Nonlinear filtering and burst imaging

universität bonn

General Approach

$$i_{\text{denoised}}(x) = \frac{1}{\sum_{x'} w(x,x')} \sum_{x'} i_{\text{noisy}}(x') \cdot w(x,x')$$

- Many (not all) denoising techniques work like this
- Idea: average a number of "similar" pixels to reduce noise
- Question/difference: how similar are two noisy pixels?

General Approach

$$i_{\text{denoised}}(x) = \frac{1}{\sum_{x'} w(x, x')} \sum_{x'} i_{\text{noisy}}(x') \cdot w(x, x')$$

1. Local linear smoothing
2. Local non-linear filtering
3. Anisotropic diffusion
4. Non-local methods

1. Local, linear smoothing

$$i_{\text{denoised}}(x) = \frac{1}{\sum_{x'} w(x, x')} \sum_{x'} i_{\text{noisy}}(x') \cdot w(x, x')$$

$$w(x, x') = \exp\left(-\frac{\|x' - x\|^2}{2\sigma^2}\right)$$

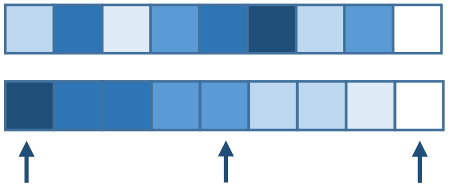
- Naïve approach: average in local neighborhood, e.g. using a Gaussian low-pass filter

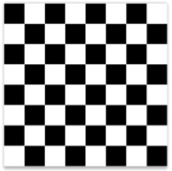
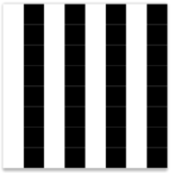
2. Local, non-linear filtering

$$i_{\text{denoised}}(x) = \text{median}\left(W\left(i_{\text{noisy}}, x\right)\right)$$

Small window of image i_{noisy}, centered at x

- Almost as naïve: use median filter in local neighborhood
- Sort pixels from darkest to brightest, take the one in the middle



- Quiz: $\text{median3x3}\Big($  $\Big) =$ 

Gaussian vs. median filters

Noisy input

3. Bilateral filtering

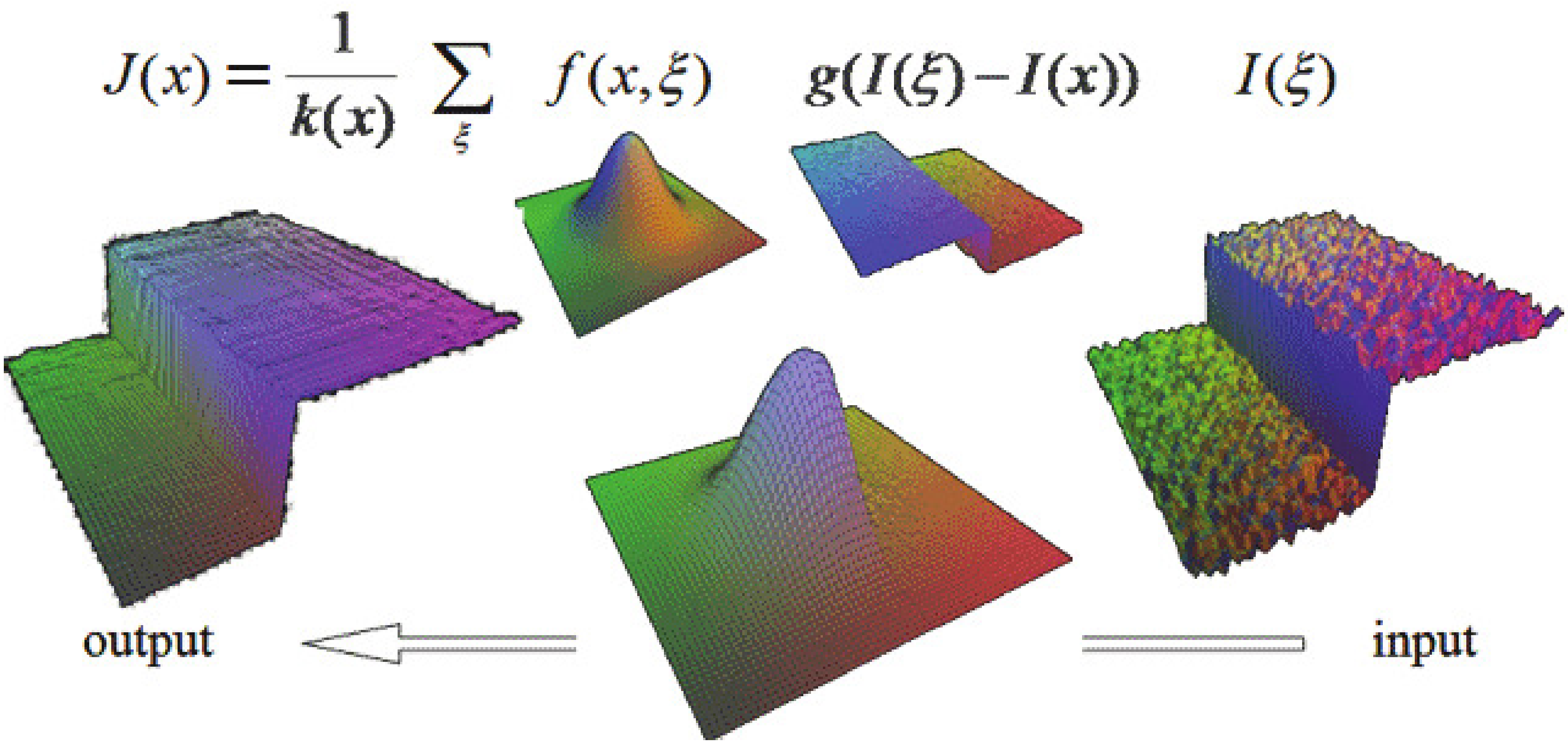
$$i_{\text{denoised}}(x) = \frac{1}{\sum_{x'} w(x,x')} \sum_{x'} i_{\text{noisy}}(x') \cdot w(x,x')$$

- $w(x,x') = \underbrace{\exp\left(-\frac{\|x'-x\|^2}{2\sigma^2}\right)}_{\text{Penalty for spatial distance}} \cdot \underbrace{\exp\left(-\frac{\|i_{noisy}(x')-i_{noisy}(x)\|^2}{2\sigma_i^2}\right)}_{\text{Penalty for intensity distance}}$

- More clever: average in local neighborhood, but give more weight to pixels of similar intensity

Bilateral filtering

- Illustration from Tomasi and Manduchi, ICCV 1998



Bilateral filter

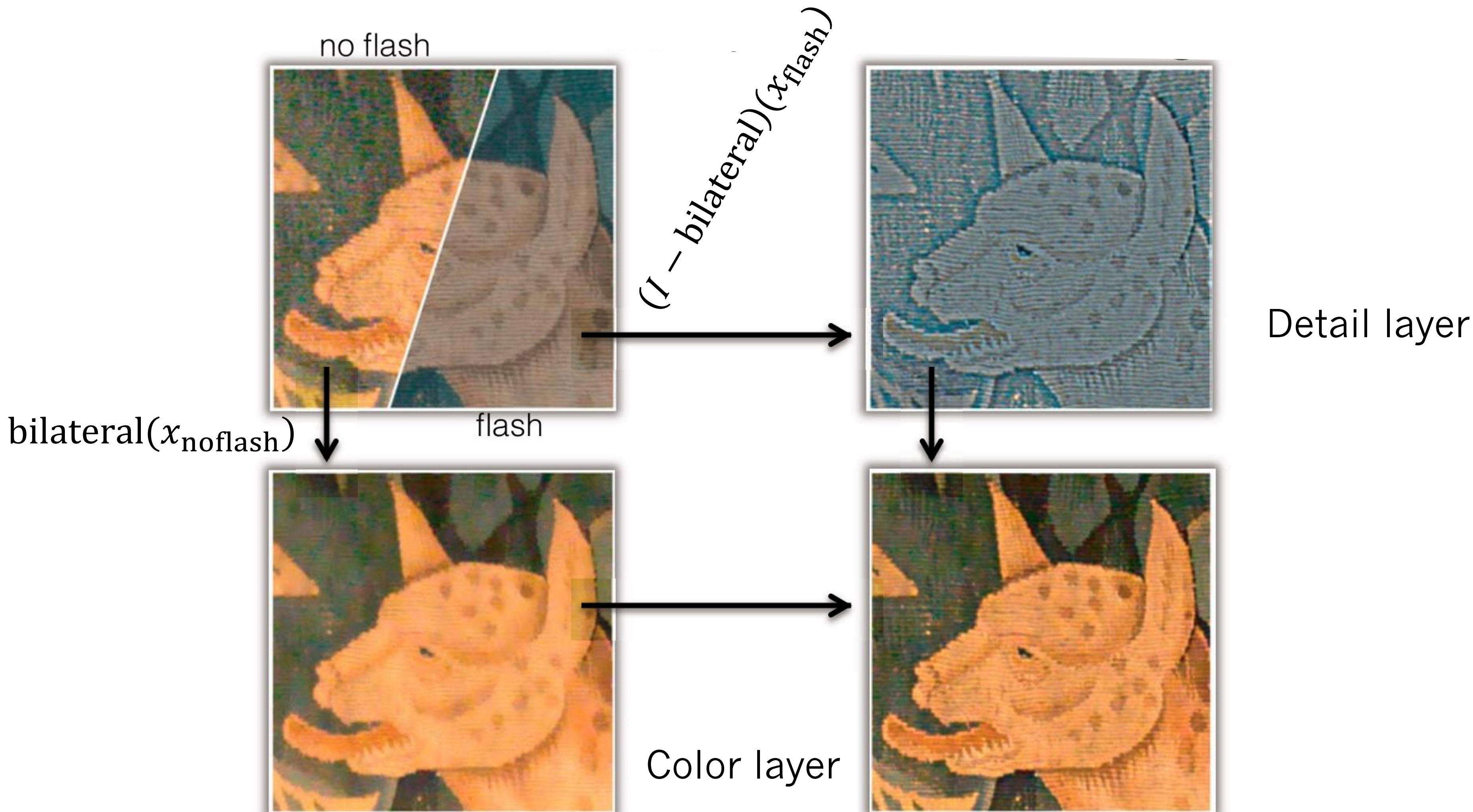
- $\sigma = 1; \sigma_i = 0.002, 0.005, 0.01, 0.05, 0.2$

- $\sigma = 3; \sigma_i = 0.002, 0.005, 0.01, 0.05, 0.2$

Gaussian vs. bilateral filter

- Both with spatial $\sigma = 3$

Bilateral filter for flash-no-flash imaging



Petschnigg et al. 2004

TV Regularization

Computational Photography
Matthias Hullin

universitätbonn

Last week

- Least-squares solution for linear problem $Cx = b$:

$$x_{opt} = \arg\min_{x} \frac{1}{2} \|Cx - b\|_2^2$$

$$\frac{1}{2} \|Cx - b\|_2^2 = \frac{1}{2} (Cx - b)^\top (Cx - b) =: \frac{1}{2} f(x)$$

Necessary condition for minimum x_{opt} of f:

$$\nabla f = (C^\top C) x_{\mathrm{opt}} - C^\top b = 0$$

Solve

$$(C^\top C) x_{\mathrm{opt}} - C^\top b = 0$$

$$\Leftrightarrow x_{\mathrm{opt}} = \underbrace{(C^\top C)^{-1} C^\top}_{} b$$

$C^+ = (C^\top C)^{-1} C^\top$
"Moore-Penrose pseudoinverse"

Last week: Deconvolution w/ Wiener filtering

$\sigma = 0.01$

$\sigma = 0.05$
("5% noise")

$\sigma = 0.1$

- Results: not too bad, but noisy
- This is a heuristic => dampen noise amplification

Total Variation (TV) regularization

$$\min_x \frac{1}{2}\|Cx - b\|_2^2 + \lambda TV(x)$$

"Data term" "Regularizer"

with

$$TV(x) = \|\nabla x\|_1 = \sum_i |(\nabla x)_i|$$

Idea: promote sparse gradients (edges)

∇ is finite difference operator, i.e. matrix

$$\begin{bmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \\ & & & -1 \end{bmatrix}$$

Total Variation

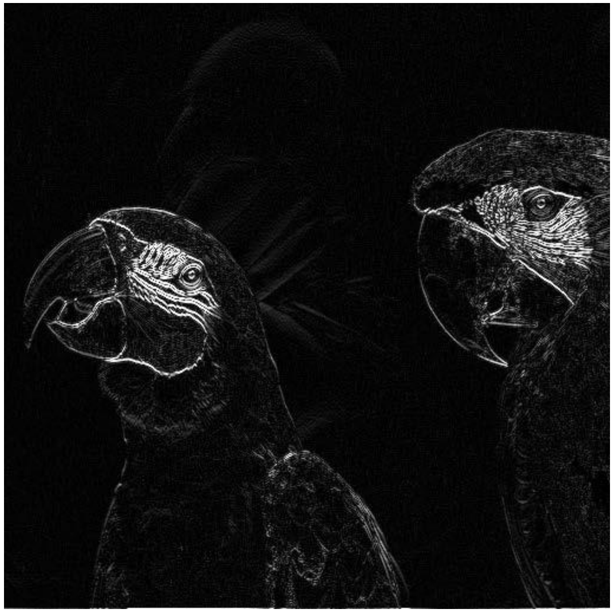
- Express (forward finite difference) gradient as convolution:

$$* \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad * \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

x

$\nabla_x x$

$\nabla_y x$



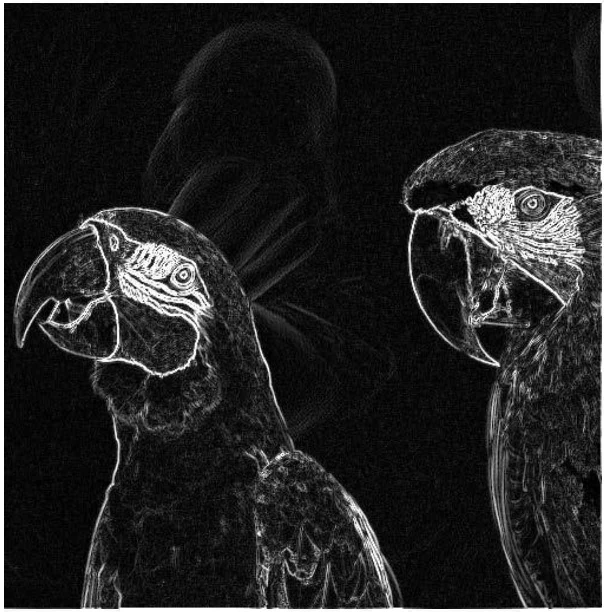
Total Variation

x

$\sqrt{(\nabla_x x)^2 + (\nabla_y x)^2}$

isotropic TV (better)

$\sqrt{(\nabla_x x)^2} + \sqrt{(\nabla_y x)^2}$



anisotropic TV (easier)

Total Variation

- For simplicity, we only discuss anisotropic TV:

$$TV(x) = \left\|\nabla_x x\right\|_1 + \left\|\nabla_y x\right\|_1 = \left\|\begin{bmatrix}\nabla_x \\ \nabla_y\end{bmatrix} x\right\|_1$$

- Problem: L1-norm is not differentiable, can't use inverse filtering

- To obtain simple solution for data fitting and simple solution for TV alone => split problem!

Deconvolution with ADMM

- Split deconvolution with TV prior:

$$\text{minimize} \quad \boxed{\frac{1}{2}\|Cx - b\|_2^2} + \boxed{\lambda\|z\|_1}$$
$$\text{subject to} \quad \boxed{\nabla x = z}$$

- General form of problem for ADMM (alternating direction method of multipliers):

$$\text{minimize} \quad \boxed{f(x)} + \boxed{g(z)}$$
$$\text{subject to} \quad \boxed{Ax + Bz = c}$$

$$f(x) = \boxed{\frac{1}{2}\|Cx - b\|_2^2}$$
$$g(z) = \boxed{\lambda\|z\|_1}$$
$$\boxed{A = \nabla, B = -I, c = 0}$$

S. Boyd et al., Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning 3, 1, 1–122, 2011.

ADMM

minimize $f(x) + g(z)$

subject to $Ax + Bz = c$

- Lagrangian (bring constraints into objective = penalty method)

Dual variable (Lagrange multiplier)

$$L(x, z, y) = f(x) + g(z) + y^\top(Ax + Bz - c)$$

- Augmented Lagrangian (differentiable under mild conditions – better convergence

$$L_\rho(x, z, y) \\ = f(x) + g(z) + y^\top(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2$$

ADMM

minimize $f(x) + g(z)$

subject to $Ax + Bz = c$

- ADMM consists of 3 steps per iteration k:

- $x^{k+1} \leftarrow \arg\min_x L_\rho(x, z^k, y^k)$
- $z^{k+1} \leftarrow \arg\min_z L_\rho(x^{k+1}, z, y^k)$
- $y^{k+1} \leftarrow y^k + \rho(Ax^{k+1} + Bz^{k+1} - c)$

These are called “proximal” or “proximity” operators

ADMM

minimize $f(x) + g(z)$
subject to $Ax + Bz = c$

- Plug in L_ρ; introduce scaled dual variable $u = \frac{1}{\rho} y$:

constant

- $x^{k+1} \leftarrow \arg\min_x \left(f(x) + \frac{\rho}{2} \| Ax + Bz^k - c + u^k \|_2^2 \right)$
- $z^{k+1} \leftarrow \arg\min_z \left(g(z) + \frac{\rho}{2} \| Ax^{k+1} + Bz - c + u^k \|_2^2 \right)$
- $u^{k+1} \leftarrow u^k + Ax^{k+1} + Bz^{k+1} - c$

- $f(x)$ and $g(z)$ are now split into independent problems, connected by u

universitätbonn

Deconvolution with ADMM

minimize $\frac{1}{2}\|Cx - b\|_2^2 + \lambda\|z\|_1$

subject to $Dx - z = 0$ (here with $D = \begin{bmatrix} \nabla_x \\ \nabla_y \end{bmatrix}$)

constant

- $x^{k+1} \leftarrow \underset{x}{\arg\min} \left(\frac{1}{2}\|Cx - b\|_2^2 + \frac{\rho}{2}\left\|Dx - z^k + u^k\right\|_2^2\right)$
- $z^{k+1} \leftarrow \underset{z}{\arg\min} \left(\lambda\|z\|_1 + \frac{\rho}{2}\left\|Dx^{k+1} - z + u^k\right\|_2^2\right)$
- $u^{k+1} \leftarrow u^k + Dx^{k+1} - z^{k+1}$

Deconvolution with ADMM

minimize $\frac{1}{2}\|Cx-b\|_2^2+\lambda\|z\|_1$

subject to $Dx-z=0$ (here with $D=\begin{bmatrix}\nabla_x\\ \nabla_y\end{bmatrix}$)

1. x-update

Constant during update:
$v := z^k - u^k$

- $x^{k+1} \leftarrow \underset{x}{\arg\min}\left(\frac{1}{2}\|Cx-b\|_2^2+\frac{\rho}{2}\|Dx-v\|_2^2\right)$

Solve normal equations:

- $x^{k+1} \leftarrow (C^\top C+\rho D^\top D)^{-1}(C^\top b+\rho D^\top v)$

or use large-scale, iterative method (gradient descent, conjugate gradient, SART). For 2D deconvolution, we can also use inverse filtering

Deconvolution with ADMM

minimize $\frac{1}{2}\|Cx - b\|_2^2 + \lambda\|z\|_1$

subject to $Dx - z = 0$

2. z-update

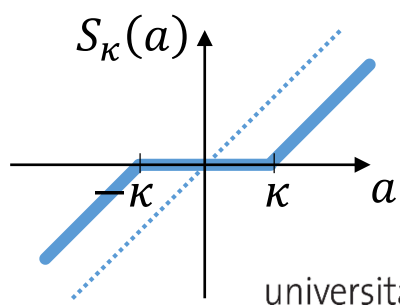
Constant during update:
$w := Dx^{k+1} + u^k$

- $z^{k+1} \leftarrow \underset{z}{\arg\min} \left(\lambda\|z\|_1 + \frac{\rho}{2}\|w - z\|_2^2\right)$

Not differentiable, but there is a closed-form solution:

- $z^{k+1} \leftarrow S_{\lambda/\rho}(w)$ with "soft thresholding" function

$$S_\kappa(a) = \begin{cases} a - \kappa, & a > \kappa \\ a + \kappa, & a < -\kappa \\ 0, & \text{else} \end{cases}$$



Deconvolution with ADMM

minimize $\frac{1}{2}\|Cx - b\|_2^2 + \lambda\|z\|_1$

subject to $Dx - z = 0$

1. Construct C, D, b; initialize ρ and λ
2. $x \leftarrow zeros(W \cdot H)$
3. $z, u \leftarrow zeros(2 \cdot W \cdot H)$
4. $\textbf{for } k = 1 \textbf{ to } \text{max\_iters } \textbf{\textit{do}}$
5. $\quad x \leftarrow \arg\min_x \frac{1}{2}\left\|\begin{bmatrix} C \\ \rho D \end{bmatrix} x - \begin{bmatrix} b \\ \rho(z-u) \end{bmatrix}\right\|_2^2$
6. $\quad z \leftarrow S_{\lambda/\rho}(Dx + u)$
7. $\quad u \leftarrow u + Dx - z$
8. $\textbf{end for}$

$D = \nabla$ produces 2 gradient values (x and y derivative) per pixel

Deconvolution

Wiener filtering

ADMM with anisotropic TV

The characteristic look of TV

- Too little TV: noisy; too much TV: “patchy”

$\lambda = 0.01, \rho = 10$

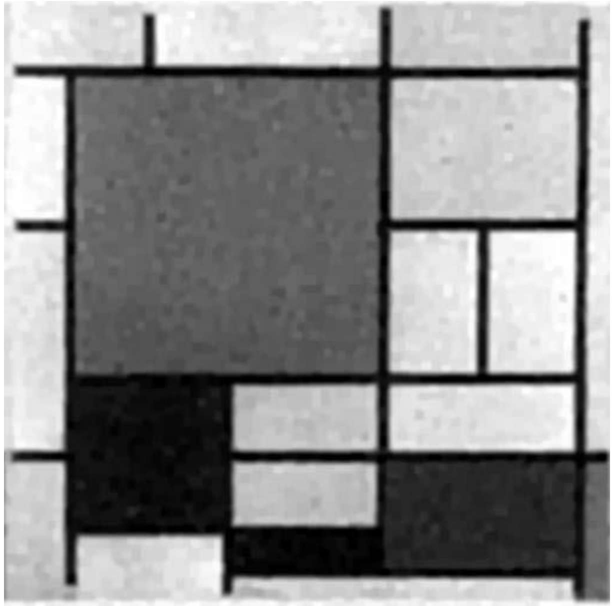
$\lambda = 0.05, \rho = 10$

$\lambda = 0.1, \rho = 10$

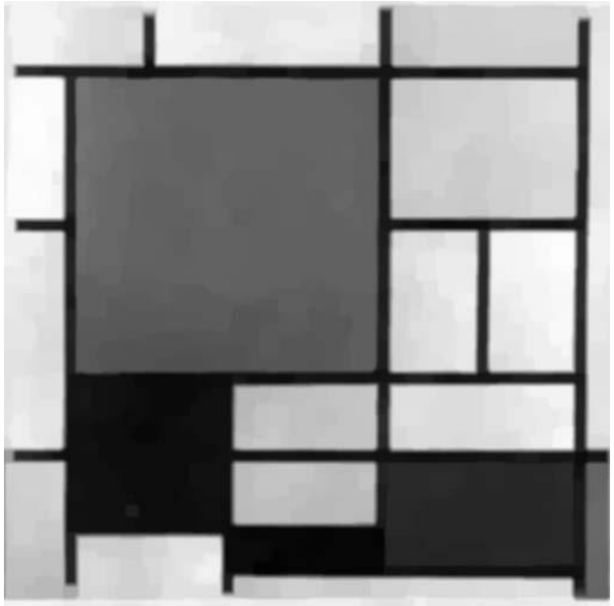
The characteristic look of TV

- Here, too much TV is okay because the image actually has sparse gradients!

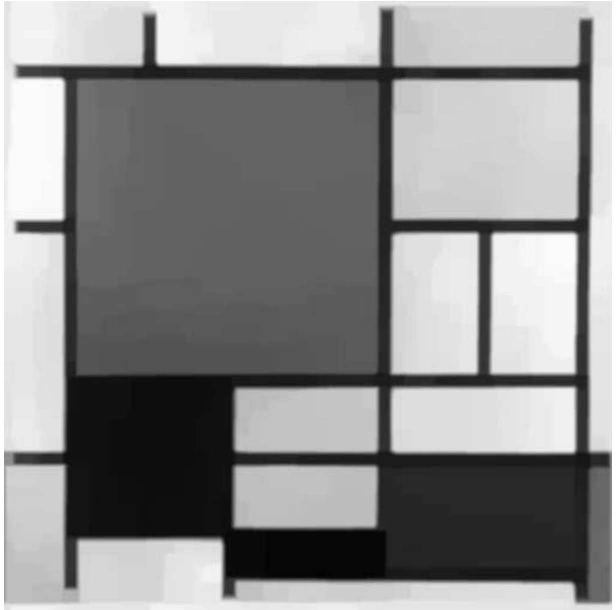
$\lambda = 0.01, \rho = 10$



$\lambda = 0.05, \rho = 10$



$\lambda = 0.1, \rho = 10$



What else to do with it?

Solve all sorts of linear inverse problems

- Denoising
- Superresolution
- Tomographic reconstruction
- Compressed sensing (more next week)

Denoising

minimize $\frac{1}{2}\|Cx - b\|_2^2 + \lambda\|z\|_1$

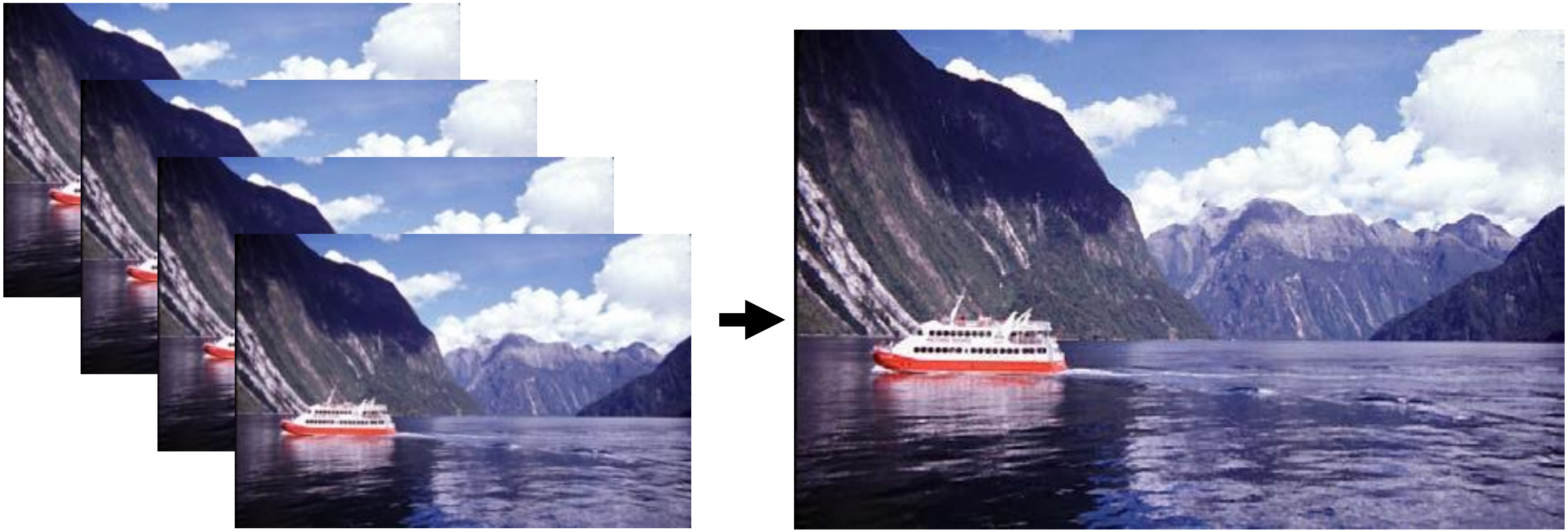
subject to $Dx - z = 0$

$C = I$ (for noisy but sharp images)

$D = \nabla$ (TV penalty)

Pixel super-resolution

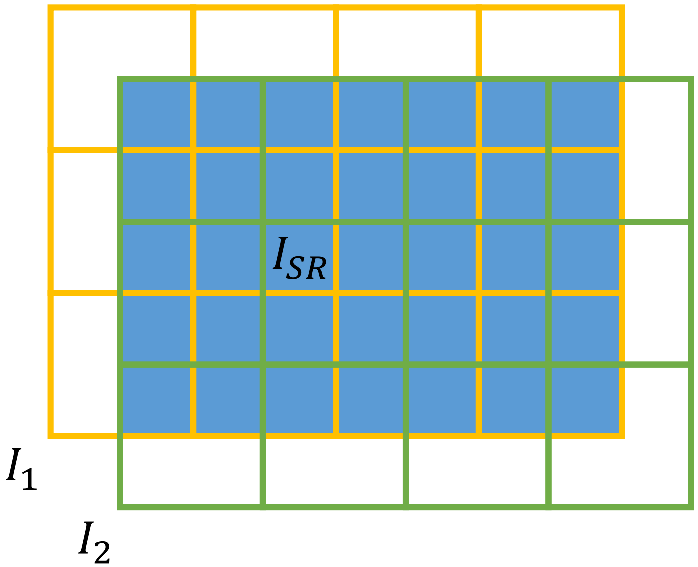
- Increase pixel count by combining multiple images of low resolution (LR) into one super-resolved (SR) image



[Ben-Ezra et al., Jitter Camera [...], CVPR 2004]

Pixel super-resolution

- LR images must be sub-pixel shifted



minimize $\frac{1}{2}\|Cx - b\|_2^2 + \lambda\|z\|_1$
subject to $Dx - z = 0$

with $C = \begin{bmatrix} C_1 \\ C_2 \\ \vdots \end{bmatrix}$, $x = I_{SR}$ and $b = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \end{bmatrix}$

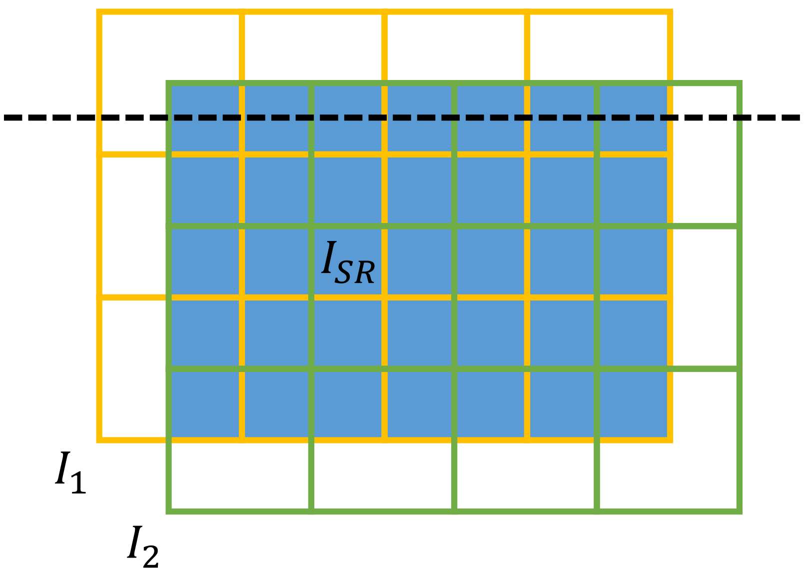
(downsampling operators) (stacked input images)

$D = \nabla$ adds TV regularization if needed

Pixel super-resolution

- LR images must be sub-pixel shifted

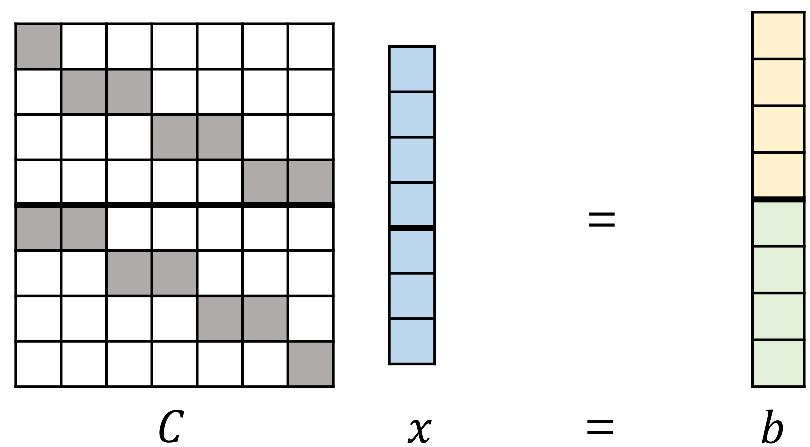
$$\text{minimize} \;\; \frac{1}{2}\|Cx - b\|_2^2 + \lambda\|z\|_1$$
$$\text{subject to } Dx - z = 0$$



$$\text{with } C = \begin{bmatrix} C_1 \\ C_2 \\ \vdots \end{bmatrix}, \quad x = I_{SR} \quad \text{and} \quad \text{b} = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \end{bmatrix}$$

(downsampling operators)

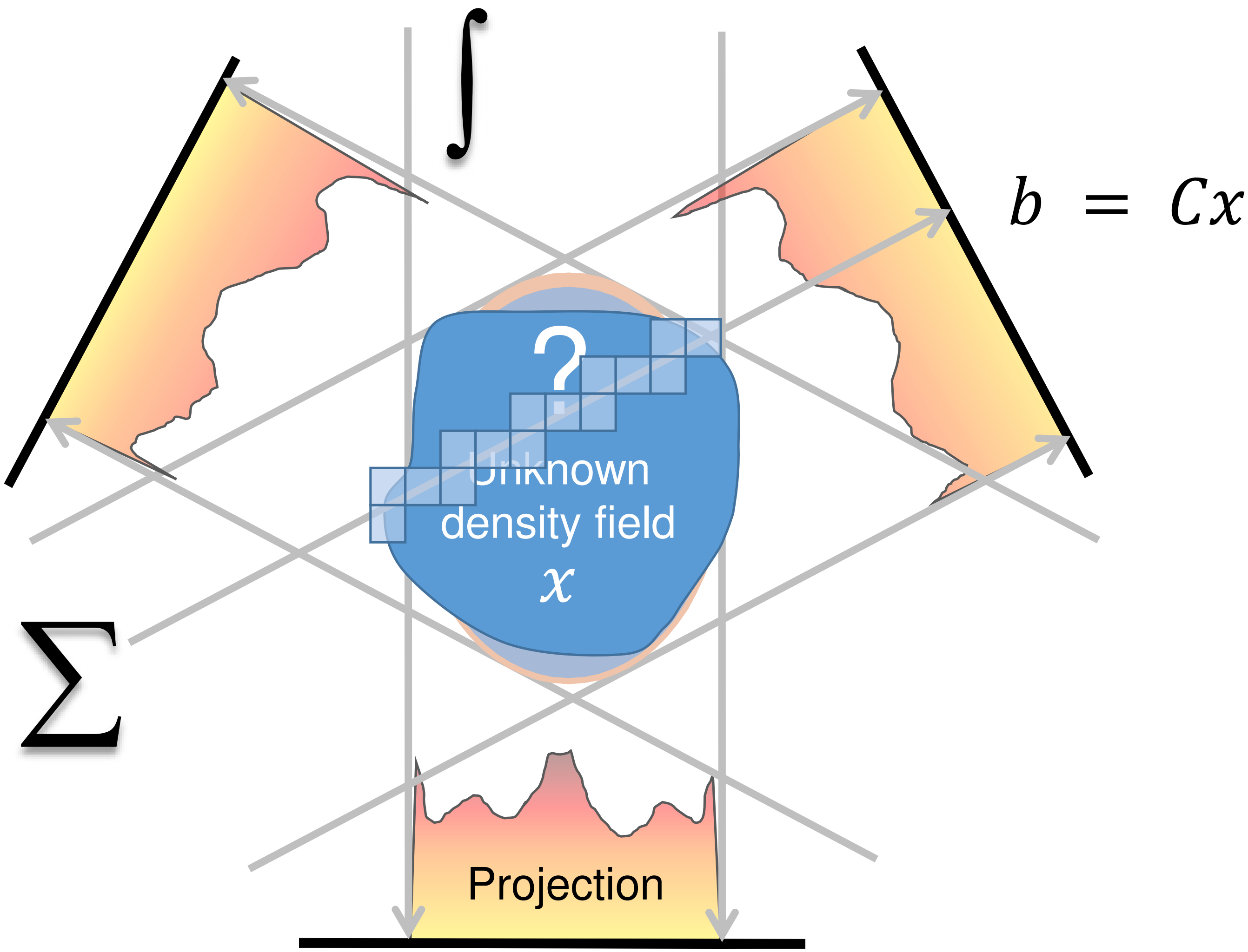
(stacked input images)



Pixel super-resolution

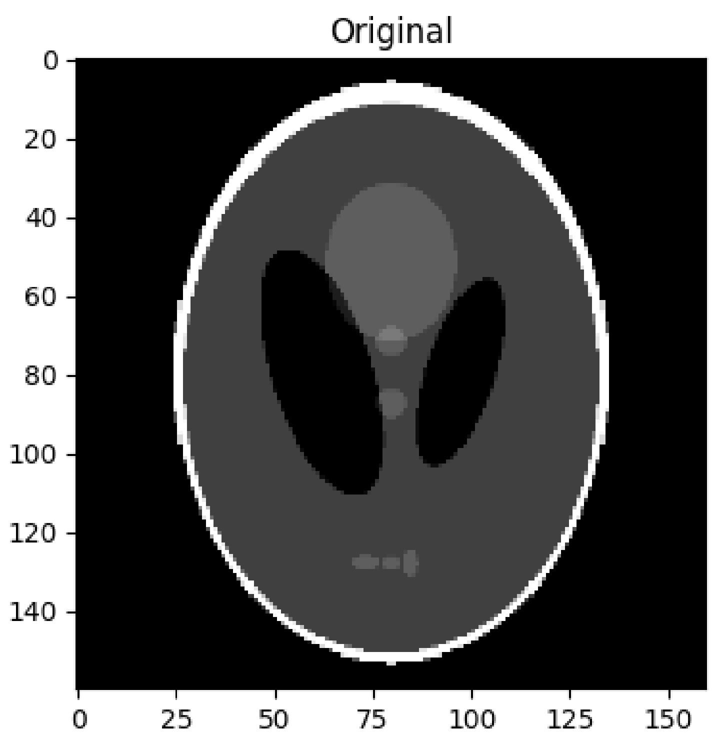
- In general, system is well-conditioned for non-integer pixel shifts and super-resolution factors of 2-3x (don't necessarily need priors/regularization)

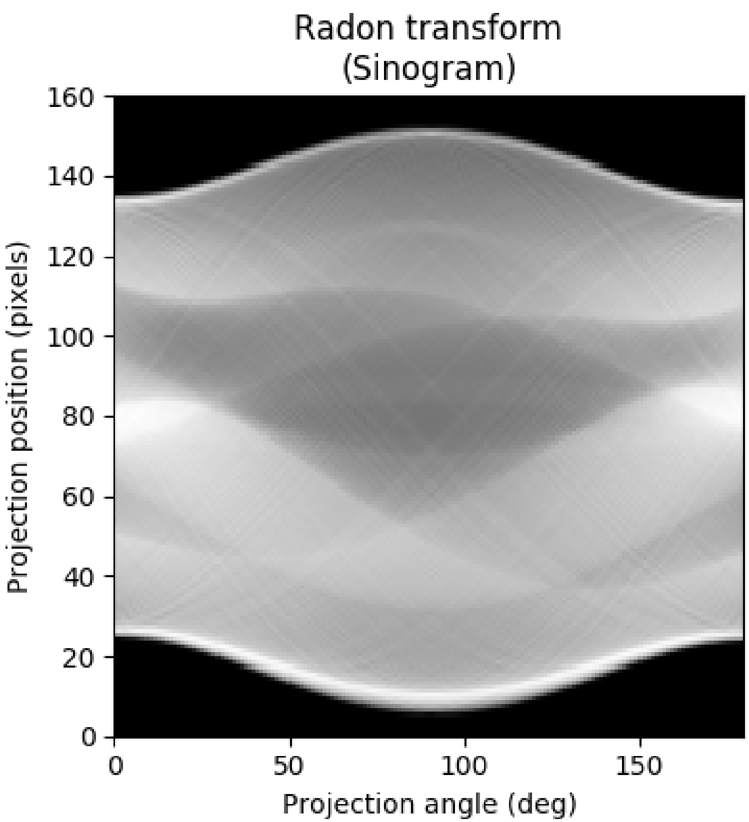
Tomographic reconstruction



Radon transform

- Image formation in tomographic imaging systems described by Radon transform (= all directional projections)
- Like Fourier, it's a basis transform





Source: http://scikit-image.org

Literature

- Tomasi and Manduchi, “Bilateral filtering for gray and color images”, ICCV 1998
- Petschnigg, Agrawala, Hoppe, Szeliski, Cohen and Toyama, “Digital Photography with Flash and No-Flash Image Pairs”, SIGGRAPH 2004
- S. Boyd et al., “Distributed optimization and statistical learning via the alternating direction method of multipliers.” Foundations and Trends in Machine Learning 3, 1, 1–122, 2011
- Ben-Ezra, Zomet and Nayar, “Jitter Camera: High Resolution Video from a Low Resolution Detector”, CVPR 2004