

# Super-resolution on Text Image Sequences

Louka Dlagnekov

November 4, 2004

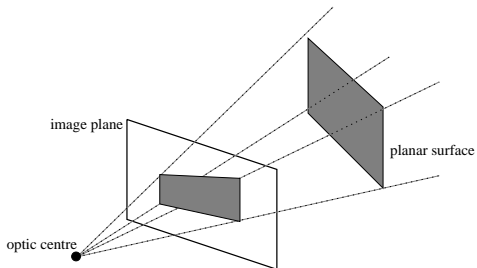
# Outline

- Introduction
- Imaging Model
- Methods
  - Maximum Likelihood
  - Irani and Peleg Algorithm
  - Maximum A Posterior
  - Total Variation
- Applications
- Questions

# Basic Idea

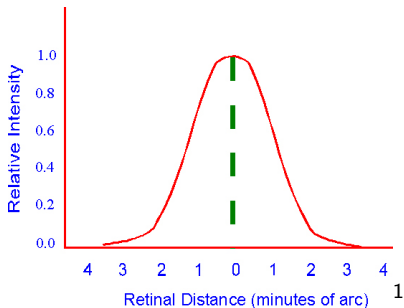
- No optical/image acquisition system is perfect
- Types of degradation:
  - Geometric distortion
  - Optical/motion blurring
  - Down-sampling
  - Additive noise
- Our goal is to compensate for this and produce a higher resolution image given a sequence of low-resolution images.

# Geometric Distortion



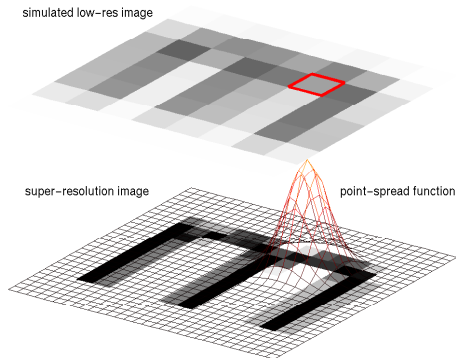
# Optical/Motion Blurring

- Optical blurring in eye:



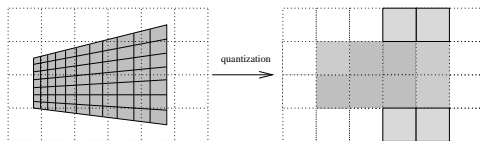
<sup>1</sup><http://www.yorku.ca/eye/psf.htm>

# Optical/Motion Blurring

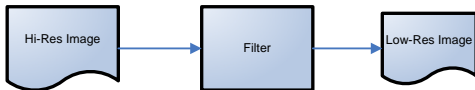


- PSF is usually unknown, but Capel and Zisserman chose Gaussian and saw good results.

# Down-Sampling



# Imaging Model



- Need Mathematical model that represents process

$$\hat{m}_n = S \downarrow (h * T [I(s)]) + \eta \quad (1)$$

- At each point  $(x, y)$ :

$$\hat{m}_n(x, y) = S \downarrow (h(u, v) * s(T(x, y))) + \eta \quad (2)$$

## In Matrix Form...

$$\hat{m}_n = S \downarrow (h * T [I(s)]) + \eta$$

Becomes:

$$\hat{m}_n = M_n s + \eta \quad (3)$$

where  $h$ ,  $T$ ,  $S \downarrow$  are combined in the matrix  $M_n$ .

For all images, we can stack vertically for an over-determined linear system:

$$\begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{N-1} \end{bmatrix} = \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_{N-1} \end{bmatrix} + \begin{bmatrix} \eta_0 \\ \eta_1 \\ \vdots \\ \eta_{N-1} \end{bmatrix} \quad (4)$$

$$m = Ms + \eta \quad (5)$$

# Maximum Likelihood Estimator

Total probability of observed image  $m_n$ , given estimate of super-resolution image  $\hat{s}$  is:

$$Pr(m_n|\hat{s}) = \prod_{\forall x,y} \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(\hat{m}_n(x,y) - m_n(x,y))^2}{2\sigma^2}} \quad (6)$$

And the associated log-likelihood function is:

$$L(m_n) = - \sum_{\forall x,y} (\hat{m}_n(x,y) - m_n(x,y))^2 \quad (7)$$

# Maximum Likelihood Estimator

To find the maximum likelihood estimate,  $s_{ML}$ , we need to maximize  $L(m_n)$  over all images:

$$s_{ML} = \operatorname{argmax}_s \sum_n L(m_n) \quad (8)$$

Using the Matrix form for the image model from before, we have:

$$\sum_n L(m_n) = - \sum_n \|M_n s - m_n\|^2 = - \|Ms - m\|^2$$

The maximization above, becomes equivalent to:

$$s_{ML} = \operatorname{argmin}_s \|Ms - m\|^2$$

# Maximum Likelihood Estimator

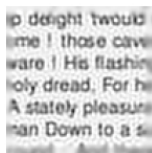
This is now a standard minimization problem:

$$s_{ML} = \underbrace{(M^T M)^{-1} M^T}_{M^+} m \quad (9)$$

Because  $M$  is very large and sparse and  $Nn^2 \times m^2$  matrix and typically  $N = 30$ ,  $n^2 = 2500$ ,  $m^2 = 10000$ , it is impractical to compute pseudo-inverse. Need iterative methods like conjugate gradient.

# Problems

Very sensitive to noise:



noise  $\sigma = 1.0$



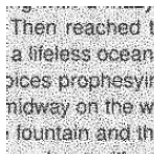
noise  $\sigma = 2.0$



noise  $\sigma = 3.0$

# Bounded Maximum Likelihood

To Solve, use Bounded Maximum Likelihood Estimator:



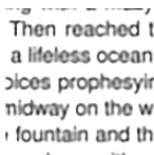
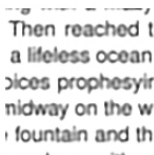
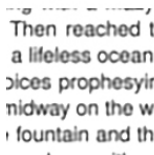
$$\sigma_n = 0.25$$



$$\sigma_n = 0.50$$



$$\sigma_n = 0.75$$



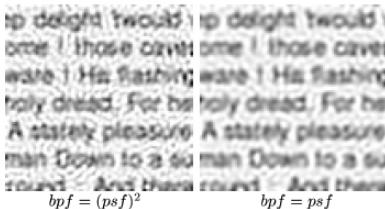
# Irani and Peleg Algorithm

Minimizes same cost function as ML estimator, but uses iterative update with error back-projection:

$$s^{i+1} = s^i + \frac{1}{C} \sum_{\forall n} T_n^{-1} [h_{bpf} * S \uparrow (\hat{m}_n - m_n)] \quad (10)$$

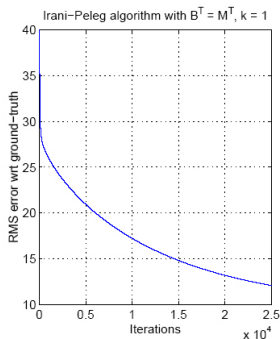
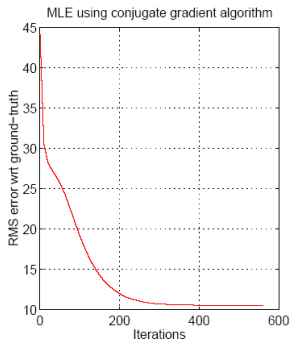
where the back-projection function  $h_{bpf} = (h_{psf})^k$ , for some  $k \geq 1$ .

# Irani and Peleg Algorithm Results



# Problems

Very slow to converge compared to Conjugate-Gradient minimization method in MLE:



# Maximum A Posterior

In the MLE, we used:

$$Pr(m_n|\hat{s}) = \prod_{\forall x,y} \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(\hat{m}_n(x,y)-m_n(x,y))^2}{2\sigma^2}}$$

Assuming independent observations, the total probability over all images in sequence is:

$$Pr(m|\hat{s}) = \prod_{\forall n} Pr(m_n|\hat{s}) \quad (11)$$

Using Bayes's theorem:

$$Pr(\hat{s}|m) = \frac{Pr(m|\hat{s})Pr(\hat{s})}{Pr(m)} \quad (12)$$

# Maximum A Posterior

The *maximum a-posterior* (MAP) estimate of  $s$  becomes:

$$\begin{aligned} s_{MAP} &= \operatorname{argmax}_s \frac{\Pr(m|\hat{s})\Pr(\hat{s})}{\Pr(m)} \\ &= \operatorname{argmax}_s \Pr(m|\hat{s})\Pr(\hat{s}) \end{aligned}$$

Taking the logarithms, we get:

$$s_{MAP} = \operatorname{argmax}_s \ln\Pr(m|\hat{s}) + \ln\Pr(\hat{s}) \quad (13)$$

$$= \operatorname{argmax}_s \sum_n L(m_n) + L(s) \quad (14)$$

# Likelihood of $s$ estimate

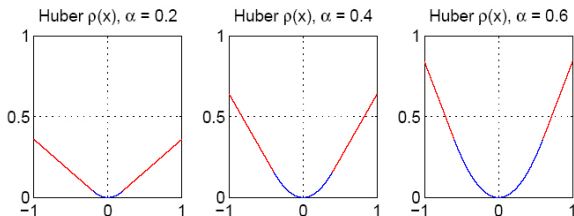
- How do we know what  $L(s)$  is?
- In this paper, Capel and Zisserman suggest a Huber cost function,  $f(x)$ :

$$s_{MAP} = \operatorname{argmax}_s \sum_n L(m_n) - \lambda^2 \sum_{\forall x,y} f(\nabla s(x,y)) \quad (15)$$

$$\begin{aligned} f(x) &= x^2, \text{ if } x \leq \alpha \\ &= 2\alpha|x| - \alpha^2, \text{ otherwise} \end{aligned}$$

# Likelihood of $s$ estimate

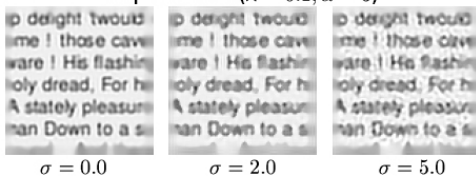
- But, why the Huber function?



- Allows for local smoothness while being lenient toward step edges

# Results

Map estimator ( $\lambda = 0.1, \alpha = 5$ )



# Total Variation Estimator

Same idea as MAP estimator, but use different prior:

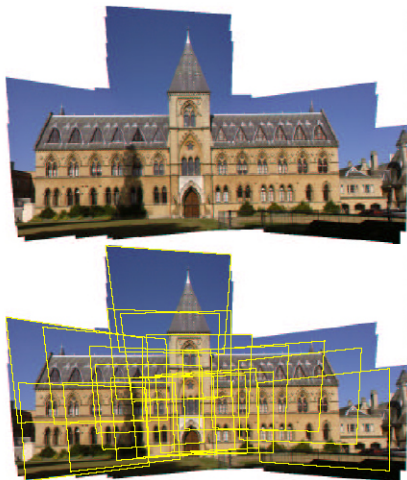
$$s_{TV} = \operatorname{argmax}_s \sum_n L(m_n) - \lambda^2 \sum_{\forall x,y} |\nabla s(x,y)| \quad (16)$$

Since,  $\frac{dT_V}{ds} = - \int \frac{\nabla \cdot \nabla s}{|\nabla s|}$ , we can use

$$|\nabla s| \approx \sqrt{s_x^2 + s_y^2 + \beta}$$

to avoid singularity at  $\nabla s = 0$ .

# Mosaicking



# License Plate Recognition



# Questions

- How do we get from Eqn 1 to Eqn 2?
- Are we using a common region of the low resolution frames to get the super-resolution image? Otherwise, some frames may not contain the whole area of the image that we are interested in.

## Recommended Resources

- David Capel's PhD Thesis: "Image Mosaicing and Super-resolution", 2001
- Capel and Zisserman. "Computer Vision Applied to Super-resolution". IEEE Signal Processing Magazine (2003)  
<http://www.robots.ox.ac.uk/vgg/publications/papers/capel03.pdf>