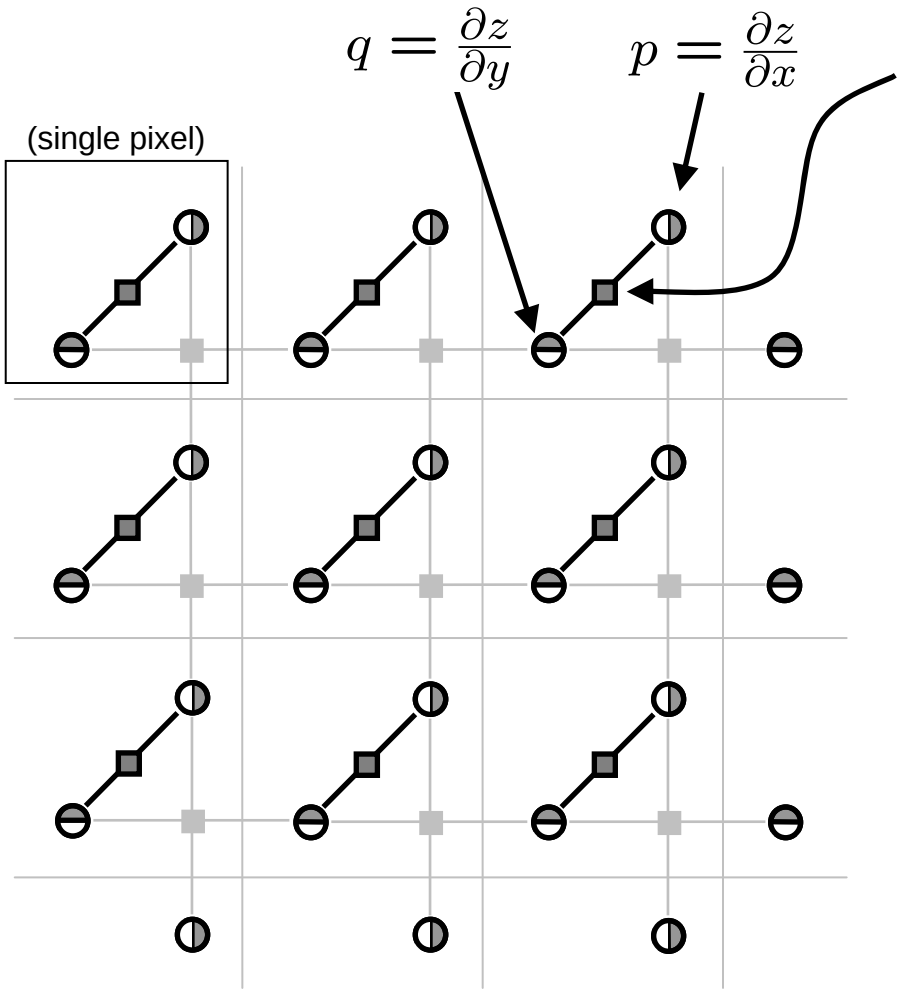
- Two variable nodes per pixel – representation is overcomplete
- Linear dependencies between variables must be enforced
- Real surfaces have zero curl

$$Z_{xy} = Z_{yx}$$

$$\frac{\partial p}{\partial y} = \frac{\partial q}{\partial x}$$

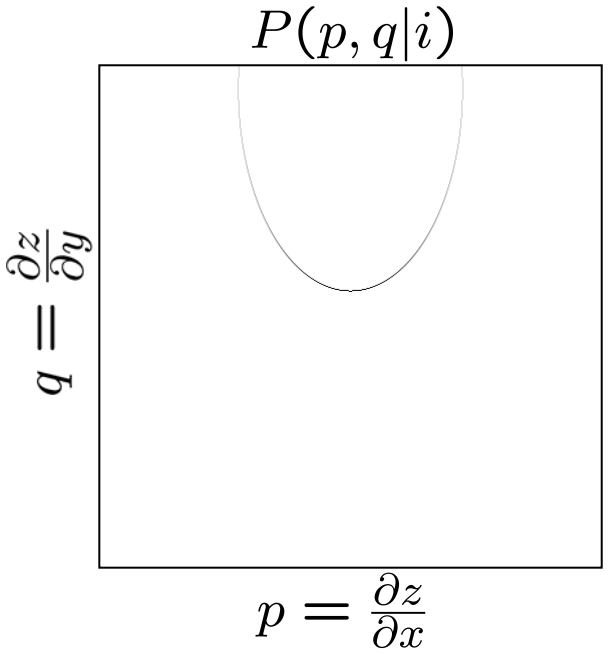
$$p_{x,y} - q_{x,y} + q_{x+1,y} - p_{x,y+1} = 0$$

Shape from Shading Using LBP

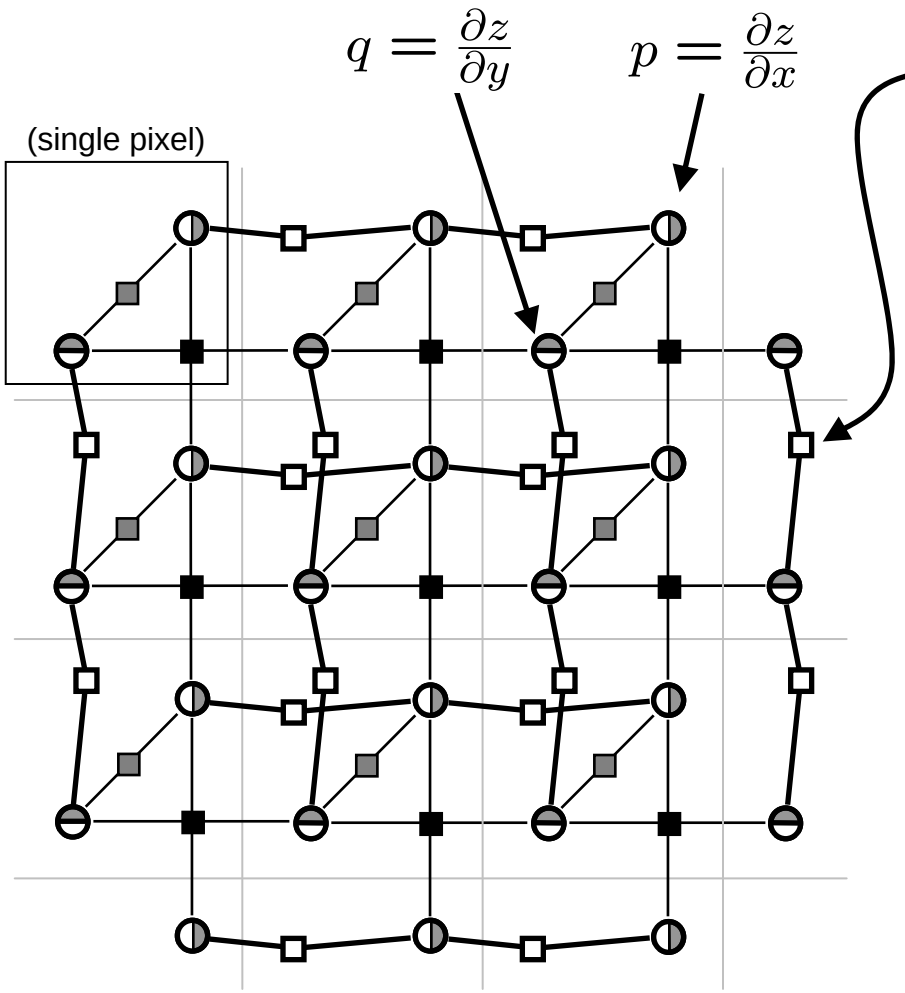


Lambertian Constraint Nodes

- Potential function is a level set of the Lambertian equation.
- Any BRDF could be used.



Shape from Shading Using LBP



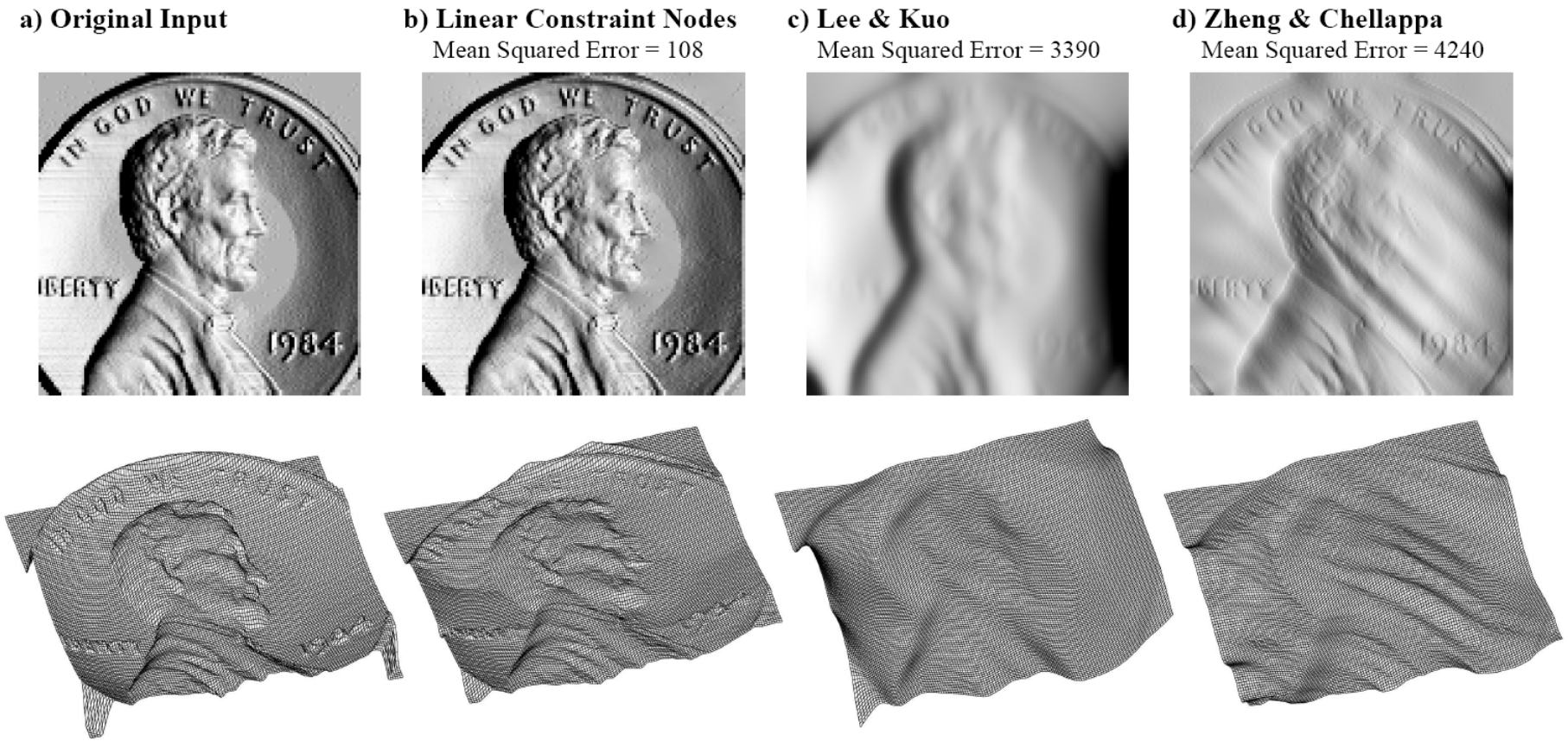
Smoothness Nodes

- Encode prior on $p_{x+1,y} - p_{x,y}$ and $q_{x,y+1} - q_{x,y}$
- Here, Laplace distributions are used.

$$P(Z) \propto \prod_{p,q} \exp(-\frac{|\Delta p| + |\Delta q|}{\sigma})$$

- Priors on $p_{x,y+1} - p_{x,y}$ and $q_{x+1,y} - q_{x,y}$ can be built into integrability nodes
- Smoothness nodes can also encode convexity priors.

Shape From Shading Results



Good image rerenderings imply that the Lambertian and Integrability constraints were wholly satisfied.

Improved performance can only be achieved through improving the spatial prior model p(Z).

Solving SFS using Belief Propagation offers several novel flexibilities:

- Any reflectance function can be used
  - handles matte or specular surfaces
- Any known light arrangement can be used
  - handles multiple or diffuse lighting
- Handles nondeterministic reflectance functions
  - handles uncertainty in lighting, reflectance, or albedo
- Handles attached shadows and missing data well.
- Computes marginals as well as point estimates.
- Can exploit rich spatial priors
  - can easily be combined with Fields of Experts
- Can be combined with other cues
  - Stereo, contours, texture, perspective, etc

Thank You!

Funded by:

NSF Graduate Research Fellowship to author

NSF IIS-0413211 To Tai Sing Lee

Thanks to:

Tai Sing Lee for helpful discussions

Stefan Roth along with X. Lan, D. P. Huttenlocher, M. J. Black for sharing filters & results.

Tom Stepleton