भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

EE6310, Image and Video Processing

Linear Image Filtering

February 14, 2023

Linear Image Filtering

- Wraparound and Linear Convolution
- Linear Image Filters
- Linear Image Denoising
- Linear Image Restoration
- Filter Banks

Linear Image Filtering – Illusions

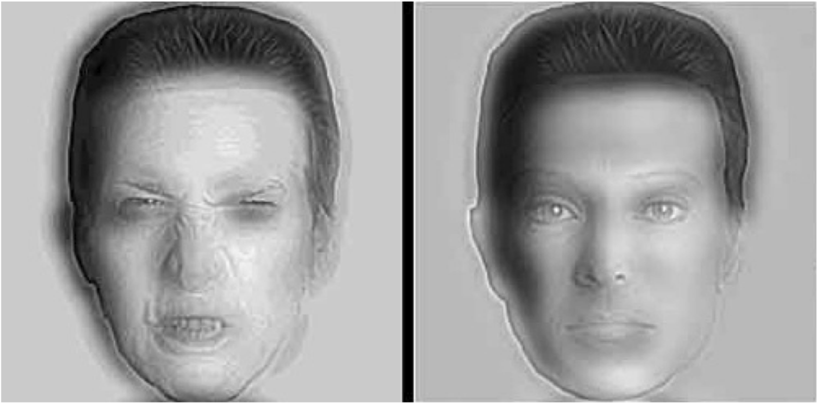

Figure: Who is angry?

Linear Image Filtering – Illusions

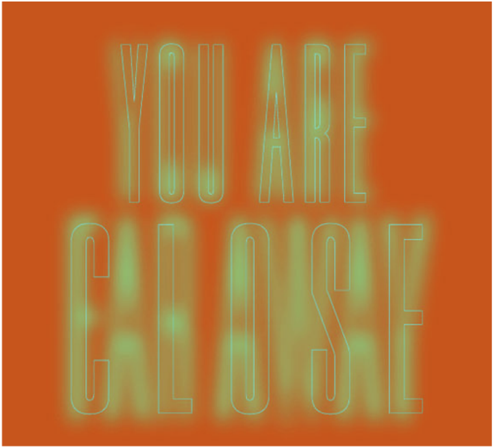

Figure: Close or far away?

Linear Image Filtering – Circular Convolution

- Modifying DFT of image changes its appearance - for e.g., applying a **zero-one mask**
- What happens if two image DFTs are **multiplied** pointwise? $\tilde{J} = \tilde{I}_1 \bigotimes \tilde{I}_2$
- A very important question whose answer has profound consequences in image processing
- The inverse DFT of $\tilde{J}$ is: $J(i,j) = \frac{1}{NM} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} \tilde{J}(u,v) W_N^{-ui} W_N^{-vj}$

Linear Image Filtering – Circular Convolution

- Replacing $\tilde{J}(u,v)$ with $\tilde{I}_1(u,v) \bigotimes \tilde{I}_2(u,v)$

$$J(i,j) = \frac{1}{NM}\sum_{u=0}^{N-1}\sum_{v=0}^{M-1} \tilde{I}_1(u,v) \bigotimes \tilde{I}_2(u,v) W_N^{-ui} W_N^{-vj}$$

$$= \frac{1}{NM}\sum_{u=0}^{N-1}\sum_{v=0}^{M-1}\{\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I_1(n,m) W_N^{un} W_M^{vm}\}\{\sum_{p=0}^{N-1}\sum_{q=0}^{M-1} I_2(p,q) W_N^{up} W_M^{vq}\} W_N^{-}$$

$$= \frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I_1(n,m) \sum_{p=0}^{N-1}\sum_{q=0}^{M-1} I_2(p,q) \sum_{u=0}^{N-1}\sum_{v=0}^{M-1} W_N^{u(n+p-i)} W_M^{v(m+q-j)}$$

$$= \frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I_1(n,m) \sum_{p=0}^{N-1}\sum_{q=0}^{M-1} I_2(p,q).NM.\delta(n+p-i, m+q-j)$$

$$= \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I_1(n,m) I_2[(i-n)_N, (j-m)_M]$$

$$= \sum_{p=0}^{N-1}\sum_{q=0}^{M-1} I_1[(i-p)_N, (j-q)_M] I_2(p,q) = I_1 \# I_2$$

Linear Image Filtering – Circular Convolution

- $p_N = p \bmod N$
- $I_1 \# I_2$ is the **circular convolution** of I_1 with I_2
- Like **linear convolution** except with indices taken **modulo N, M**

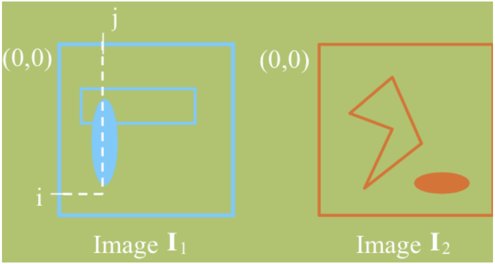

Figure: Depicting circular convolution

Linear Image Filtering – Circular Convolution

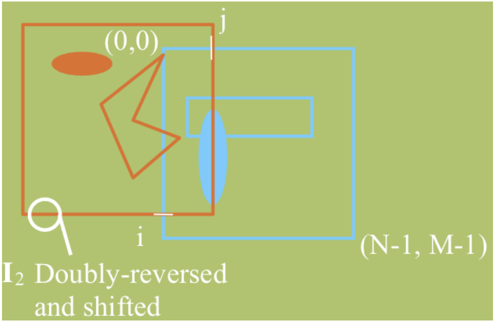

Figure: Without circular convolution

- Modulo arithmetic defines the product for all $0 \le i \le N-1; 0 \le j \le M-1$

Linear Image Filtering – Circular Convolution

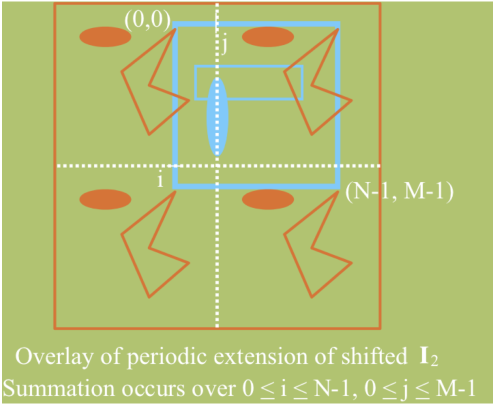

Figure: Cyclic convolution

- Overlay of periodic extension of I_2 for all $0 \leq i \leq N-1; 0 \leq j \leq M-1$

Linear Image Filtering – Computing Circular Convolution

- Direct computation of $I_1 \# I_2$ is simple but expensive
- For an $N \times M$ image:
  - For each of NM coordinates: NM multiplications and additions
  - For the entire image: $NM.NM$ multiplications
  - If $N = M = 512$, it turns out $2^{36} = 6.9 \times 10^{10}$
- An alternative is to use FFT to compute $\#$
- $J = I_1 \# I_2 = \mathrm{IFFT}_{N\times M}[\mathrm{FFT}_{N\times M}[I_1] \bigotimes \mathrm{FFT}_{N\times M}[I_2]]$
- Computing an $N \times M$ FFT is $O[NM.\log(NM)]$, and so is the computation of $\#$
- To make $\#$ useful, it must be modified

Linear Image Filtering – Linear Convolution

- Cyclic convolution is a consequence of the **DFT**, which is a sampled **DSFT**.
- If two **DSFTs** are multiplied together, then useful **linear convolution** results:
  $\tilde{J}_D = \tilde{I_{D1}}\tilde{I_{D2}} \implies J(i,j) = I_1(i,j) * I_2(i,j)$
- **Cyclic convolution** is an **artifact** of sampling the DSFT – which causes **spatial periodicity**
- Most of circuit theory, optics and analog filter theory based on **linear convolution**

Linear Image Filtering – Linear Convolution

- Linear digital filter theory also requires the concept of **linear convolution**
- It turns out that **circular convolution** can be used to **compute** linear convolution
- However, circular convolution has drawbacks and cannot be used as-is
- Consider the average filter: output is the average of pixel values in a square window

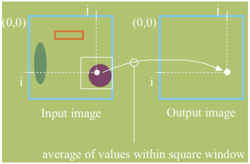

Figure: Averaging filter

Linear Image Filtering – Averaging Filter

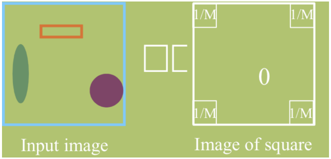

Figure: Averaging using circular convolution

The averaging filter may be expressed as the circular convolution of the image with the image of a square with intensity $1/M$ where M is the number of pixels in the square.

Linear Image Filtering – Averaging Filter

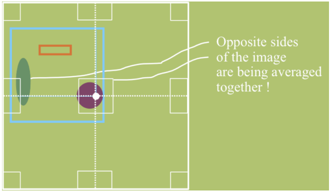

Figure: Undesirability of circular convolution

- Wraparound effects occur at image borders
- Desirable to average only neighboring elements and weight images based on **spatial ordering** rather than DFT-induced **periodic ordering**
- Large filters **worsen** effect

Linear Image Filtering – Linear Convolution by Zero Padding

- **Conceptually simple:** Pad images to be convolved with **zeros**
- **Typically**, both image arrays are doubled in size
- **Wraparound** eliminated since the "moving" image is weighted by zero outside image domain

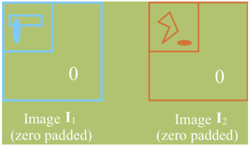

Figure: Zero padded images

Linear Image Filtering – Visualizing Wraparound Cancellation

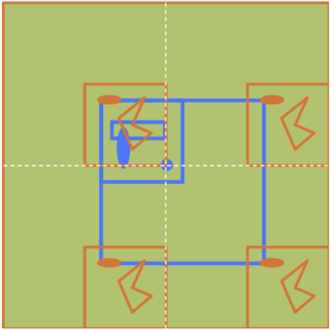

Figure: Wraparound cancellation

Remember, summation takes place within blue region
$(0 \leq i \leq 2N - 1; 0 \leq j \leq 2M - 1)$

Linear Image Filtering – DFT Computation of Linear Convolution

- Let J', I'_1, I'_2 be **zero-padded** $2N \times 2M$ versions of $J = I_1 * I_2$
- If $J' = I'_1 \# I'_2 = \text{IFFT}_{2N\times 2M}[\text{FFT}_{2N\times 2M}[I'_1]\text{FFT}_{2N\times 2M}[I'_2]]$, the $N \times M$ image with elements $J(i,j) = J'(i,i); 0 \leq i \leq N-1, 0 \leq j \leq M-1$ contains the **linear convolution** result
- Therefore, by multiplying **zero-padded** DFTs and taking the IFFT, we get $J = I_1 * I_2$
- The linear convolution is larger than $N \times M$ $(2N \times 2M)$ but the interesting part is contained in $N \times M$ region of J
- To convolve an $N \times M$ image with a $P \times Q$ filter:
  - If $P, Q < N, M$, pad filter with zeros to size $N \times M$
  - If $P, Q << N, M$, faster to convolve in space domain

Linear Image Filtering – Direct Linear Convolution

- Assume I_1 and I_2 are not periodically extended and assume $I_1(i,j) = I_2(i,j) = 0$ if $i < 0, J < 0$ and $i > N-1, j > M-1$
- In this case,
$$J(i,j) = I_1(i,j) * I_2(i,j) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} I_1(n,m) I_2(i-n, j-m)$$

Linear Image Filtering – Goals

- Linear filtering refers to a process that transforms an image I by linear convolution
- Goals include transforming an image:
  - Into one of **better quality**
  - With certain features **enhanced**
  - With certain features **de-emphasised** or **removed**
- Specifically:
  - **Smoothing:** remove noise due to bit errors, transmission etc
  - **Deblurring:** increase **sharpness** of blurred images
  - **Sharpening:** emphasize significant features like **edges**
  - **Combinations** of these

Linear Image Filtering – Characterizing Linear Filters

- Any **linear** filter can be characterized in one of two equivalent ways:
  - The **impulse response** $\mathbf{H} = H(i, j)$
  - The **frequency response** $\tilde{\mathbf{H}} = \tilde{H}(u, v)$
- $\tilde{\mathbf{H}} = \mathrm{DFT}[\mathbf{H}], \mathbf{H} = \mathrm{IDFT}[\tilde{\mathbf{H}}]$
- The **frequency response** describes how image frequencies are affected by the system
- Since $\tilde{H}(u, v) = |\tilde{H}(u, v)| \exp\{\sqrt{-1}\angle\tilde{H}(u, v)\}$, an image frequency component at $(u, v) = (a, b)$ is amplified or attenuated by $|\tilde{H}(a, b)|$ and shifted by $\angle\tilde{H}(a, b)$

Linear Image Filtering – Frequency Response Example

- If the **input** to a system H is a cosine image:
  $I(i,j) = \cos[2\pi(\frac{b}{N}i + \frac{c}{M}j)] = \frac{1}{2}\{W_N^{bi}W_M^{cj} + W_N^{-bi}W_M^{-cj}\}$
- The **output** is:
  $J(i,j) = I(i,j) * H(i,j) = |\tilde{H}(b,c)|\cos[2\pi(\frac{b}{N}i + \frac{c}{M}j) + \angle\tilde{H}(b,c)]$
- As with 1-D signals, the impulse response is an effective way to model responses since:
  $$I(i,j) = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I(n,m)\delta(i-n, j-m)$$

Linear Image Filtering – Linear Filter Design

- Given **analog** spec: $H_c(x, y) \overset{\mathfrak{F}}{\Longleftrightarrow} \tilde{H}_c(\Omega, \Lambda)$
- Two **simple** methods of filter design:
  - Space-sampled approximation:
    $(X = 1, Y = 1), H(i, j) = H_C(i, j); -\infty < i, j < \infty$
  - Frequency-sampled approximation using DFT:
    $\tilde{H}(u, v) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} \tilde{H}_c[(\frac{u}{N} - n), (\frac{v}{M} - m)]$

Linear Image Filtering – Space-Sampled Approximation

- Truncate the continuous response:
  $H_{trunc}(i,j) = H_C(i,j); 0 \leq |i| \leq (N/2)-1; 0 \leq |j| \leq (M/2)-1$
- Truncation of the analog spec leads to Gibbs phenomena at jump discontinuities of $\tilde{H}_C(\Omega, \Lambda)$
- DFT/IDFT pair $\tilde{H}_{trunc}(u,v) \overset{\text{IDFT}}{\Longleftrightarrow} H_{trunc}(i,j)$

Linear Image Filtering – Frequency Sampled Approximation

- $\tilde{H}_{fs}(u,v) = \tilde{H}_C(\frac{u}{N}, \frac{v}{M}); 0 \le |u| \le (N/2) - 1, 0 \le |v| \le (M/2) - 1$
- DSFT NOT specified between samples
- CFT is **centered** and **non-periodic**
- The discrete impulse response is then: $H_{fs}(i,j) \stackrel{\text{DFT}}{\Longleftrightarrow} \tilde{H}_{fs}(u,v)$

Linear Image Filtering – Low-Pass, Band-Pass and High-Pass Filters

- **Low-pass:** Attenuates all but "lower" frequencies
  - **smooth** noise
  - **blur** detail to highlight gross features
- **Band-pass:** Attenuates all but "middle" frequencies
  - **Special-purpose**
- **High-pass:** Attenuates all but "higher" frequencies
  - **enhance** image detail and contrast
  - **remove** image blur

Linear Image Filtering – Example Low-Pass Filter

- To smooth an image: replace each pixel in a noisy image by the **average** of its $M \times M$ neighbors
- How to pick window size M?
- Example: for a 512×512 image, **typical** window sizes range from 3×3, 5×5, . . ., 15×15 (lots of smoothing)

Linear Image Filtering – Example Low-Pass Filter

- Design an **ideal low-pass filter** in the DFT domain:

$$\tilde{H}(u,v) = \begin{cases} 1, & \text{if } \sqrt{u^2+v^2} \leq U_{cutoff} \\ 0, & \text{otherwise} \end{cases}$$

- Possibly useful if **radial** frequency U_{cutoff} can be estimated in the original image

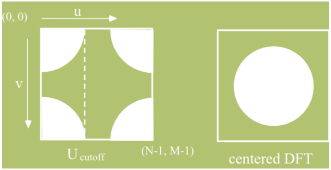

Figure: Ideal low-pass filter

Linear Image Filtering – Example Low-Pass Filter

- The **gaussian filter** with frequency response $\tilde{H}_C(\Omega, \Lambda) = \exp[-2(\pi\sigma)^2(\Omega^2 + \Lambda^2)]$, hence $\tilde{H}(u, v) = \exp\{-2\pi^2\sigma^2[(\frac{u}{N})^2 + (\frac{v}{M})^2]\}$
- Response falls quickly at high frequencies
- **Important** low-pass filter

Linear Image Filtering – Example Band-Pass Filter

- Define a BP filter as the **difference of two LPFs** that differ only by a scaling factor
- Common example is a **difference of gaussian (DOG)** filter: $\tilde{H}_C(\Omega, \Lambda) = \exp[-2(\pi\sigma)^2(\Omega^2+\Lambda^2)] - \exp[-2(K\pi\sigma)^2(\Omega^2+\Lambda^2)]$ hence $\tilde{H}(u, v) =$ $\exp\{-2\pi^2\sigma^2[(\frac{u}{N})^2 + (\frac{v}{M})^2]\} - \exp\{-2K^2\pi^2\sigma^2[(\frac{u}{N})^2 + (\frac{v}{M})^2]\}$
- Typically, $K \approx 1.5$
- DOG filters are very useful for image analysis and human visual modeling

Linear Image Filtering – Example High-Pass Filter

- **Laplacian** filter is important as well
- A **severely truncated** approximation is $\tilde{H}_C(\Omega, \Lambda) = A(\Omega^2 + \Lambda^2)$, hence $\tilde{H}(u, v) = A[(\frac{u}{N})^2 + (\frac{v}{M})^2]$
- An approximation to the Fourier transform of the **continuous Laplacian**: $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$

Linear Image Filtering – Linear Image Denoising

- **Linear image denoising** is a process of smoothing noise **without** destroying image information
- Noise modeled as **additive** or **multiplicative**
- Consider **additive** noise now
- **Multiplicative** noise better handled by **non-linear** filtering

Linear Image Filtering – Additive White Noise Model

- Model **additive white noise** as an image N with random elements
- Can be **thermal noise, channel noise, sensor noise** etc.
- Noise may affect the **continuous image** before sampling: $J_C(x, y) = I_C(x, y) + N(x, y)$

Linear Image Filtering – Additive White Noise Model

- Noise is **zero-mean** if limit of the average of P realizations of the noise image vanishes as $P \to \infty$:
  $\frac{1}{P} \sum_{p=1}^{P} N_{C,p}(x, y) \to 0$ for all (x, y) as $P \to \infty$
- **On average**, noise falls close to 0

Linear Image Filtering – Spectrum of White Noise

- The **noise energy spectrum** is $N_C(\Omega, \Lambda) = \mathfrak{F}\{N_c(x, y)\}$
- If **noise is white**, then, on average, the **energy spectrum will be flat** (or 'white'):
  $\frac{1}{P} \sum_{p=1}^{P} |N_{C,p}(\Omega, \Lambda)| \to \eta$ for all (Ω, Λ) as $P \to \infty$
- η^2 is called **noise power**

Linear Image Filtering – Linear Filtering

- **Objective: Remove** as much of the high frequency noise as possible while **preserving** as much image spectrum as possible
- Achieved by a LPF with large bandwidth (images are fairly wideband)

Linear Image Filtering – Digital White Noise

- Similar model for **digital** zero-mean additive white noise:
  $\underset{\text{observed}}{J} = \underset{\text{original}}{I} + \underset{\text{noise}}{N}$
- **On average**, elements of N will be zero
- DFT of noisy image is: $\underset{\text{observed}}{\tilde{J}} = \underset{\text{original}}{\tilde{I}} + \underset{\text{noise}}{\tilde{N}}$
- **On average**, the noise DFT will contain a **broad band** of frequencies

Linear Image Filtering – Denoising with Average Filter

- To smooth an image: replace each pixel in a noisy image by the **average** of its $M \times M$ neighbors
- **Rationale:** Averaging elements **reduces the noise mean towards zero**
- How to pick window size M? A **tradeoff** between noise smoothing and image smoothing
- Example: for a 512×512 image, **typical** window sizes range from 3×3, 5×5, . . ., 15×15 (lots of smoothing)
- **Linear filtering** the image with zero padding affects the image and noise spectra in the same way:
  $K = H * J = H * I + H * N \implies \tilde{K} = \tilde{H} \bigotimes \tilde{I} + \tilde{H} \bigotimes \tilde{N}$

Linear Image Filtering – Denoising with Ideal Low-Pass Filter

- Design an **ideal low-pass filter** in the DFT domain:
$$\tilde{H}(u,v) = \begin{cases} 1, & \text{if } \sqrt{u^2+v^2} \leq U_{cutoff} \\ 0, & \text{otherwise} \end{cases}$$
- Possibly useful if **radial** frequency U_{cutoff} can be estimated in the original image

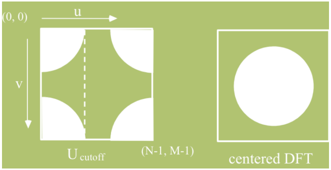

Figure: Ideal low-pass filter

Linear Image Filtering – Denoising with Gaussian Filter

- The **isotropic** Gaussian filter is an **effective** smoother: $\tilde{H}(u, v) = \exp[-2\pi^2\sigma^2(\frac{u^2+v^2}{N^2})]$
- More weight given to **"closer"** neighbors
- DFT design: set **half-peak** bandwidth to U_{cutoff} and solve for σ: $\exp[-2\pi^2\sigma^2(\frac{U_{cutoff}^2}{N^2})] = \frac{1}{2} \implies \sigma = \frac{N}{\pi U_{cutoff}}\sqrt{\log\sqrt{2}} \approx 0.19(\frac{N}{U_{cutoff}})$

Linear Image Filtering – Summary of Denoising Filters

- **Average filter:** Noise leakage through frequency ripple (spatial discontinuity)
- **Ideal low-pass filter:** Ringing from spatial ripple (frequency discontinuity)
- **Gaussian filter:** No ringing, no discontinuities and no leakage

Linear Image Filtering – Minimum Uncertainty

- Amongst **all real functions** and in any dimension, the no-ripple Gaussian functions **uniquely** minimize the uncertainty principle: $(\frac{\int |xf(x)|^2 dx}{\int |f(x)|^2 dx})(\frac{\int |uf(u)|^2 du}{\int |f(u)|^2 du}) \geq \frac{1}{4}$
- Similar for y, v
- **Minimal simultaneous space-time duration**

Linear Image Filtering – Linear Image Deblurring

- Often a digital image is corrupted by a linear process
- **Motion blur, defocusing etc.** causes of blurring
- Blurring can be modeled using **linear convolution**:
  $J_C(x, y) = G_C(x, y) * I_C(x, y)$
  $\implies \tilde{J}_C(\Omega, \Lambda) = \tilde{G}_C(\Omega, \Lambda)\tilde{I}_C(\Omega, \Lambda)$

Linear Image Filtering – Digital Blur Function

- The sample image will be of the form: $J = G * I$ and $\tilde{J} = \tilde{G} \bigotimes \tilde{I}$
- The distortion G is **almost always low-pass**
- Goal: Use digital filtering to reduce blur - a **very** hard problem!

Linear Image Filtering – Inverse Filter Deblurring

- Often possible to **estimate** G
- **Physics** of the situation helps:
  - **Motion blur** along **one direction**. Determining this direction can help with filter design
  - Modulation transfer function (MTF) of camera system can be determined and a suitable **digital deblurring** filter can be designed

Linear Image Filtering – Deconvolution

- Reversing the linear blur G is **deconvolution**. It is the **inverse filter** of the distortion: $\tilde{G}_{inverse}(u, v) = \frac{1}{\tilde{G}(u,v)}$ if $\tilde{G}(u, v) \neq 0$ for any (u, v)
- The restored image is then: $\tilde{K} = \tilde{G}_{inverse} \bigotimes \tilde{G} \bigotimes \tilde{I} = \tilde{I}$!

Linear Image Filtering – Blur Estimation

- An **estimate** of blur G might be obtainable
- The inverse of **low-pass** blur is **high-pass**
- Designer must be **careful** at high frequencies

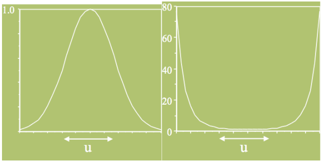

Figure: Inverting blurring filter

Linear Image Filtering – Blur Estimation

- Real world far from "ideal"
- Blur frequency response become **zero**
- Directly inverting $\tilde{G}(u, v)$ **meaningless** when $\tilde{G}(u, v) = 0$
- Frequencies **zeroed** by linear distortion are **irrecoverable**
- The best one can do is to reverse distortion at the **non-zero** values

Linear Image Filtering – Pseudo-Inverse Filter

- The **pseudo-inverse** filter is defined as:
  $$\tilde{G}_{p-inverse}(u,v) = \begin{cases} 1/\tilde{G}(u,v), & \text{if } \tilde{G}(u,v) \neq 0 \\ 0, & \text{otherwise} \end{cases}$$
- No attempt to recover lost frequencies
- Psuedo-inverse set to 0 in regions of missing frequencies
- Spurious noise frequencies **eradicated**

Linear Image Filtering – Blur and Additive Noise

- Things get worse when I is distorted by linear blur G and additive noise N: $J = G * I + N$
- An example is image sent over a noisy communication channel
- The DFT: $\tilde{J} = \tilde{G} \bigotimes \tilde{I} + \tilde{N}$

Linear Image Filtering – Failure of Inverse Filters

- Filtering with a linear filter H will produce:
  $K = H * J = H * G * I + H * N$ or
  $\tilde{K} = \tilde{H} \bigotimes \tilde{G} \bigotimes \tilde{I} + \tilde{H} \bigotimes \tilde{N}$
- We have a problem:
  - Low-pass filter smoothes over noise but doesn't remove blur
  - High-pass filter amplifies noise

Linear Image Filtering – Wiener Filter

- **Wiener filter** (after Norbert Wiener) is a minimum mean squared error (MMSE) filter
- For **blur** G and **white noise** N, $\tilde{G}_{Wiener}(u, v) = \frac{\tilde{G}(u,v)}{|\tilde{G}(u,v)|^2 + \eta^2}$
- Noise factor η usually not known - commonly estimated from flat image regions

Linear Image Filtering – Wiener Filter Rationale

- In case of no noise ($\eta = 0$), the filter reduces to $\tilde{G}_{Wiener}(u, v) = \frac{\tilde{G}(u,v)}{|\tilde{G}(u,v)|^2} = \frac{1}{\tilde{G}(u,v)}$
- This is the highly desirable **inverse filter**
- In case of no blur, the filter reduces to $\tilde{G}_{Wiener}(u, v) = \frac{1}{1+\eta^2}$
- This essentially scales the variance to minimize MSE

Linear Image Filtering – Pseudo-Wiener Filter

- To handle zeroed frequencies, define a **pseudo-Wiener** filter:
$$\tilde{G}_{Wiener}(u,v) = \begin{cases} \frac{\tilde{G}(u,v)}{|\tilde{G}(u,v)|^2+\eta^2}, & \text{if } \tilde{G}(u,v) \neq 0 \\ 0, & \text{otherwise} \end{cases}$$
- Noise in the zeroed frequency region **eradicated**