Home Assignment 2 – Blind Super Resolution

1. Write expressions for $l[\mathbf{n}]$ and $h[\mathbf{n}]$ as integrals over $f(\mathbf{x})$ and $p_L(\mathbf{x}), p_H(\mathbf{x})$ respectively.

The action of the camera expressed as $y(x) = (f * p)(x)$, where $f(x)$ is the continuous scene, $p(x)$ is the PSF of the camera and $y(x)$ is the resulting continuous image.

$h(x) = (f * p_H)(x)$ and $l(x) = (f * p_L)(x)$ correspond to capturing continuous high and low resolution images using two different PSF's, given $p_H(x)$ is narrower than $p_L(x)$.

The discrete counterparts, $l[n]$ and $h[n]$, are sampled on different lattices, $\mathbb{Z}^2$ and $\frac{1}{\alpha}\mathbb{Z}^2$ respectively:

$$l[n] = (f * p_L)(n) \overset{convolution}{\triangleq} \int_{x\in\mathbb{R}^2} f(x)p_L(n-x)dx\,, \forall\, n \in \mathbb{Z}^2$$

$$h[n] = (f * p_H)(n) \overset{convolution}{\triangleq} \int_{x\in\mathbb{R}^2} f(x)p_H(n-x)dx\,, \forall\, n \in \frac{1}{\alpha}\mathbb{Z}^2$$

$$= \int_{x\in\mathbb{R}^2} f(x)p_H\left(\frac{n}{\alpha} - x\right)dx\,, \forall\, n \in \mathbb{Z}^2$$

2. Write an expression for $l[\mathbf{n}]$ as a sum over $h[\mathbf{n}]$ and $k[\mathbf{n}]$.

Given, $l[n]$ can be retrieved from $h[n]$ by decimation $l[n] = \downarrow_\alpha (h * k)[n]$:

$$l[n] = \downarrow_\alpha (h*k)[n] \overset{\substack{discrete \\ convolution \\ + \\ h\ sampled\ on \\ lattice\ \frac{1}{\alpha}\mathbb{Z}^2}}{\triangleq} \downarrow_\alpha \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} h[m]k[n-m] \overset{\substack{sub \\ sampling}}{\triangleq} \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} h[m]k[\alpha n - m]$$

3. Write an equation relating $f(\mathbf{x})$, $p_H(\mathbf{x})$, $p_L(\mathbf{x})$ and $k[\mathbf{n}]$. Assume it holds for every possible function f, and derive an expression for $p_L[\mathbf{n}]$ as a sum over $p_H[\mathbf{n}]$ and $k[\mathbf{n}]$, all of which sampled on the lattice $\frac{1}{\alpha}\mathbb{Z}^2$.

By question 2, $l[n] = \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} h[m]k[\alpha n - m]\,, \forall n \in \mathbb{Z}^2 \implies l[n] = \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} h[m]k[n-m]\,, \forall n \in \frac{1}{\alpha}\mathbb{Z}^2$

By question 1: $l[n] = \int_{x\in\mathbb{R}^2} f(x)p_L(\alpha n - x)dx\,,\ \ h[n] = \int_{x\in\mathbb{R}^2} f(x)p_H(n-x)dx\,, \forall\, n \in \frac{1}{\alpha}\mathbb{Z}^2$

Therefore,

$$\int_{x\in\mathbb{R}^2} f(x)p_L(\alpha n - x)dx = \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} h[m]k[n-m]$$

$$\sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} h[m]k[n-m] = \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2}(\int_{x\in\mathbb{R}^2} f(x)p_H(m-x)dx)k[n-m]$$

$$= \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2}(\int_{x\in\mathbb{R}^2} f(x)p_H(m-x)dx)k[n-m] \overset{\substack{\textit{property of linearity} \\ \textit{of integration}}}{\triangleq} \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2}\left(\int_{x\in\mathbb{R}^2} f(x)p_H(m-x)k[n-m]dx\right)$$

$$= \int_{x\in\mathbb{R}^2} f(x)(\sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} p_H(m-x)k[n-m])\,dx$$

Therefore,

$$\int_{x\in\mathbb{R}^2} f(x)p_L(\alpha n-x)dx = \int_{x\in\mathbb{R}^2} f(x)(\sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} p_H(m-x)k[n-m])\,dx$$

Given it holds for every f , we can choose $f(x) = \delta(x)$:

$$p_L[\alpha n] = \sum_{m\in\frac{1}{\alpha}\mathbb{Z}^2} p_H[m]k[n-m] \overset{m=n-z}{\triangleq} \sum_{z\in\frac{1}{\alpha}\mathbb{Z}^2} p_H[n-z]k[z] \Rightarrow p_L[\alpha n] = (P_H * k)(n)$$

$$p_L[n] = \downarrow_\alpha (P_H * k)(n)$$

4. Assume $k_c(\mathbf{x})$ is a continuous-domain filter which yields $k[\mathbf{n}]$ when sampled on the lattice $\frac{1}{\alpha}\mathbb{Z}^2$. Further assume your expression for $p_L[\mathbf{n}]$ as a sum over $p_H[\mathbf{n}]$ and $k[\mathbf{n}]$ holds true in the continuous case as well, i.e. as an integral expressing $p_L(\mathbf{x})$ using $p_H(\mathbf{x})$ and $k_c(\mathbf{x})$. Use this to express $K_c(\boldsymbol{\xi})$ using $P_L(\boldsymbol{\xi})$, for $\boldsymbol{\xi}\in\mathbb{R}^2$ such that $P_H(\boldsymbol{\xi})\neq 0$.

---

by question 3, we assume that $p_L(x) = \downarrow_\alpha (p_H * k_c)(x)$

applying Fourier Transform: $P_L(\xi) = \frac{1}{\alpha}P_H(\xi)K_c(\xi)$

given $P_H(\xi)\neq 0$, $K_c(\xi) = \frac{\alpha P_L(\xi)}{P_H(\xi)}$

given $p_H(x) = \alpha^2 p_L(\alpha x)$, applying Fourier Transform (time scaling) : $P_H(\xi) = \alpha^2\cdot\frac{1}{\alpha}P_L\left(\frac{\xi}{\alpha}\right) = \alpha P_L\left(\frac{\xi}{\alpha}\right)$

therefore, $K_c(\xi) = \frac{\alpha P_L(\xi)}{P_H(\xi)} = \frac{\alpha P_L(\xi)}{\alpha P_L\left(\frac{\xi}{\alpha}\right)} = \frac{P_L(\xi)}{P_L\left(\frac{\xi}{\alpha}\right)}$

5. Most non-blind super resolution algorithms assume $k = p_L$. Does this assumption holds when $p_L = sinc$? What about when p_L is an isotropic Gaussian? Which PSF is more likely in real life? Base your answer on your estimated expression of $K_c(\boldsymbol{\xi})$.

If we assume that $p_L = sinc$, and we apply the Fourier Transform, we get $P_L(\xi) = Box(\xi)$.

Using the result from question 4:

$$K_c(\xi) = \frac{P_L(\xi)}{P_L\left(\frac{\xi}{\alpha}\right)} = \frac{Box(\xi)}{Box\left(\frac{\xi}{\alpha}\right)} = \frac{Box(\xi)}{1} = P_L(\xi)$$

therefore, $K_c(\xi) = P_L(\xi)$.

If we assume that p_L is a 2D isotropic Gaussian, $p_L(x) = \mathcal{N}(0, \sigma^2 I) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}x^T x}$, and

Applying Fourier Transform:

$$P_L(\xi) = \int_{-\infty}^{\infty} p_L(x) e^{-2\pi i \xi^T x} dx = \int_{-\infty}^{\infty} \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}x^T x} e^{-2\pi i \xi^T x} dx = \frac{1}{2\pi\sigma^2} \int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}x^T x - 2\pi i \xi^T x} dx$$

Next, we rewrite the exponent:

$$-\frac{1}{2\sigma^2}x^T x - 2\pi i \xi^T x = -\frac{1}{2\sigma^2}(x^T x + 4\pi i \sigma^2 \xi^T x) = -\frac{1}{2\sigma^2}(x^T x + 4\pi i \sigma^2 \xi^T x - 4\pi^2\sigma^4 \xi^T \xi + 4\pi^2\sigma^4\xi^T\xi) =$$

$$-\frac{1}{2\sigma^2}(x^T x + 4\pi i \sigma^2 \xi^T x + 4\pi^2\sigma^4\xi^T\xi - 4\pi^2\sigma^4\xi^T\xi) = -\frac{1}{2\sigma^2}(\| x + 2\pi i\, \sigma^2 \xi \|^2 - 4\pi^2\sigma^4\xi^T\xi)$$

Now, we can rewrite the integral with the modified exponent:

$$= \frac{1}{2\pi\sigma^2} \int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}(\|x + 2\pi i\, \sigma^2\xi\|^2 - 4\pi^2\sigma^4\xi^T\xi)} dx = \frac{1}{2\pi\sigma^2} \int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}(\|x+2\pi i\, \sigma^2 \xi\|^2)} e^{-2\pi^2\sigma^2\xi^T\xi} dx$$

$$= e^{-2\pi^2\sigma^2\xi^T\xi} \int_{-\infty}^{\infty} \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}(\|x+2\pi i\, \sigma^2\xi\|^2)} dx \overset{\substack{y = x + 2\pi i \sigma^2 \xi \\ dy = dx}}{\cong} e^{-2\pi^2\sigma^2\xi^T\xi} \int_{-\infty}^{\infty} \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}(\|y\|^2)} dy$$

We recognize that this is just the gaussian integral with y as the variable,

Thus, the integral evaluates to : $\int_{-\infty}^{\infty} \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2}(\|y\|^2)} dy = \frac{1}{2\pi\sigma^2}\sqrt{\frac{\pi}{\frac{1}{2\sigma^2}}} = \frac{1}{2\pi\sigma^2}\sqrt{2\pi\sigma^2} = \frac{1}{\sqrt{2\pi\sigma^2}}$

Therefore,

$$P_L(\xi) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-2\pi^2\sigma^2\xi^T\xi}$$

Using the result from question 4:

$$K_c(\xi) = \frac{P_L(\xi)}{P_L\left(\frac{\xi}{\alpha}\right)} = \frac{\frac{1}{\sqrt{2\pi\sigma^2}} e^{-2\pi^2\sigma^2\xi^T\xi}}{\frac{1}{\sqrt{2\pi\sigma^2}} e^{-2\pi^2\sigma^2 \frac{\xi^T}{\alpha}\frac{\xi}{\alpha}}} = \frac{e^{-2\pi^2\sigma^2\xi^T\xi}}{e^{-\frac{2\pi^2\sigma^2}{\alpha^2}\xi^T\xi}} = e^{-2\pi^2\sigma^2\xi^T\xi - \frac{2\pi^2\sigma^2}{\alpha^2}\xi^T\xi} = e^{\left(-2\pi^2\sigma^2 - \frac{2\pi^2\sigma^2}{\alpha^2}\right)\xi^T\xi}$$

$$K_c(\xi) = e^{-2\pi^2\sigma^2(1+\frac{1}{\alpha^2})\xi^T\xi} \neq P_L(\xi) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-2\pi^2\sigma^2\xi^T\xi}$$

We get that $K_c(\xi)$ also a gaussian.
But $K_c(\xi) \neq P_L(\xi)$

In real-life scenarios, Gaussian PSFs are more common due to their calculability and widespread applicability. Unlike Gaussian PSFs, sinc functions are challenging to calculate and use directly. Although the assumption $k = P_L$ holds for sinc PSFs, in practical terms, it's more feasible to work with Gaussian PSFs.

6. Assume we have access to a collection of clean high-resolution patches $\{p_1, \ldots, p_N\}$. Further assume every patch q_i in l is a noisy downsampled version of one of the high-resolution patches p_{j_i}, where j_i is uniformly distributed over $\{1, \ldots, N\}$. Hence $q_i = \downarrow_\alpha (p_{j_i} * k) + n_i$, where $n_i \sim \mathcal{N}\left(0, \sigma_N^2\right)$. Derive an objective function whose minimum argument corresponds to the ML estimation of k given l, i.e. maximizes $\mathbf{P}\left(l \mid k\right)$. You can assume that $\{p_i\}$, $\{n_i\}$ and k are statistically independent.

(1) We have a collection of clean-resolution patches denoted as $\{p_1, \ldots, p_N\}$.
(2) Each patch q_i in the low-resolution set is a noisy down sampled version of one of the high-resolution patches p_{j_i}, where j_i is uniformly distributed over $\{1, \ldots, N\}$, $P(j_i = j) = \frac{1}{N}$, this means that each high-resolution patch p_{j_i} has an equal probability of being selected from clean high-resolution patches $p_1, \ldots, p_N$, therefore $P(p_{j_i}) = \frac{1}{N}$.
(3) this relationship can be expressed as $q_i = \downarrow_\alpha (p_{j_i} * k) + n_i$, where n_i is the noise following the normal distribution, $n_i = q_i - \downarrow_\alpha (p_{j_i} * k)$
(4) $\{p_i\}, \{n_i\}, k$ are statically independent therefore, $\{q_i\}$ is statically independent.
(5) Adding a constant to a normally distributed random variable does not change its distribution.
(6) $P(q_i | p_{j_i} = p_j; k) = P(\downarrow_\alpha (p_j * k) + n_i | p_{j_i} = p_j; k) \overset{\substack{(3),(4) \\ +\downarrow_\alpha(p_{j_i} * k) = const \\ +(5)}}{\triangleq} \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}$

We want to maximize $P(l; k)$, let $\hat{k}$ to be the maximum likelihood estimate of the kernel k.

$$\hat{k} = argmax_k P(l; k)$$

$$P(\mathrm{l}; \mathrm{k}) = P(q_1, \ldots\ldots, q_N; k) \overset{(4)}{\triangleq} \prod_{i=1}^{N} P(q_i; \mathrm{k}) \overset{\substack{conditional \\ probability}}{\triangleq} \prod_{i=1}^{N} \left(\sum_{j=1}^{N} P(q_i | p_j ; k) \cdot P(p_j) \right) \overset{(2)}{\triangleq}$$

$$\prod_{i=1}^{N} \left(\frac{1}{N} \sum_{j=1}^{N} P(q_i | p_j ; k) \right) \overset{(6)}{\triangleq} \prod_{i=1}^{N} \left(\frac{1}{N} \sum_{j=1}^{N} \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}} \right)$$

We use logarithm of the likelihood function because it preserves the maximum value and transforms products into sums.

$$\hat{k} = argmax_k logP(l;k) \Longrightarrow logP(l;k) = \sum_{i=1}^{N} \log\left(\frac{1}{N}\sum_{j=1}^{N} \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right)$$

$$logP(l;k) = \sum_{i=1}^{N} \log\left(\frac{1}{N}\sum_{j=1}^{N} \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right)$$

$$= \sum_{i=1}^{N} \log\left(\frac{1}{N}\right) + \sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right) = const + \sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} const \cdot e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right)$$

We disregard constants since they do not influence the location of the minimum. Our objective is to derive an objective function whose minimum corresponds to maximizing the likelihood. Maximizing the logarithm of the likelihood is equivalent to minimizing the negative logarithm due to the behavior of log function.

$$\widehat{k_{ML}} = argmax_k logP(l;k) = argmin_k - logP(l;k) = argmin_k\left\{-\sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right)\right\}$$

7. Assume $\mathcal{D}k \sim \mathcal{N}\left(0, \sigma_D^2\right)$, where $\mathcal{D}$ is some operator. Suggest a reasonable choice for $\mathcal{D}$ and derive an objective function whose minimum argument corresponds to the MAP estimation of k given l, i.e. maximizes $\mathbf{P}(k \mid l)$. In your calculations, you can approximate $\mathbf{P}(k)$ as $\mathbf{P}(\mathcal{D}k)$.

We observed in the lecture that $\hat{f} = argmin_f L(y_1, \dots, y_N; f) - \log P(f)$, where $L(y_1, \dots, y_N; f)$ represents the likelihood function discussed in the previous question.

Therefore, we will leverage the developments made in the previous question:

$$\hat{k} = argmin_k = \{-logP(l;k) - \log P(k)\} \overset{given\ P(k)\approx P(Dk)}{\triangleq} argmin_k = \{-logP(l;k) - \log P(Dk)\}$$

$$= argmin_k\left\{-\sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right) - \log P(Dk)\right\}$$

$$\overset{given\ Dk \sim \mathcal{N}(0,\sigma_D^2)}{\triangleq} argmin_k\left\{-\sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right) - \log e^{-\frac{\|Dk\|^2}{2\sigma_D^2}}\right\}$$

$$= argmin_k\left\{-\sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right) + \frac{\| Dk \|^2}{2\sigma_D^2}\right\}$$

$$\hat{k} = argmin_k \frac{1}{2\sigma_D^2} \| Dk \|^2 - \sum_{i=1}^{N} \log\left(\sum_{j=1}^{N} e^{-\frac{\|q_i - \downarrow_\alpha(p_j * k)\|^2}{2\sigma_N^2}}\right)$$

When D is operator , choosing the appropriate operator depends on the specific characteristics of the kernel k but we want smoothness in the kernel k, so we choose D to be an operator that penalizes sharp changes or discontinuities in k, such as a spatial gradient or Laplacian operator.

8. Write the derivative of the previous function and suggest an iterative algorithm to estimate k.

First, let's compute the derivative of the previous function:

$$\frac{\partial}{\partial k}\left(\frac{1}{2\sigma_D^2}\parallel Dk\parallel^2-\sum_{i=1}^{N}\log\left(\sum_{j=1}^{N}e^{-\frac{\|q_i-\downarrow_\alpha(p_j*k)\|^2}{2\sigma_N^2}}\right)\right)=\frac{1}{2\sigma_D^2}\left(\frac{\partial}{\partial k}\parallel Dk\parallel^2\right)-\sum_{i=1}^{N}\frac{\partial}{\partial k}\log\left(\sum_{j=1}^{N}e^{-\frac{\|q_i-\downarrow_\alpha(p_j*k)\|^2}{2\sigma_N^2}}\right)$$

$$=\frac{1}{2\sigma_D^2}\cdot 2D^TDk-\sum_{i=1}^{N}\frac{\frac{\partial}{\partial k}\left(\sum_{j=1}^{N}e^{-\frac{\|q_i-\downarrow_\alpha(p_j*k)\|^2}{2\sigma_N^2}}\right)}{\sum_{j=1}^{N}e^{-\frac{\|q_i-\downarrow_\alpha(p_j*k)\|^2}{2\sigma_N^2}}}$$

$$=\frac{1}{2\sigma_D^2}\cdot 2D^TDk-\sum_{i=1}^{N}\frac{\sum_{j=1}^{N}\left(e^{-\frac{\|q_i-\downarrow_\alpha(p_j*k)\|^2}{2\sigma_N^2}}\frac{\partial}{\partial k}\left(-\frac{\parallel q_i-\downarrow_\alpha\left(p_j*k\right)\parallel^2}{2\sigma_N^2}\right)\right)}{\sum_{j=1}^{N}e^{-\frac{\|q_i-\downarrow_\alpha(p_j*k)\|^2}{2\sigma_N^2}}}$$

$(*)$ *let* R_j *be a matrix corresponding to convolution of* p_j *with* k *and subsampling by* α *so,* $R_jk=\downarrow_\alpha\left(p_j*k\right)$

$$(**)\ \frac{\partial}{\partial k}\left(-\frac{\parallel q_i-\downarrow_\alpha(p_j*k)\parallel^2}{2\sigma_N^2}\right)\overset{(*)}{\cong}\frac{\partial}{\partial k}\left(-\frac{\parallel q_i-R_jk\parallel^2}{2\sigma_N^2}\right)=\frac{R_j^T(q_i-R_jk)}{\sigma_N^2}$$

$$\overset{(**)}{\cong}\frac{1}{2\sigma_D^2}\cdot 2D^TDk-\sum_{i=1}^{N}\frac{\sum_{j=1}^{N}\left(e^{-\frac{\|q_i-R_jk\|^2}{2\sigma_N^2}}\cdot\frac{R_j^T(q_i-R_jk)}{\sigma_N^2}\right)}{\sum_{j=1}^{N}e^{-\frac{\|q_i-R_jk\|^2}{2\sigma_N^2}}}$$

$$(***)let\ w_{ij}=\frac{e^{-\frac{\|q_i-R_jk\|^2}{2\sigma_N^2}}}{\sum_{j=1}^{N}e^{-\frac{\|q_i-R_jk\|^2}{2\sigma_N^2}}}$$

$$\overset{(***)}{\cong}\frac{D^TDk}{\sigma_D^2}-\sum_{i=1}^{N}\sum_{j=1}^{N}w_{ij}\cdot\frac{R_j^T(q_i-R_jk)}{\sigma_N^2}$$

Setting the derivative to zero to find k:

$$\frac{D^T Dk}{\sigma_D^2} - \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} \cdot \frac{R_j^T(q_i - R_j k)}{\sigma_N^2} = 0$$

$$\frac{D^T Dk}{\sigma_D^2} = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{w_{ij}}{\sigma_N^2} \cdot R_j^T q_i - R_j^T R_j k\right) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{w_{ij}}{\sigma_N^2} \cdot R_j^T q_i\right) - \sum_{i=1}^{N}\sum_{j=1}^{N}\left(R_j^T R_j k\right)$$

$$k\left(\frac{D^T D}{\sigma_D^2} + \sum_{i=1}^{N}\sum_{j=1}^{N} R_j^T R_j\right) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{w_{ij}}{\sigma_N^2} \cdot R_j^T q_i\right)$$

$$\Longrightarrow \hat{k} = \left(\frac{D^T D}{\sigma_D^2} + \sum_{i=1}^{N}\sum_{j=1}^{N} R_j^T R_j\right)^{-1} \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{w_{ij}}{\sigma_N^2} \cdot R_j^T q_i\right)$$

Minimizing can be accomplished iteratively. In each step, we down-sample the low-resolution patches $\{p_1, \dots., p_N\}$ using the current estimate $\hat{k}$ of k, identify nearest neighbors to the patches $\{q_i\}$, and update $\hat{k}$ by solving a weighted least-squares problem. We assign non-negligible weights only to those patches whose distance to q is small.

solving a weighted least-squares problem by setting the gradient of the objective from Q7 to zero, leads to a version of the iteratively re-weighted least squares (IRLS) procedure.

An iterative algorithm to estimate k :
Input : query low-res patches $\{q_1, \dots, q_N\}$ and example high resolution patches $\{p_1, \dots., p_N\}$
Output : The MAP_k estimate $\hat{k}$.

- initialize $\hat{k}$ with gaussian (an initial guess $\hat{k}$ to a delta function).
- for T=1,..,T do
    1. Down sampling the training patches $\{p_1, \dots., p_N\}$ using the current estimate $\hat{k}$ of k :
       $r_j^\alpha = R_j k = \downarrow_\alpha (p_j * k)$
    2. Find NNS and compute weights: $w_{ij} = \frac{e^{-\frac{\|q_i - r_j^\alpha\|^2}{2\sigma_N^2}}}{\sum_{j=1}^{N} e^{-\frac{\|q_i - r_j^\alpha\|^2}{2\sigma_N^2}}}$
    3. update $\hat{k} = \left(\frac{D^T D}{\sigma_D^2} + \sum_{i=1}^{N}\sum_{j=1}^{N} R_j^T R_j\right)^{-1} \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{w_{ij}}{\sigma_N^2} \cdot R_j^T q_i\right)$

(*) the experience in the article shows that this process converges to correct k vert quickly, typically 15 iterations, so we use T =15

9. Let us look at a zoomed-in version of our scene $f\left(\frac{\mathbf{x}}{\alpha}\right)$, and assume it is being captured using $p_L(\mathbf{x})$ on the lattice $\mathbb{Z}^2$. Write an expression describing the resulting image $z[\mathbf{n}]$ as an integral over $f\left(\frac{\mathbf{x}}{\alpha}\right)$ and $p_H(\mathbf{x})$.

$$z[n] = (f\left(\frac{x}{\alpha}\right) * p_L(x)[n] = \int_{x\in\mathbb{R}^2} f\left(\frac{x}{\alpha}\right) p_L(n-x)dx$$

Given $p_L(\alpha x) = \frac{1}{\alpha^2} p_H(x)$

$$z[n] = \int_{x\in\mathbb{R}^2} f\left(\frac{x}{\alpha}\right)\frac{1}{\alpha^2} p_H\left(\frac{n-x}{\alpha}\right)dx \overset{\substack{z=\frac{x}{\alpha} \\ dz\cdot\alpha^2=dx}}{\triangleq} \int_{x\in\mathbb{R}^2} f(z)\frac{1}{\alpha^2} p_H\left(\frac{n}{\alpha}-z\right)dz\cdot\alpha^2$$

$$= \int_{z\in\mathbb{R}^2} f(z)p_H\left(\frac{n}{\alpha}-z\right)dz = \int_{z\in\mathbb{R}^2} f(z)p_H\left(\frac{n}{\alpha}-z\right)dz = (f*p_H)\left(\frac{n}{\alpha}\right) =\downarrow_\alpha (f*p_H)(n)$$

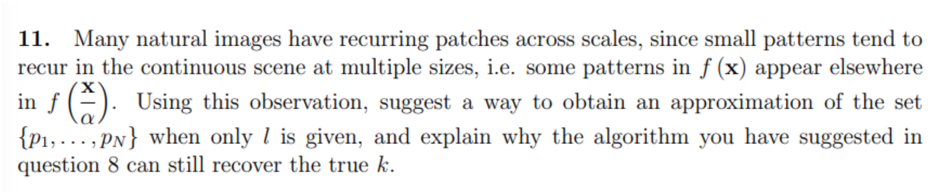
10. Prove $l[\mathbf{n}] =\downarrow_\alpha (z*k)[\mathbf{n}]$.

$h[n] = (f*p_H)(n) = \int_{x\in\mathbb{R}^2} f(x)p_H\left(\frac{n}{\alpha}-x\right)dx\ ,\forall\ n\in\mathbb{Z}^2$

by q9 : $z[n] = \int_{z\in\mathbb{R}^2} f(z)p_H\left(\frac{n}{\alpha}-z\right)dz\ ,\forall\ n\in\mathbb{Z}^2$

therefore, $h[n] = z[n]$.

given $l[n] =\downarrow_\alpha (h*k)[n]$, we get that $l[n] =\downarrow_\alpha (z*k)[n]$



11. Many natural images have recurring patches across scales, since small patterns tend to recur in the continuous scene at multiple sizes, i.e. some patterns in $f(\mathbf{x})$ appear elsewhere in $f\left(\frac{\mathbf{x}}{\alpha}\right)$. Using this observation, suggest a way to obtain an approximation of the set $\{p_1, \dots, p_N\}$ when only l is given, and explain why the algorithm you have suggested in question 8 can still recover the true k.

The observation that small image patches recur across scales of an image, implies that small patterns recur in the continuous scene at multiple sizes.
Let $f(x)$ be such a pattern, recurring elsewhere in the continuous scene as $f\left(\frac{x}{\alpha}\right)$.
These two continuous patterns are observed in the low-res image by two discrete patterns of different sizes, contained in patches q and r.
Patches q and r are generated by the low-res PSF p_L as:

$$q[n] = \int f(x) p_L(n-x)dx$$

$$r[n] = \int f\left(\frac{x}{\alpha}\right) p_L(n-x)dx$$

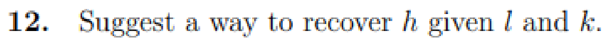
With a change of integration variables,

$$r[n] \overset{\substack{y=\frac{x}{\alpha} \\ dy\alpha^2=dx}}{\hat{=}} \int f(y)\alpha^2 p_L(n-\alpha y)dy \overset{p_H(x)=\alpha^2 p_L(\alpha x)}{\hat{=}} \int f(y) p_H\left(\frac{n}{\alpha}-y\right)dy = h[n]$$

Therefore, when only l is given, we can:

1. Sample patches from the image l.
2. Zoom in these patches .
3. Identify similar patches across different scales, leveraging the recurring nature of patches in natural images.
4. Utilize these patches as input to the algorithm described in Question 8 to approximate the kernel k.

We show in q10 (and this question too) Using zoom-in image is equal to using the high-resolution image, Therefore, The algorithm proposed in Question 8 remains effective in recovering the true k.
It will still recover the true k because we are using recurring patches and is it given that many natural images have recurring patches across scales, since small patterns tend to recur in the continuous scene.



12. Suggest a way to recover h given l and k.

To recover the high-resolution kernel h given the low-resolution image l and kernel k, we can use Maximum Likelihood Estimation (MLE).
This involves maximizing the likelihood of observing l given h and k, expressed as $argmax_h\{P(h\,|l,k\}\}$,
By defining the negative log-likelihood function and utilizing optimization algorithms, we aim to find the value of h that minimizes the negative log-likelihood, thereby recovering the high-resolution kernel.
Another option, we can define a loss function to quantify the difference between the observed low-resolution image pixels and the down sampled convolution of the estimated high-resolution image pixels with the blur kernel. One common choice for the loss function is the Mean Squared Error (MSE), expressed as $MSE(l, \downarrow_\alpha (h * k))$,
With machine learning algorithm we can find the best h.