Home Assignment 2 – Blind Super Resolution

In the first homework we dealt with correcting the motion blur occurring in images acquired through burst mode, where the motion of the camera is faster than the shutter speed of the lens. In this exercise, we will look at another instance of image blur called the *out-of-focus* blur, which is observed when the camera's focus is not properly adjusted by the user.

The point spread function (*PSF*) describes the distribution of light in the camera focal plane for a point source. This function strictly relates to the resolution and blur of the optical device. In the absence of any imaging noise, the action of the camera can be mathematically expressed as $y(\mathbf{x}) = (f * p)(\mathbf{x})$, where $f(\mathbf{x})$ is the continuous scene, $p(\mathbf{x})$ is the PSF of the camera and $y(\mathbf{x})$ is the resulting continuous image.

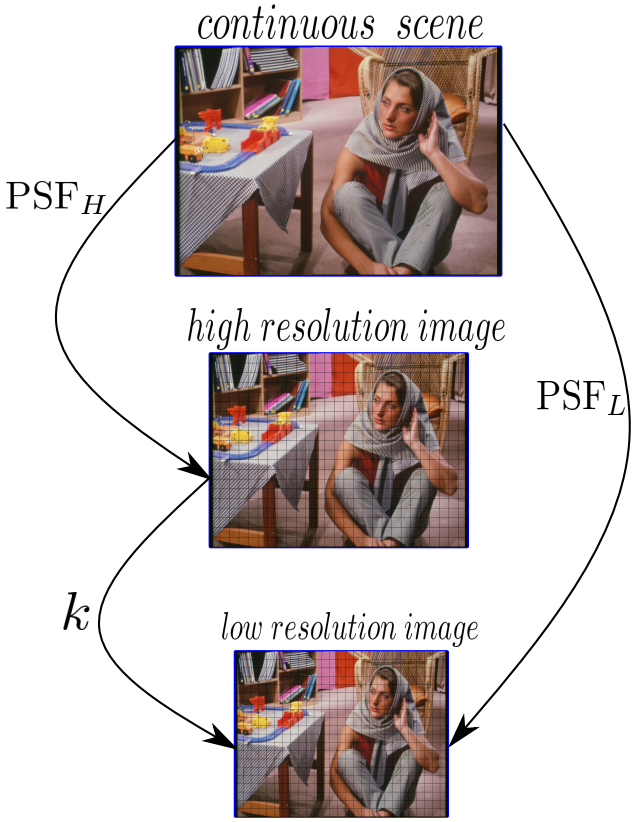
Super resolution is one of the cornerstone problems in image processing because it exemplifies the scenario of faithfully reconstructing a high-resolution image from an image captured out-of-focus. This can be modeled similarly to the above equation, where $h(\mathbf{x}) = (f * p_H)(\mathbf{x})$ and $l(\mathbf{x}) = (f * p_L)(\mathbf{x})$ correspond to capturing continuous high and low resolution images using two different PSFs, given $p_H(\mathbf{x})$ is narrower than $p_L(\mathbf{x})$ by $\alpha \in \mathbb{N}$, i.e. $p_H(\mathbf{x}) = \alpha p_L(\alpha \mathbf{x})$. In practice, since $l(\mathbf{x})$ contains less details than $h(\mathbf{x})$, their discrete counterparts, $l[\mathbf{n}]$ and $h[\mathbf{n}]$, are sampled on different lattices, $\mathbb{Z}^2$ and $\frac{1}{\alpha}\mathbb{Z}^2$ respectively, resulting with different resolutions. Thus we will also assume $l[\mathbf{n}]$ can be retrieved from $h[\mathbf{n}]$ by decimation $l[\mathbf{n}] = \downarrow_\alpha (h * k)[\mathbf{n}]$, where k is a discrete low-pass filter (see the figure in the next page for an illustration).

Usually super resolution algorithms try to induce $h[\mathbf{n}]$ given $l[\mathbf{n}]$ and $k[\mathbf{n}]$. In this assignment, however, we will solve the blind super resolution problem, where $k[\mathbf{n}]$ is unknown as well. Thus, we will first try to deduce $k[\mathbf{n}]$ from $l[\mathbf{n}]$, and then use our estimation of $k[\mathbf{n}]$ to retrieve $h[\mathbf{n}]$.

1 Theoretical questions

In the following questions you may ignore aliasing, i.e. assume every function g sampled on $\mathcal{L}^*$ holds $G[\boldsymbol{\omega}] = G(\boldsymbol{\omega}), \forall \boldsymbol{\omega} \in \mathcal{L}_0$.

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.


continuous scene
PSF$_H$
high resolution image
PSF$_L$
k
low resolution image

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

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 $p_L(\mathbf{x})$ using $p_H(\mathbf{x})$ and $k[\mathbf{n}]$.

4. Consider $\boldsymbol{\omega} \in \left[-\frac{\alpha}{2}, \frac{\alpha}{2}\right]$ for which $P_H[\boldsymbol{\omega}] \neq 0$. Write an expression for $K[\boldsymbol{\omega}]$ using $P_L[\boldsymbol{\omega}]$.

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 $K[\boldsymbol{\omega}]$.

From now on we will assume h, p_l, p_h and k and are unknown.

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 $n_i \sim \mathcal{N}(0, \sigma_N)$. Derive an objective function whose minimum argument corresponds to the ML estimation of k given l, i.e. maximizes $\mathbf{P}(l \mid k)$. You can assume that $\{p_i\}$, $\{n_i\}$ and k are statistically independent.

7. Assume $\mathcal{D}k \sim \mathcal{N}(0, \sigma_D)$, 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 $\mathbf{P}(k \mid l)$.

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

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 $p_H(\mathbf{x})$.

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

11. Many natural images have recurring patches across scales, since small patterns tend to recur in the continuous scene at multiple sizes. Using this observation, suggest a way to obtain an approximation of the set $\{p_1, \ldots, p_N\}$ when only l is given, and explain why the algorithm you have suggested in question 8 can still recover the true k.

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

2 Practical questions

To do this exercise, please download the source image DIPSourceHW2.png from here. This image will be considered as our high-resolution image.

- Create two low-resolution versions of the source image, once using a *sinc* filter and once using a Gaussian filter. You may choose any $\alpha \in \mathbb{N}$.
- Estimate the filters from the downsampled images using the algorithm you have designed. It is advisable to obtain the largest possible collection of high-resolution patches $\{p_i\}$, while adopting some of the implementation tricks from NLM in order to improve complexity.
- Construct high-resolution images using the low-resolution images and the kernels you have recovered.
- Construct high-resolution images using the wrong kernels.
- Evaluate the PSNR for each of the high-resolution images you have obtained.

3 Deliverables

- **For Part I:** Write the answers for questions 1-12 in a digital file.
- **For Part II:**
  - The code
  - 2 low-resolution images
  - 4 high-resolution images (2 obtained using the correct kernels and 2 using the wrong ones)
  - 4 PSNR values for each of the high-resolution images
  - A brief explanation discussing your results

Submit all the above mentioned deliverables by **18th January, 11:55PM**.

Submission will soon be available here.

Thanks and enjoy!
Alex and Omer