Digital Image Processing

Berlin University of Technology (TUB),
Computer Vision and Remote Sensing Group
Berlin, Germany

Image Restoration

- Image analysis often begins with pre-processing

Enhancement

- Contrast, brightness, sharpening etc.
- Working with information inherent in the signal

Restoration

- Sensor defects (noise, blur)
- Unfortunate conditions (moving objects)
- Ageing originals
- Restoring information that has been lost

Image Restoration

Image Restoration – Inverse Filter

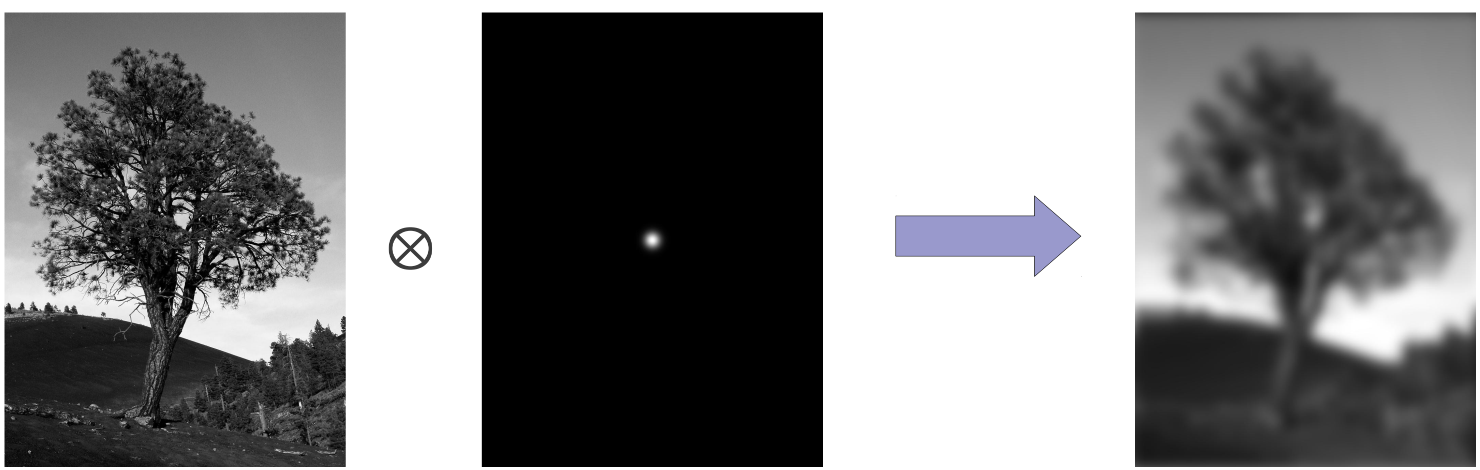
Signal Model

$$s_i = \sum_j o_j \, p_{i-j} \xrightarrow{\text{FFT}} S_i = O_i \cdot P_i$$

s_i: Observed signal
o_i: Original signal
p_i: Impulse response

S_i: Spectrum of the observed signal
O_i: Spectrum of the original signal
P_i: Spectrum of the impulse response

- E.g. camera with a small aperture (kernel p causes blurring)



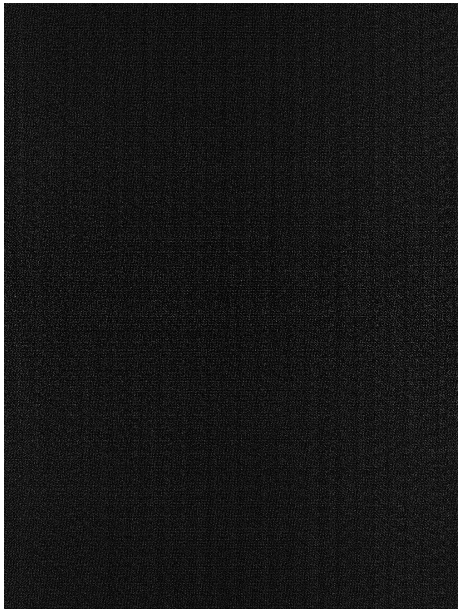
o_i $\quad$ p_i $\quad$ s_i

Image Restoration – Inverse Filter

- **Convolution Theorem**:
  - Convolution is equivalent to multiplication in the frequency domain
  - Multiplication is easly reversed by division!

$$S_i = O_i \cdot P_i \longrightarrow O_i = S_i \cdot 1/P_i \longrightarrow o_i = IFFT(1/P_i) \otimes s_i$$

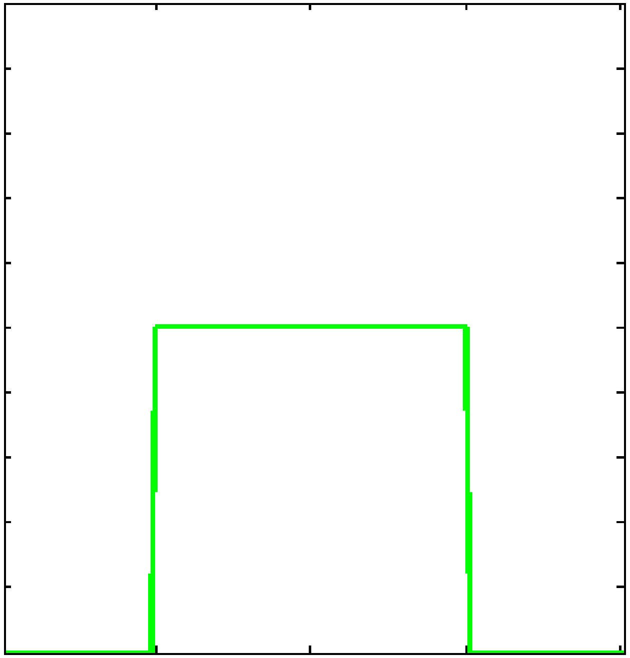
$$\otimes \; IFFT(1/P_i) =$$



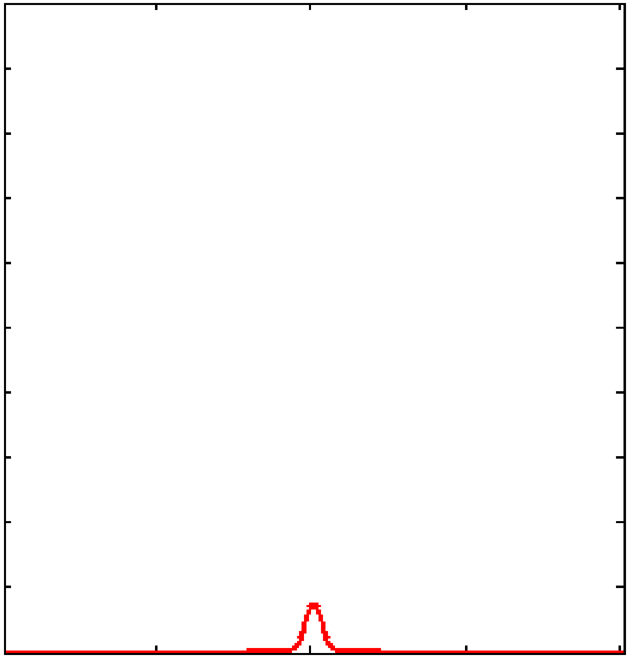
?

Image Restoration – Inverse Filter

- **Problem**: P_i is practically equal to zero in some parts of the spectrum
  - Lowpass filters (blur) induce $P_i = 0$ at high frequencies
  - Inversion is not feasible due to limited numerical accuracy
  - E.g. small inaccuracies in the FFT are strongly amplified



$\otimes$



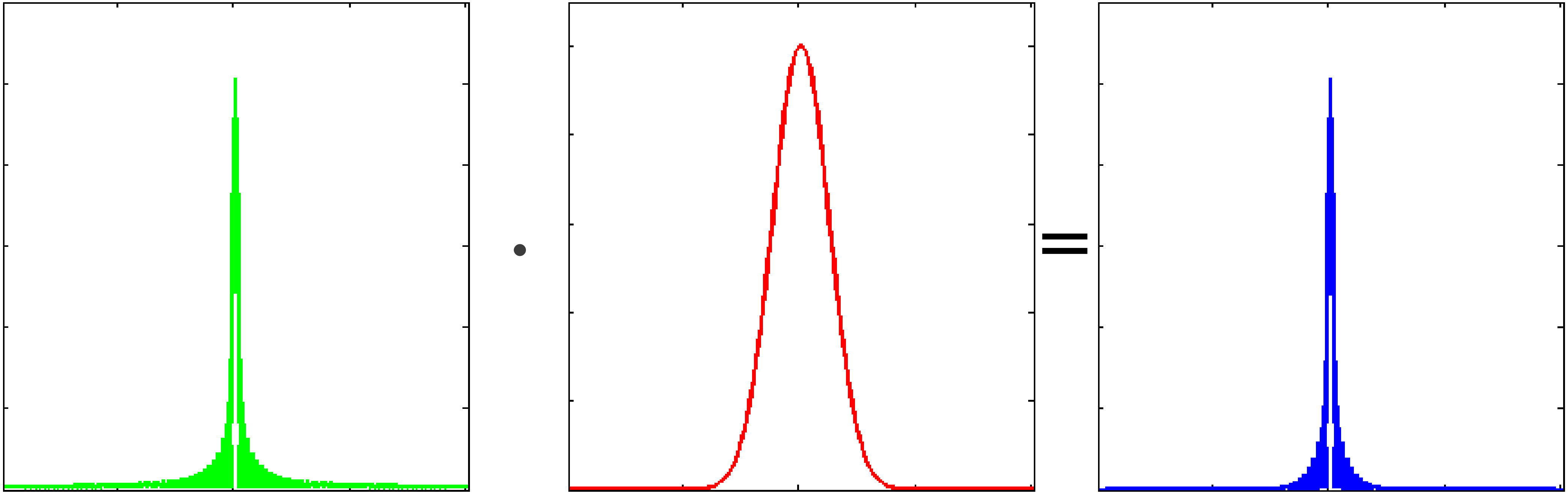
$=$

o

p

Image Restoration – Inverse Filter

- **Problem**: P_i is practically equal to zero in some parts of the spectrum
  - Lowpass filters (blur) induce $P_i = 0$ at high frequencies
  - Inversion is not feasible due to limited numerical accuracy
  - E.g. small inaccuracies in the FFT are strongly amplified



O = FFT(o) P = FFT(p) S

Image Restoration – Inverse Filter

- **Problem**: P_i is practically equal to zero in some parts of the spectrum
  - Lowpass filters (blur) induce $P_i = 0$ at high frequencies
  - Inversion is not feasible due to limited numerical accuracy
  - E.g. small inaccuracies in the FFT are strongly amplified

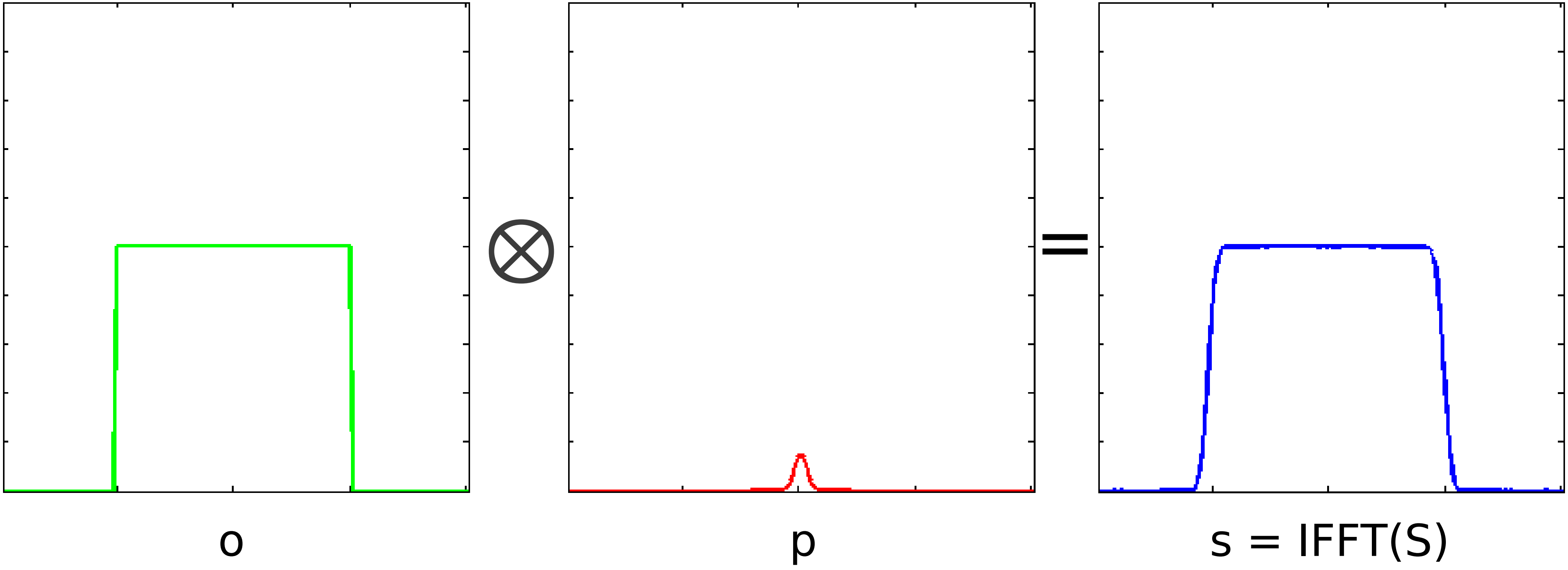
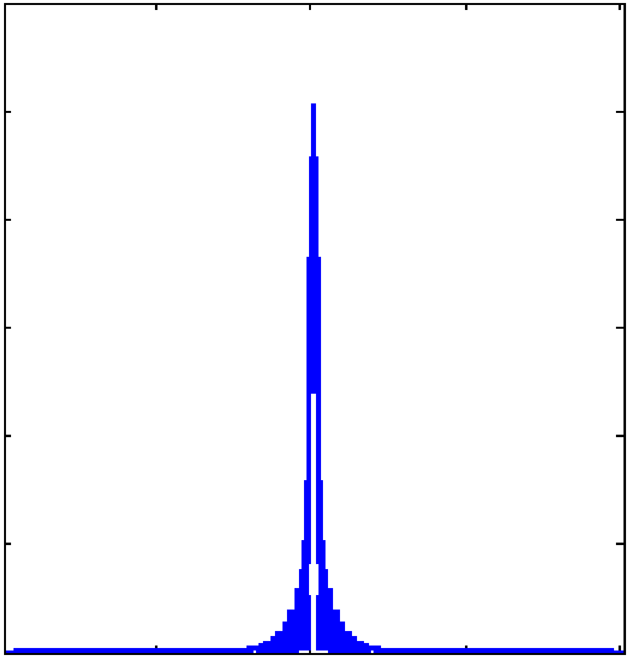


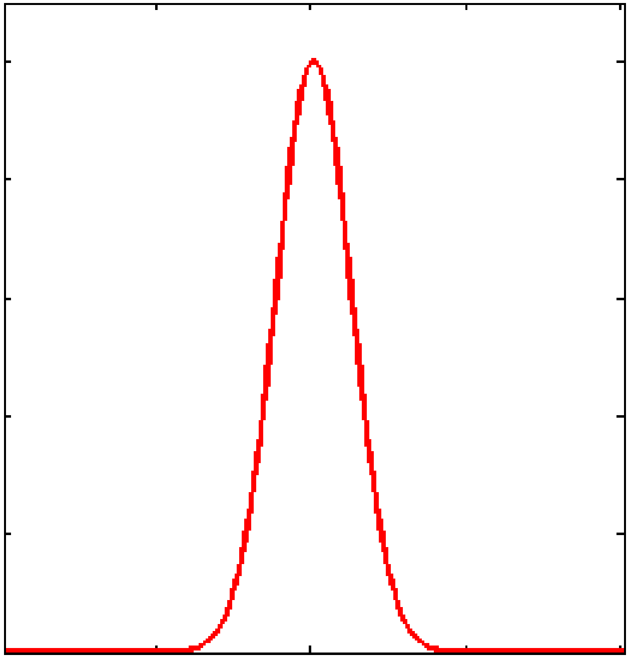
Image Restoration – Inverse Filter

- **Problem**: P_i is practically equal to zero in some parts of the spectrum
  - Lowpass filters (blur) induce $P_i = 0$ at high frequencies
  - Inversion is not feasible due to limited numerical accuracy
  - E.g. small inaccuracies in the FFT are strongly amplified



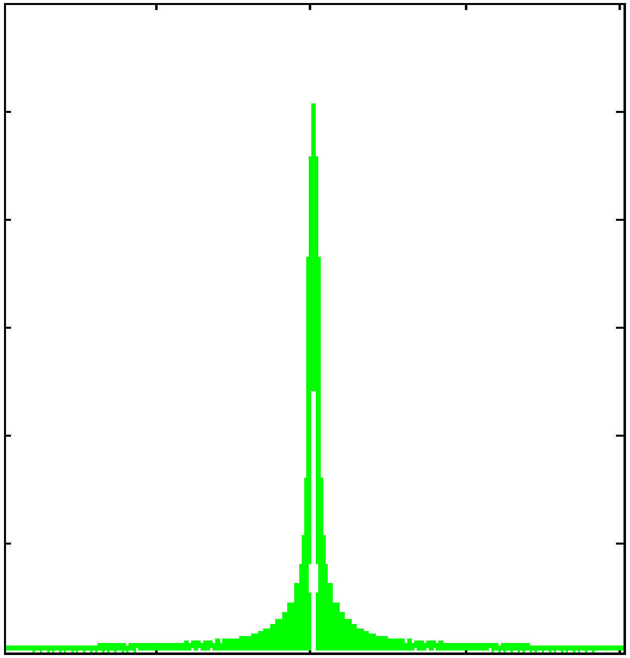
S = FFT(s)

/



P = FFT(p)

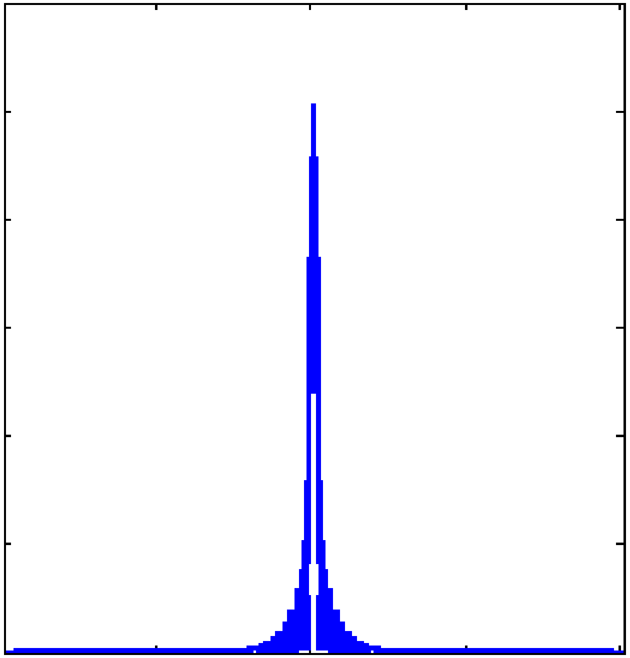
$\overset{!}{=}$



O

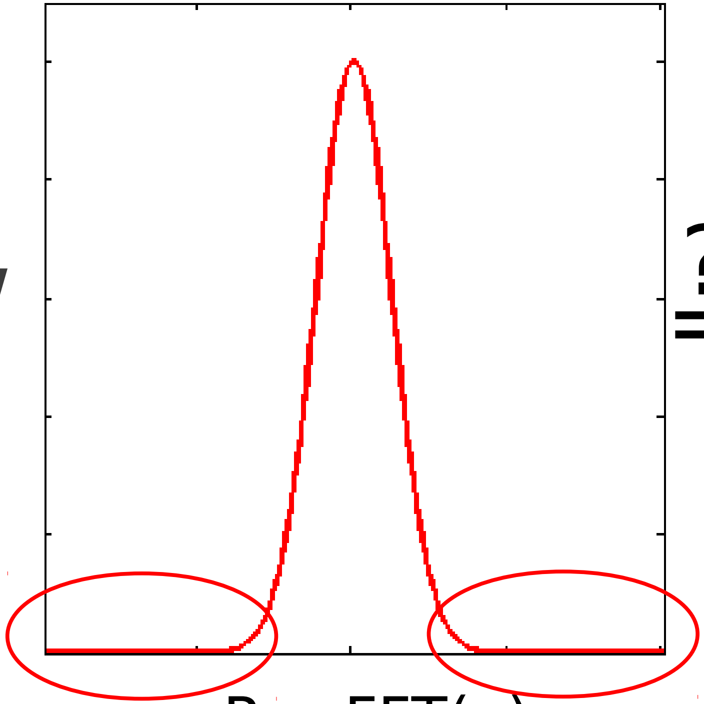
Image Restoration – Inverse Filter

- **Problem**: P_i is practically equal to zero in some parts of the spectrum
  - Lowpass filters (blur) induce $P_i = 0$ at high frequencies
  - Inversion is not feasible due to limited numerical accuracy
  - E.g. small inaccuracies in the FFT are strongly amplified



S = FFT(s)

/



P = FFT(p)

$\stackrel{?}{=}$

Image Restoration – Inverse Filter

- **Problem**: P_i is practically equal to zero in some parts of the spectrum
  - Lowpass filters (blur) induce $P_i = 0$ at high frequencies
  - Inversion is not feasible due to limited numerical accuracy
  - E.g. small inaccuracies in the FFT are strongly amplified
- **Solution**: Replace the inverse filter $1/P_i$ *by* Q_i

$$Q_i = \begin{cases} 1/P_i & |P_i| \geq \epsilon \, max_j(|P_j|) \\ \epsilon \, max_j(|P_j|) & |P_i| < \epsilon \, max_j(|P_j|) \end{cases}$$

$$\otimes \, IFFT(Q_i) =$$

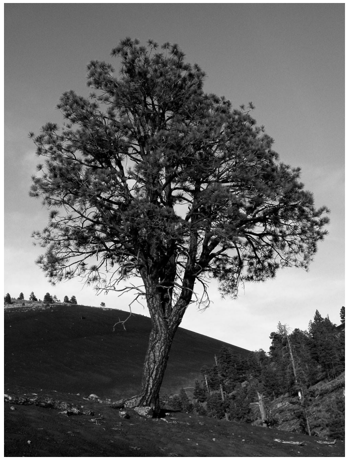
Image Restoration – Wiener Filter

Signal Model

$$s_i = \sum_j o_j \, p_{i-j} + n_i \quad \xrightarrow{\text{FFT}} \quad S_i = O_i \cdot P_i + N_i$$

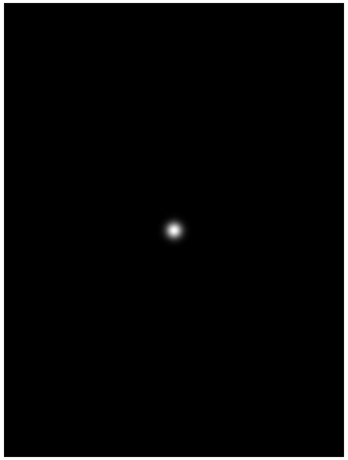
s_i: Observed signal	S_i: Spectrum of the observed signal
o_i: Original signal	O_i: Spectrum of the original signal
p_i: Impulse response	P_i: Spectrum of the impulse response
n_i: Noise	N_i: Noise spectrum

More realistic: real sensors are not perfect and numerical accuracy is limited



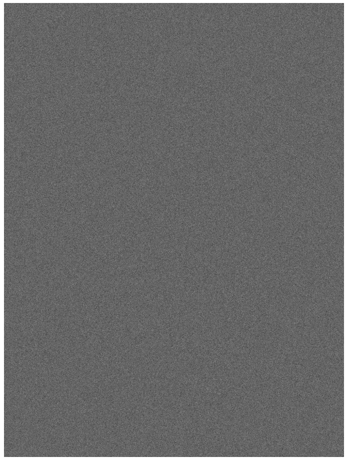
o_i

$\otimes$

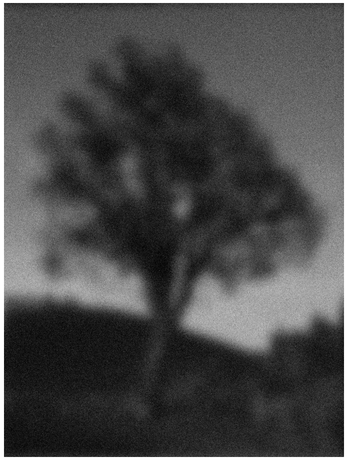


p_i

$+$



n_i



s_i

Image Restoration – Wiener Filter

- Find a filter q_i that is convolved with signal s_i to approximate the original o_i
  - Minimize the difference between o and $s \otimes q$

$$e = \sum (o - q \otimes s)^2 = \sum_i \left(o_i - \sum_j q_j s_{i-j} \right)^2 = min$$

$$d_k = \frac{\partial}{\partial q_k} e = 2 \sum_i s_{i-k} \left(o_i - \sum_j q_j s_{i-j} \right) = 2 (o - q \otimes s) \odot s = 0$$

$$D_k = 2 (O_k - Q_k S_k) S_k^* = 0$$

$$\boxed{Q_k = \frac{S_k^* O_k}{|S_k|^2} = \frac{P_k^* |O_k|^2 + N_k^* O_k}{|P_k|^2 |O_k|^2 + |N_k|^2 + P_k O_k N_k^* + P_k^* O_k^* N_k}}$$

(using $S_i = O_i \cdot P_i + N_i$)

Image Restoration – Wiener Filter

$$Q_k = \frac{P_k^* |O_k|^2 + N_k^* O_k}{|P_k|^2 |O_k|^2 + |N_k|^2 + P_k O_k N_k^* + P_k^* O_k^* N_k}$$

1. Signal o and noise n are not correlated

$$O_k^* N_k = 0 \quad \Rightarrow \quad Q_k = \frac{P_k^* |O_k|^2}{|P_k|^2 |O_k|^2 + |N_k|^2} = \frac{P_k^*}{|P_k|^2 + |N_k|^2 / |O_k|^2}$$

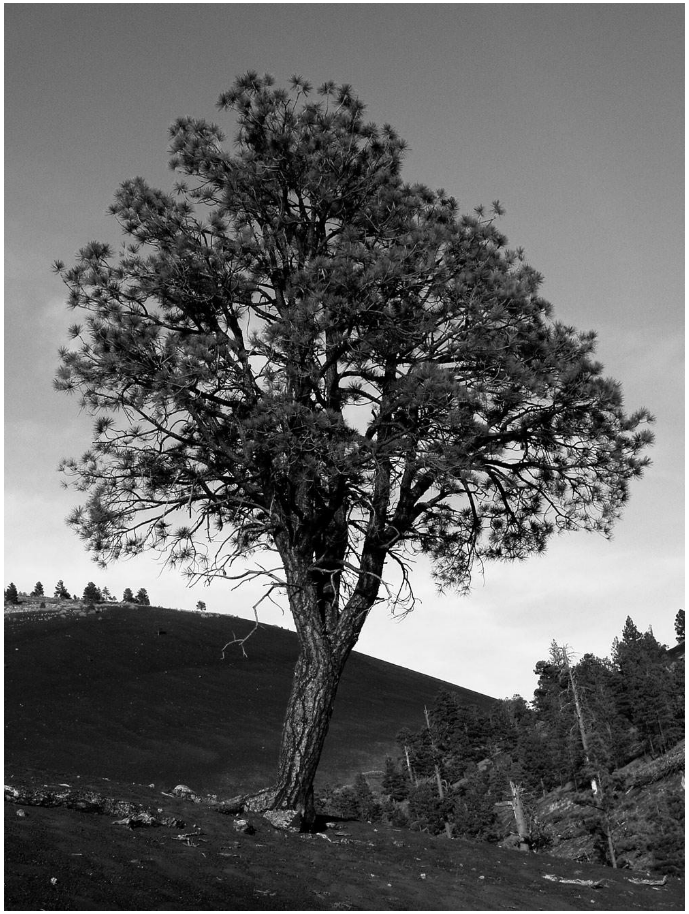
2. n and o are unknown

Signal to noise ratio $SNR = \left\langle \frac{|o|}{|n|} \right\rangle \quad \Rightarrow \quad \boxed{Q_k = \frac{P_k^*}{|P_k|^2 + 1/SNR^2}}$

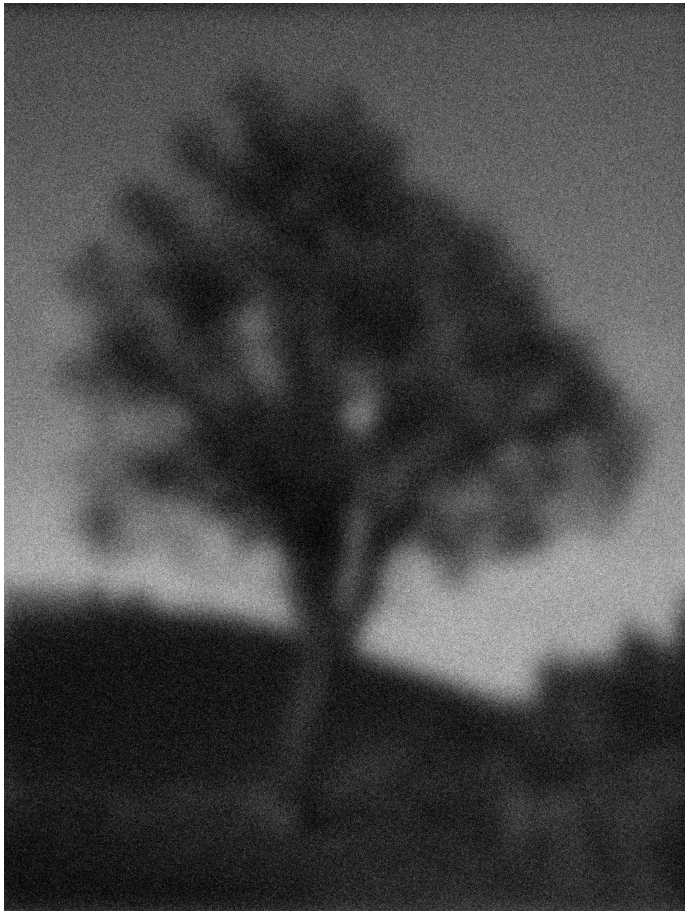
$$SNR = \infty \quad \Rightarrow \quad Q_k = \frac{P_k^*}{|P_k|^2 + 0} = \frac{1}{P_k}$$

(Inverse convolution!)

Image Restoration – Wiener Filter



Original o



Degraded s

$s = o \otimes p + n$

Restored

$s \otimes q$

5. Exercise - Given

`main(int argc, char** argv)`

- Loads image, path given in argv[1]
- Adds distortion (blur and noise)
- Calls restoration functions
- Saves restored images

`Mat degradeImage(Mat& img, Mat& degradedImg, double filterDev, double snr)`

`img`	input image
`degradedImage`	output image
`filterDev`	standard deviation of gaussian blur
`snr`	signal-to-noise ration
`return`	filter kernel used for blurring

- Adds gaussian blur and gaussian noise

5. Exercise – To Do

```
Mat inverseFilter(Mat& degraded, Mat& filter)
```

`degraded`	input image
`filter`	filter that caused distortion
`return`	restorated image

- Applies (modified) inverse filter to restore image (e.g. $\epsilon = 0.05$)

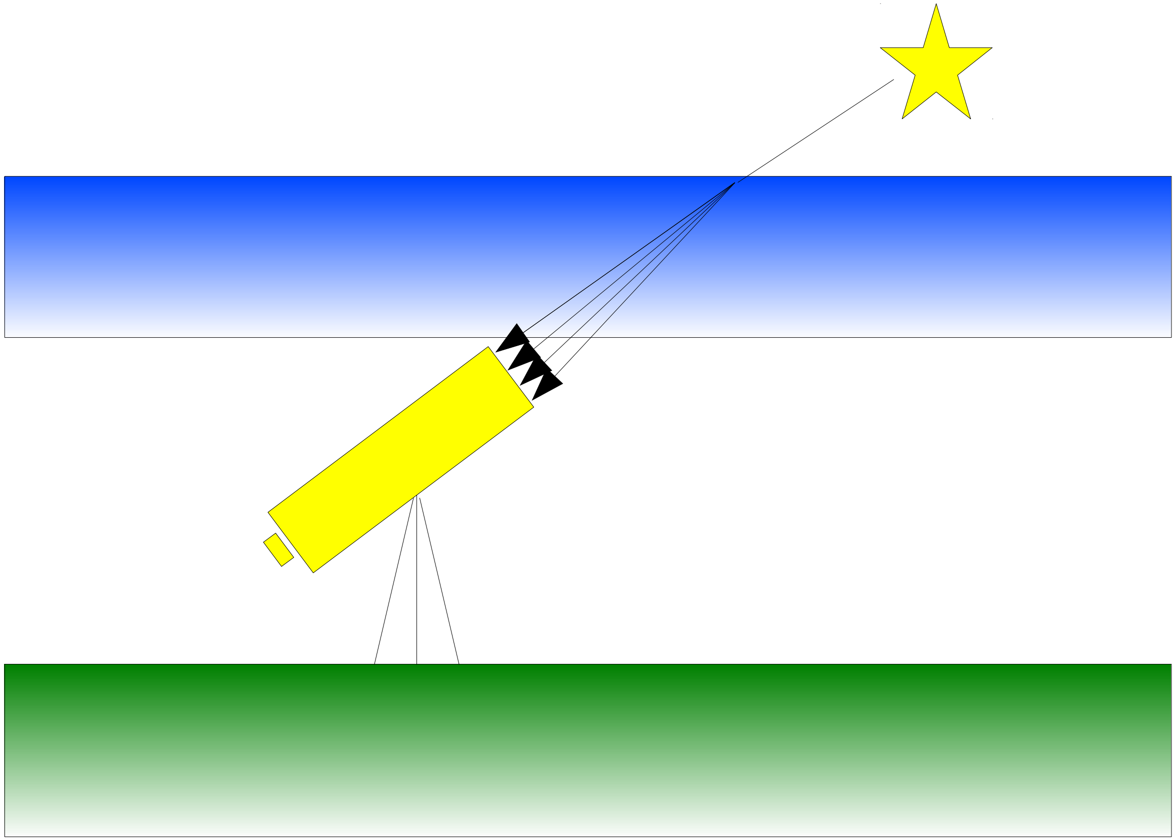
```
Mat wienerFilter(Mat& degraded, Mat& filter, double snr)
```

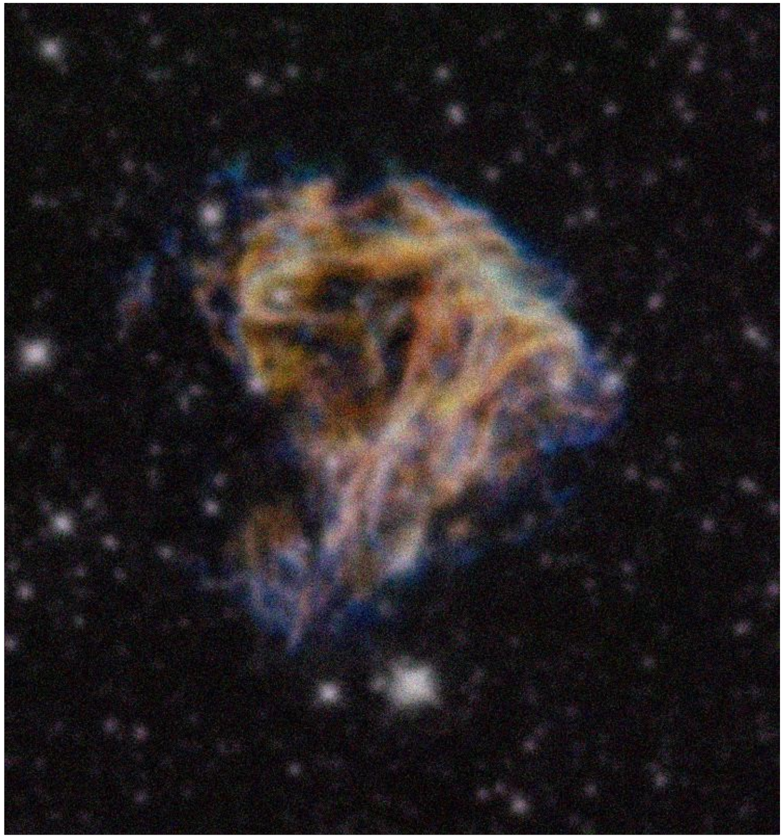
`degraded`	input image
`filter`	filter that caused distortion
`snr`	signal-to-noise ratio
`return`	restorated image

- Applies Wiener filter to restore image

Note:
- circShift(..)
- Proper usage of cv::dft(..), i.e. compressed output format
- Spectra are complex valued, i.e. 1/P is complex-valued!!
- Output image might contain large values. i.e. |out(x,y)| > 255
- Useful functions: cv::merge(..), cv::split(..), cv::threshold(..)

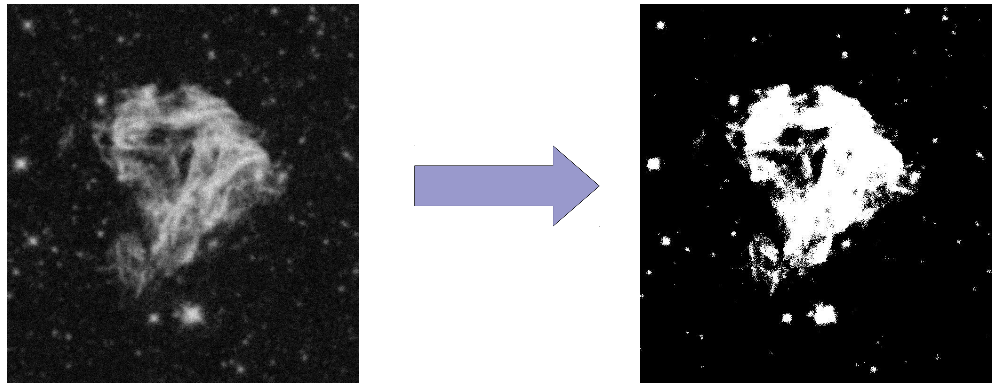
Image Restoration





- Acquisitions by earth-based telescopes are often severely degraded
  - Atmosphere: refraction and anisotropy cause distortions (blurring)
  - CCD (sensor): Extremely dim objects imply significant thermal noise
  - **Wiener Signal Model**: Convolution (Atmosphere) + Noise (CCD)
  - However: SNR and impulse response p are unknown

Image Restoration



S(x,y) T(x,y)

- Neither impulse response p_i nor SNR are known
- Variable with respect to atmospheric conditions

- **Automatic determination of the SNR**
  - Use threshold δ to separate foreground and background

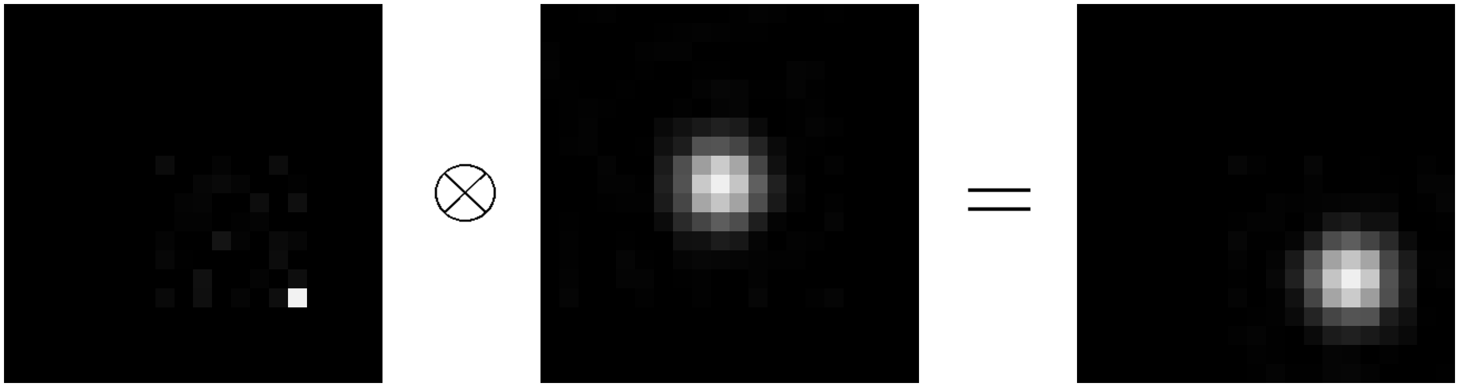
$$T(x,y) = \begin{cases} 1 & S(x,y) \geq \delta \\ 0 & S(x,y) < \delta \end{cases} \qquad \delta = 2\sigma\left[S(x,y)\right]$$

(s […]: standard deviation)

  - The ratio of mean foreground to mean background intensity is the SNR:

$$\mathrm{SNR} = \frac{\sum \mathrm{S}(x,y)\; T(x,y)}{\sum T(x,y)} \Big/ \frac{\sum \mathrm{S}(x,y)\,(1 - T(x,y))}{\sum 1 - T(x,y)}$$

Image Restoration



$$\delta_{a,b}(x,y) \;\otimes\; p(x,y) \;=\; c\,p(x-a, y-b)$$

$$\delta_{a,b}(x,y) \stackrel{\text{def}}{=} \begin{cases} c & (x,y) = (a,b) \\ 0 & (x,y) \neq (a,b) \end{cases}$$

- Convolving the impulse response with a delta function yields:
  - The original impulse response centred at the delta

- An image was convolved with an unknown kernel *p*
  - If the image contained a delta impulse, it will be replaced with *p*
  - The kernel can be established from the neighbourhood of the original delta!

Image Restoration

- Stars are (almost) delta impulses
- The neighborhood $N_S(x,y)$ of a star consists of p and thermal noise

$$N_S(x,y) \;=\; I\;p(x,y) \;+\; n(x,y) \quad -R \;\leq\; x,y \;\leq R$$

  - ➔ I: True intensity of the star

- Normalize intensity

$$N(x,y) \;=\; N_S(x,y)/\max(N_S(x,y)) \;=\; p(x,y) \;+\; n(x,y)/I$$

- Averaging the response around numerous stars reduces noise

$$M(x,y) \;=\; \langle N(x,y)\rangle \;=\; p(x,y) \;+\; r$$

- **Estimating *p***: Eliminate offset r and normalize intensity

$$p(x,y) \;\simeq\; \left(M(x,y) \;-\; \min(M(x,y))\right) / \sum \left(M(x,y) \;-\; \min(M(x,y))\right)$$

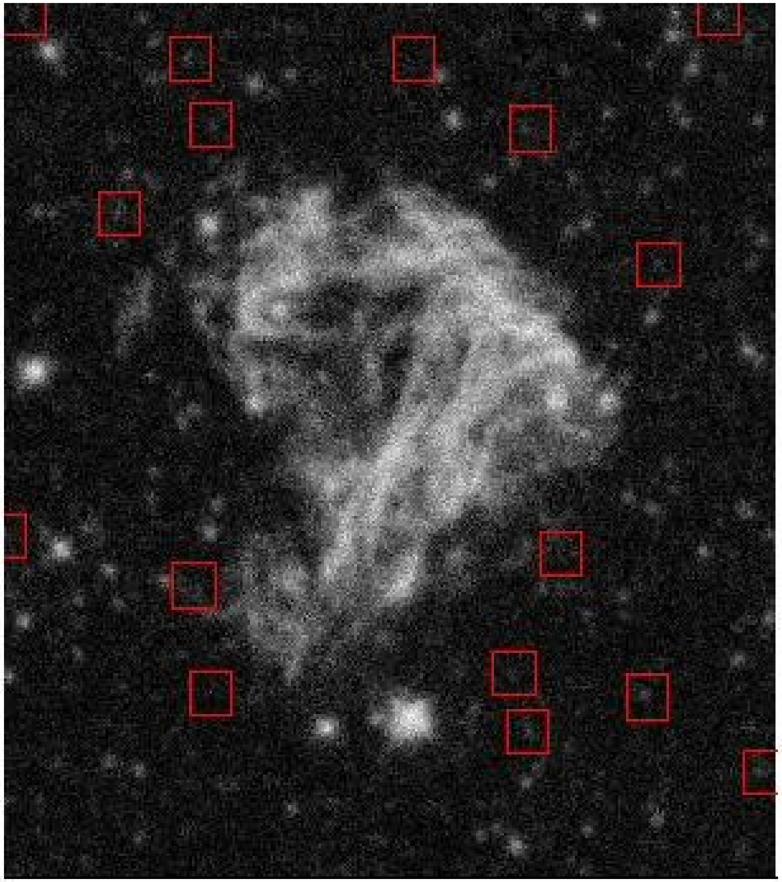
  - ➔ Normalize the kernel to unity (it should integrate to 1)

Image Restoration

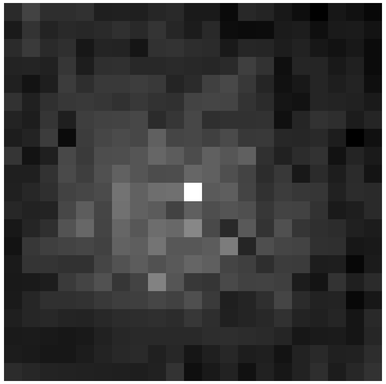
- Threshold: Separate foreground from background
- Enumerate foreground regions
- **For each region:**
  1. Determine indices of region
  2. Discard the region if it contains more than K pixels (galaxy or nebula?)
  3. Discard the region if there is another foreground structure in the vicinity (impulse responses overlap)
  4. Cut out the neighbourhood around the star (contains p)
  5. After scaling and averaging neighbourhoods, p is determined

- **Concerning Step 3**
  - Size of the neighbourhood: $2R+1$
  - Convolve T(x,y) with a mask of size $(2R+1)^2$ containing ones
  - This counts the number of foreground pixels in every neighbourhood in `T`
  - If the number of foreground pixels counted within a region exceeds the size of the region, there must be another star nearby

Image Restoration



Detected Stars



Impulse Response *p*

Image Restoration