國立清華大學

碩士論文

貝氏推論在影像去模糊化上的應用

Blind Image Deconvolution via Bayesian Inference

所別： 數學系數學組

學號： 108021513

姓名： 馮偉哲 (Wei-Jhe Fong)

學號： 108021513

指導教授： 吳金典 博士 (Chin-Tien Wu, Ph.D.)

蔡志強 博士 (Je-Chiang Tsai, Ph.D.)

中華民國 111 年 01 月

摘要

在影像清晰的條件底下，我們能從其中獲得大量資訊。但時常由於硬體的設定或人為地因素，使得拍下照片變得模糊。由於無法清楚得知模糊的原因，因此在重建影像時需要同時估計模糊核。在只有一張模糊照片的情況下，同時重建影像及模糊核是個不適定性問題。雖然已經有許多演算法可以重建出相對清晰的影像，但這些方法裡時常忽略或是不清楚初始模糊核的估計。此篇論文探討利用 MAP_k 的方法以廣義的狄拉克核為初始核來估計一個相對較接近真實的模糊核。我們會詳細推導其中的理論，並且應用於真實世界的影像上。

關鍵詞: 點擴散函數、盲反迴旋捲積、非盲反迴旋捲積、貝氏推論、最大事後機率估計、最大期望算法

Abstract

Blind deconvolution is the recovery of a sharp version of a blurred image when the blur kernel is unknown. This is an ill-posed problem since both true kernel with latent image and delta kernel with blur image can minimize the energy function. Therefore a good initial kernel that helps avoiding blur image with delta kernel for deblur process plays an essential role. Many research have already developed reasonable algorithms, but among them the initial blur kernel required for the algorithms are often unclear. This paper look into the MAP_k algorithm based on Bayesian Inference, while the initial kernel of the process is always a delta kernel. The goal of this paper is to analyze and evaluate the blind image deconvolution method-MAP_k both theoretically and experimentally. We derive the method mathematically and implement it on a real world images.

Keywords: image convolution, blind image deconvolution, Bayesian inference, Maximum a posteriori estimation, Expectation-maximization algorithm

Contents

List of Figures

1 Background

1.1 Image Model

In our experience in taking images, there are great chances that one would accidentally shake his or her hands a little bit causing the image become fuzzy. Another common situation is that we often take the images without correct focus resulting unpleasant image quality. These are just two examples that makes the images "blurry". Many photographs capture transient moments that cannot be recaptured under controlled conditions or repeated with different camera settings if blur occurs in the image for any reason, then that moment is "lost". Hence recovering the blur images become an important task.

Let $y(i,j)$ be the blurry image that we observe where i,j are the index of pixels. Generally the blurry image with noise can be mathematically modeled as:

$$y = x \otimes k + n \tag{1.1}$$

where x is the sharp latent image, k denotes either a point spread function or a blur kernel that makes the image blurry, and n denote the additive noise. Usually the noise n is assumed to be an gaussian with zero mean and small variance. The symbol $\otimes$ denote the 2D convolution between the image x and kernel k. It can be written as:

$$\begin{aligned} y(i,j) &= (x \otimes k)(i,j) + n(i,j) \\ &= \sum_{l_1=-\frac{m-1}{2}}^{\frac{m-1}{2}} \sum_{l_2=-\frac{t-1}{2}}^{\frac{t-1}{2}} x(i+l_1, j+l_2) k(\frac{m+1}{2} + l_1, \frac{t+1}{2} + l_2) + n(i,j) \end{aligned} \tag{1.2}$$

where i,j are the pixel index in the image y and m,t are the index of kernel k.

One may notice that there are some boundary issues in the convolution part of equation (1.2) if the kernel size is beyond the pixel located in image. As the result, there are four methods which are ignoring border, padding the boundary with zeros, extending with reflection and extending circularly that deal this problem. Each method has its pros and cons, for example, extending the boundary circularly allows us to perform the convolution as multiplication in frequency domain but the boundary may become more blurry since the pixels on one boundary

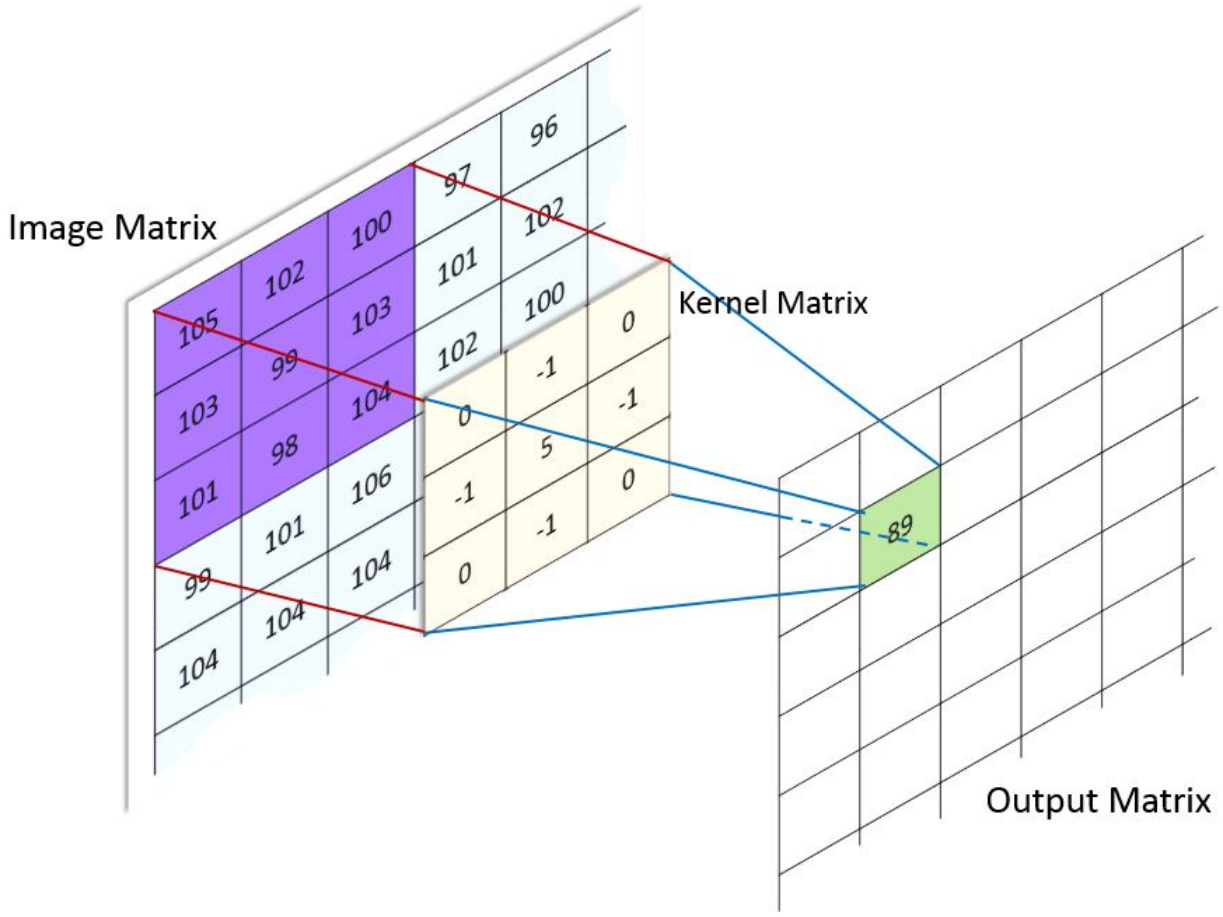


Figure 1: illustration of image convolution

would be affect by opposite boundary which make the noise to increase. This phenomenon may be take into consideration in our deblur process, that is when deblurring an image we tend to take out the boundary part, since there may be some noise that comes from the calculation of convolution rather than real noise.

Focus on the kernel, also known as Point Spread Function (PSF), it describes the focal length of the lens and the dynamic relation between the subject and the camera. There are two common types of PSF in image system. The first one is the Gaussian kernel which spreads a Gaussian distribution from center point and making the image blurry like the one with wrong focus length. The next common seen PSF is motion blur which is simply a line or curve that blur the image like the one we take photo with shaking hands. The sum of kernel will always be one in our experiment. It is difficult to reconstruct the kernel with only the blurred image. In this paper we will review some of classic methods about reconstructing the kernel and latent image with only blurred image as input which is called "blind deconvolution". And we will perform

some experiments to deblur the image with either motion, gaussian or even ring types kernels.

In order to characterize the experimental results in numbers, we introduce two common measurement of image quality that are often adopted in signal process - Signal to Noise Ratio (SNR) and Peak Signal to Noise Ratio (PSNR). SNR is a measure used in science and engineering that compares the level of a desired signal to the level of background noise.

SNR is defined as the ratio of signal power to the noise power, often expressed in decibels. The definition of SNR is:

$$\text{SNR} = 10 \log_{10} \left(\frac{\|y\|^2}{\|y - x\|^2} \right) \text{ db} \tag{1.3}$$

Here the latent and blur images x, y are flatten into vector.

On the other hand, the definition of PSNR is:

$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2 N}{\|y - x\|^2} \right) \text{ db} \tag{1.4}$$

where N is the number of pixels in the image. It is believed that the result is good enough when PSNR of the image is greater than 30 db.

1.2 Probability Assumptions

The task of blind deconvolution is to retrieve the latent image x with only the blur image y as input. It can be almost impossible to solve the problem without any assumption. There are two main assumptions and both of them come in probability sense. To speak of probability, we must clarify the definition of random variable and the sample space. In the rest of this paper, we always assume that the sample space $\mathbb{S}$ is each individual pixel in the images, and the random variable is always a function that maps a pixel to the real number space $\mathbb{R}$. When we say "the probability of an image", it refer to the product of probability at each pixels in that image, since we assume each pixel has a random variable, and they are independent and identically distributed with each other.

The first assumption according to is the noise of an image is gaussian, that is the probability $p(y|x, k)$ can be describe as:

$$p(y|x, k) = \frac{1}{(\sqrt{2\pi}\eta)^N} \exp^{-\frac{1}{2\eta^2}\|x \otimes k - y\|^2} \tag{1.5}$$

Where N is the number of pixels and η is the deviation of the gaussain distribution. Note that the probability is of entire noise image, one can see that it is the product of probability in each pixel that follows an independent and identically gaussian distribution. The intuition arise during acquisition of photo. The sensor of camera has inherent noise due to the level of illumination and its own temperature, and the electronic circuits connected to the sensor inject their own share of electronic circuit noise. The second assumption is that the gradient histogram of natural image follows a mixture of J gaussian distribution:

$$p(x) = \prod_{i=1}^{N} p(x(i)) \tag{1.6}$$

$$p(x(i)) = \prod_{l=1}^{2} \left(\sum_{j=1}^{J} \frac{\pi_j}{\sqrt{2\pi}\sigma_j} \exp^{-\frac{1}{2\sigma_j^2}|(g_l \otimes x)(i)|^2} \right) \tag{1.7}$$

where g_1, g_2 representing the horizontal and vertical derivative respectively, here we simply use $\{g_1, g_2\} = \{[-1, 1], [-1, 1]^T\}$. The intuition comes from the work of [4], their research shows that although images of real-world scenes vary greatly in their absolute color distributions, they obey heavy-tail distributions in their gradient: the distribution of gradients has most of its mass on small values but gives significantly more probability to large values. This correspond to the phenomenon that image often contain large section of constant or smooth intensity area with occasionally interrupted by large change at the edge between different sections. For example,

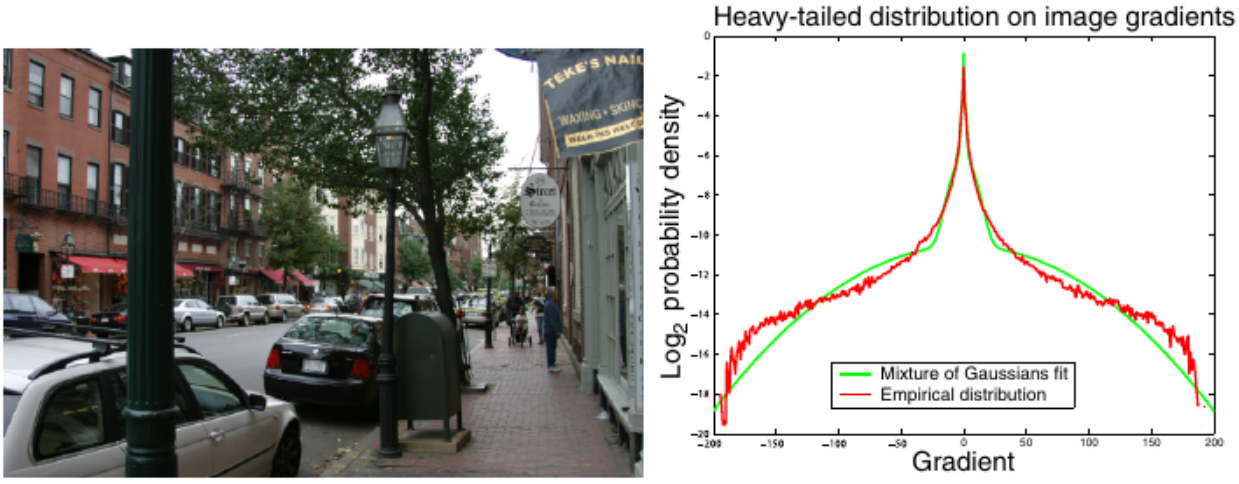


Figure 2: (a) A natural image. (b) The distribution of horizontal and vertical gradients value are shown in red. The y-axis has a logarithmic scale to show the heavy tails of the distribution. The mixture of Gaussians approximation used in our experiments is shown in green [3].

figure 2 shows a natural image and its gradient histogram, we can see that most of the gradient

are with smalls absolute values but also a few gradient have large values than gaussian distribution. Many researches base on heavy tail distributions give state-of -art results in image denoise and super resolution In our paper, we use the zero mean mixture of gaussian which is sparse to characterize the heavy tail property of image gradients. We perform two blind deconvolution algorithms that correspond to 1 component MOG which is just a zero mean gaussian distribution and 4 components MOG distribution respectively. Although the gaussain distribution may not well describe the heavy-tail property but it still give surprisingly great result. But overall the sparse prior give the best result at all.

In equation (1.5), we assume that the noise existed in spatial domain, but as shown in , performing deconvolution in gradient domain maybe a better choice since the prior in (1.6) is in gradient domain. Hence we may also assume:

$$p(\nabla y|\nabla x, k) = \frac{1}{(\sqrt{2\pi}\eta)^N} \exp^{-\frac{1}{2\eta^2}\|\nabla x \otimes k - \nabla y\|^2} \tag{1.8}$$

the corresponding image model to (1.1) and prior (1.7) are

$$\nabla y = \nabla x \otimes k + n \tag{1.9}$$

$$p(\nabla x) = \prod_{i=1}^{N}\prod_{l=1}^{2}\left(\sum_{j=1}^{J}\frac{\pi_j}{\sqrt{2\pi}\sigma_j}\exp^{-\frac{1}{2\sigma_j^2}|(g_l \otimes x)(i)|^2}\right) \tag{1.10}$$

Last but not the least, the kernel k in (1.1) is often sparse. That is we typically assume a sparse prior $p(k)$ on the kernel k. The sparsity may be characterize be L0-norm

$$p(k) = \exp^{-\beta\sum_i |k(i)|^0} \tag{1.11}$$

where $\lambda \in \mathbb{R}$. Note that we also assume the kernel always sum to 1 that is $\sum_{i=1}^{M} k(i) = 1$, where M is the number of pixels in the kernel.

2 Fitting the Image Prior

In this section, we present the algorithm that fits the MOG distribution in (1.7) from a number of natural images. The main algorithm is called the Expectation-Maximization Algorithm which is introduced by Dempster, Laird and Rubin [2].

2.1 The Expectation-Maximization Algorithm

Assuming the complete data consists of $\mathcal{Z} = (\mathcal{X}, \mathcal{H})$ but only the data $\mathcal{X}$ is observed. In our case, the observed data $\mathcal{X} = \{x_1, \cdots, x_n\}$ are the pixel-wise horizontal and vertical gradient values of a number of natural images. The goal of EM algorithm is to compute the Maximum Likelihood Estimate (MLE) of unknown parameter vector $\Theta_j = (\pi_j, \sigma_j),\ j = 1, 2, \cdots, J$ given the observed data $\mathcal{X}$. The hidden data $\mathcal{H} = \{h_1, \cdots, h_n\}$ is a vector consist of n random variable s.t $h_i \in \{1, 2, \cdots, J\}$ and $p(h_i = j) = \pi_j$ (i.e $p(h_i) = \pi_{h_i}$), $\forall i \in \{1, 2, \cdots, n\}\ \forall j \in \{1, 2, \cdots, J\}$. The realized value of h_i determines which of the J normal distributions generates the corresponding value of x_i. The reason why we include the "hidden" variable and calculate the MLE for $p(x, h; \Theta)$ instead of $p(x; \Theta)$ is because the MLE of $p(x; \Theta)$ doesn't has a closed form and may cause singularity problem[1]. Note that the relation of $p(x, h)$ and $p(x)$ are:

$$\begin{aligned} p(x, h; \Theta) &= p(x|h; \Theta)p(h; \Theta) \\ &= \frac{1}{\sqrt{2\pi}\sigma_h} \exp^{-\frac{1}{2\sigma_h^2}|x|^2} \pi_h \end{aligned} \tag{2.1}$$

and

$$\begin{aligned} p(x) &= \sum_{j=1}^{J} p(x, h = j; \Theta) \\ &= \sum_{j=1}^{J} \frac{\pi_j}{\sqrt{2\pi}\sigma_j} \exp^{-\frac{1}{2\sigma_j^2}|x|^2} \end{aligned} \tag{2.2}$$

where $p(x)$ is exactly the density function of the mixture of gaussian we want to fit. The EM algorithm is done iteratively, where each iteration consists of two steps:

- **E-step:** The E-step of the EM algorithm computes the expected value of log likelihood of $\mathcal{L}(\mathcal{X}, \mathcal{H}; \Theta)$ w.r.t the random vector $\mathcal{H}$ given the observed data $\mathcal{X}$, and the current

parameter estimate Θ_{old}. In particular, we define

$$\begin{aligned}
\mathcal{L}(\mathcal{X},\mathcal{H};\Theta) :&= \log\left(\prod_{i=1}^{n} p(x_i,h_i;\Theta)\right)\\
&= \log\left(\prod_{i=1}^{n} p(x_i|h_i;\Theta)p(h_i;\Theta)\right)\\
&= \log\left(\prod_{i=1}^{n} \frac{\pi_{h_i}}{\sqrt{2\pi}\sigma_{h_i}} \exp^{-\frac{1}{2\sigma_{h_i}^2}|x_i|^2}\right)\\
&= \sum_{i=1}^{n} \log\left(\frac{\pi_{h_i}}{\sqrt{2\pi}\sigma_{h_i}} \exp^{-\frac{1}{2\sigma_{h_i}^2}|x_i|^2}\right) \qquad (2.3)
\end{aligned}$$

and

$$\begin{aligned}
p(h_i;\mathcal{X},\Theta_{old}) &= \frac{p(h_i,x_i;\Theta_{old})}{p(x_i;\Theta_{old})}\\
&= \frac{p(h_i,x_i;\Theta_{old})}{\sum_{j=1}^{J} p(x_i,h=j;\Theta_{old})} \qquad (2.4)\\
&= \frac{\frac{\pi_{h_{i,old}}}{\sqrt{2\pi}\sigma_{h_i,old}} \exp^{-\frac{1}{2\sigma_{h_i,old}^2}|x_i|^2}}{\sum_{j=1}^{J} \frac{\pi_{j,old}}{\sqrt{2\pi}\sigma_{j,old}} \exp^{-\frac{1}{2\sigma_{j,old}^2}|x_i|^2}} \qquad (2.5)
\end{aligned}$$

the last equation (2.5) can be derive from (2.1) and (2.2).

By (2.3) and take the expected value of $\mathcal{L}(\mathcal{X},\mathcal{H};\Theta)$ w.r.t random vector $\mathcal{H}$ we have

$$\begin{aligned}
Q(\Theta;\Theta_{old}) &:= E[\mathcal{L}(\mathcal{X},\mathcal{H};\Theta) \mid \mathcal{X},\Theta_{old}]\\
&= \int \mathcal{L}(\mathcal{X},H;\Theta)p(H;\mathcal{X},\Theta_{old})dH \qquad (2.6)\\
&= \sum_{h_1=1}^{J}\cdots\sum_{h_n=1}^{J}\sum_{i=1}^{n} \log\left(\frac{\pi_{h_i}}{\sqrt{2\pi}\sigma_{h_i}} \exp^{-\frac{1}{2\sigma_{h_i}^2}|x_i|^2}\right)\prod_{k=1}^{n} p(h_k;\mathcal{X},\Theta_{old})\\
&= \sum_{h_1=1}^{J}\cdots\sum_{h_n=1}^{J}\sum_{i=1}^{n}\sum_{l=1}^{J} \delta_{h_i,l}\log\left(\frac{\pi_{l}}{\sqrt{2\pi}\sigma_{l}} \exp^{-\frac{1}{2\sigma_{l}^2}|x_i|^2}\right)\prod_{k=1}^{n} p(h_k;\mathcal{X},\Theta_{old}) \qquad (2.7)\\
&= \sum_{i=1}^{n}\sum_{l=1}^{J} \log\left(\frac{\pi_{l}}{\sqrt{2\pi}\sigma_{l}} \exp^{-\frac{1}{2\sigma_{l}^2}|x_i|^2}\right)\sum_{h_1=1}^{J}\cdots\sum_{h_n=1}^{J}\delta_{h_i,l}\prod_{k=1}^{n} p(h_k;\mathcal{X},\Theta_{old}) \qquad (2.8)
\end{aligned}$$

the integral in (2.6) is of Lebesgue sense, and the variable $\delta_{h_i,l}$ in (2.7) is the value of a

delta function s.t $\delta_{h_i,l} = \delta(h_i - l)$. For fixed i, l, in (2.8) we have

$$\begin{aligned}
&\sum_{h_1=1}^{J} \cdots \sum_{h_n=1}^{J} \delta_{h_i,l} \prod_{k=1}^{n} p(h_k; \mathcal{X}, \Theta_{old}) \\
&= \left(\sum_{h_1=1}^{J} \cdots \sum_{h_{i-1}=1}^{J} \sum_{h_{i+1}=1}^{J} \cdots \sum_{h_n=1}^{J} \prod_{k=1,k\neq i}^{n} p(h_k; \mathcal{X}, \Theta_{old}) \right) p_{h_i}(l; \mathcal{X}, \Theta_{old}) \\
&= \prod_{k=1,k\neq i}^{n} \left(\sum_{h_k=1}^{J} p(h_k; \mathcal{X}, \Theta_{old}) \right) p_{h_i}(l; \mathcal{X}, \Theta_{old}) = p_{h_i}(l; \mathcal{X}, \Theta_{old})
\end{aligned} \tag{2.9}$$

Since $\sum_{h_k=1}^{J} p(h_k; x_k, \Theta_{old}) = 1$. From (2.9) we can rewrite (2.8) as

$$\begin{aligned}
Q(\Theta; \Theta_{old}) &= \sum_{i=1}^{n} \left(\sum_{l=1}^{J} \log \left(\frac{\pi_l}{\sqrt{2\pi}\sigma_l} \exp^{-\frac{1}{2\sigma_l^2}|x_i|^2} \right) p_{h_i}(l; \mathcal{X}, \Theta_{old}) \right) \\
&= \sum_{i=1}^{n} \sum_{l=1}^{J} \log(\pi_l) p_{h_i}(l; \mathcal{X}, \Theta_{old}) + \sum_{i=1}^{n} \sum_{l=1}^{J} \log(\frac{1}{\sqrt{2\pi}\sigma_l} \exp^{-\frac{1}{2\sigma_l^2}|x_i|^2}) p_{h_i}(l; \mathcal{X}, \Theta_{old})
\end{aligned} \tag{2.10}$$

- **M-step:** In M-step, we optimize $Q(\Theta; \Theta_{old})$ w.r.t Θ. To maximize (2.10), we can maximize the term containing π_l and the term containing σ_l independently since they are not related. To find the expression for π_l, we introduce the Lagrange multiplier λ with the constraint that $\sum_l \pi_l = 1$, and solve the following equations:

$$\frac{\partial}{\partial \pi_s} \left[\sum_{i=1}^{n} \sum_{l=1}^{J} \log(\pi_l) p_{h_i}(l; \mathcal{X}, \Theta_{old}) + \lambda \left(\sum_{l=1}^{J} \pi_l - 1 \right) \right] = 0 \quad , s = 1, 2, \cdots, J$$

which is equivalent to

$$\frac{1}{\pi_s} \sum_{i=1}^{n} p_{h_i}(s; \mathcal{X}, \Theta_{old}) + \lambda = 0 \quad , s = 1, 2, \cdots, J \tag{2.11}$$

Multiply π_s on both side of (2.11) and sum the equations over s, by (2.4) we have

$$\sum_{i=1}^{n} \sum_{s=1}^{J} \frac{p_{h_i}(s, x_i; \Theta_{old})}{\sum_{j=1}^{J} p_{h_i}(j, x_i; \Theta_{old})} + \lambda = 0$$

Hence, we have $\lambda = -n$. Therefore we can solve for π_s:

$$\pi_s = \frac{1}{n} \sum_{i=1}^{n} p_{h_i}(s; \mathcal{X}, \Theta_{old}), \quad s = 1, 2, \cdots, J \tag{2.12}$$

To find expression for σ_l, we rewrite the term containing σ_l as

$$\sum_{i=1}^{n}\sum_{l=1}^{J}\left(-\log\sqrt{2\pi}+\frac{1}{2}\log(\sigma_l^{-2}-\frac{|x_i|^2}{2\sigma_l^2})\right)p_{h_i}(l;\mathcal{X},\Theta_{old}). \tag{2.13}$$

and take the derivative w.r.t σ_s^{-2} for $s=1,2,\cdots,J$ to solve the optimization problem:

$$\frac{\partial}{\partial\sigma_s^{-2}}\sum_{i=1}^{n}\sum_{l=1}^{J}\left(-\log\sqrt{2\pi}+\frac{1}{2}\log(\sigma_l^{-2}-\frac{|x_i|^2}{2\sigma_l^2})\right)p_{h_i}(l;\mathcal{X},\Theta_{old})=0.\quad s=1,2,\cdots,J$$

which is equivalent to

$$\sum_{i=1}^{n}\left(\frac{1}{2}\sigma_s^2-\frac{|x_i|^2}{2}\right)p_{h_i}(s;\mathcal{X},\Theta_{old})=0,\quad s=1,2,\cdots,J$$

Hence

$$\sigma_s^2=\frac{\sum_{i=1}^{n}|x_i|^2p_{h_i}(s;\mathcal{X},\Theta_{old})}{\sum_{i=1}^{n}p_{h_i}(s;\mathcal{X},\Theta_{old})},\quad s=1,2,\cdots,J \tag{2.14}$$

2.2 Why it works

Let $\mathcal{X}=\{x_1,x_2,\cdots,x_n\}$ be the observe data. Suppose that these data follows an distribution $p(x)$ with unknown parameters Θ and they are independent and identically distributed. The goal of EM algorithm is to maximize the log likelihood function of $p(x;\Theta)$ without calculating the derivatives of it directly since the derivatives may be complicated and the solution may not has a closed form. Suppose $P(\mathcal{X};\Theta)=p(x_1;\Theta)\cdots p(x_n;\Theta)$ and $\mathcal{L}(\mathcal{X};\Theta)=\log P(\mathcal{X};\Theta)$. The EM algorithm is perform iteratively in two step until convergence. Observe that

$$\begin{aligned}
\mathcal{L}(\mathcal{X};\Theta)=\log P(\mathcal{X};\Theta)&=\log\int P(\mathcal{X},H;\Theta)dH\\
&=\log\int\frac{P(\mathcal{X},H;\Theta)}{P(H|\mathcal{X};\Theta_{old})}P(H|\mathcal{X};\Theta_{old})dH \qquad (2.15)\\
&=\log E\left[\frac{P(\mathcal{X},\mathcal{H};\Theta)}{P(\mathcal{H}|\mathcal{X};\Theta_{old})}\mid\mathcal{X},\Theta_{old}\right]\\
&\geq E\left[\log\left(\frac{P(\mathcal{X},\mathcal{H};\Theta)}{P(\mathcal{H}|\mathcal{X};\Theta_{old})}\right)\mid\mathcal{X},\Theta_{old}\right] \qquad (2.16)\\
&=E[\log(P(\mathcal{X},\mathcal{H};\Theta))\mid\mathcal{X},\Theta_{old}]-E[\log(P(\mathcal{H}|\mathcal{X};\Theta_{old}))\mid\mathcal{X},\Theta_{old}]\\
&=E[\mathcal{L}(\mathcal{X},\mathcal{H};\Theta)\mid\mathcal{X},\Theta_{old}]-E[\log(P(\mathcal{H}|\mathcal{X};\Theta_{old}))\mid\mathcal{X},\Theta_{old}]\\
&=Q(\Theta;\Theta_{old})-E[\log(P(\mathcal{H}|\mathcal{X};\Theta_{old}))\mid\mathcal{X},\Theta_{old}] \qquad (2.17)
\end{aligned}$$

where (2.16) is derive from Jenson inequality and the fact that log is a concave function. It is also clear that when $\Theta=\Theta_{old}$, both side of inequality in (2.16) will equal to $\log(P(\mathcal{X};\Theta_{old}))$.

Letting $g(\Theta)$ be the right hand side of (2.17) then

$$\mathcal{L}(\mathcal{X};\Theta) \geq g(\Theta)$$

for all Θ, while the equality hold when $\Theta = \Theta_{old}$. Therefore any value of Θ that increases $g(\Theta)$ beyond $g(\Theta_{old})$ must also increase $\mathcal{L}(\mathcal{X};\Theta)$ beyond $\mathcal{L}(\mathcal{X};\Theta_{old})$. On the other hand maximizing $g(\Theta)$ over Θ is equivalent to maximizing $Q(\Theta;\Theta_{old})$ over Θ since the second term of $g(\Theta)$ doesn't contain variable Θ.

3 $\text{MAP}_{x,k}$ and MAP_k

In this section we will briefly go through the non-blind deconvolution $\text{MAP}_{x,k}$ brought out by Fergus in [3] , and introduce the main algorithm MAP_k which is carried out by Levin in [8]. The reason that we introduce $\text{MAP}_{x,k}$ first is because it is closed related to MAP_k and also sheds light on the following MAP_k algorithm. Both of the methods are close related to MLE but with some changes which we'll discuss later.

3.1 Maximum A Posteriori estimation

The Maximum A Posteriori estimation (MAP) is very similar to MLE but with additional prior knowledge. To be specific, the MLE can be obtain from optimizing the likekihood function directly

$$\hat{\Theta}_{MLE} = \arg\max_{\Theta} P(\mathcal{X};\Theta)$$

where $\mathcal{X} = \{\nabla y\}$ is the observed blur image in gradient domain, $\Theta = \{\nabla x, k\}$ is the unknown parameters we wish to estimate and $P(\mathcal{X};\Theta) = P(\mathcal{X}|\Theta) = p(\nabla y|\nabla x, k)$. One can find that MLE doesn't take account the prior information. Since one feature of natural image is that the gradient histogram of it is often similar to a heavy tail distribution as we mention in subsection 1.2. We may want to make use of this prior property. Hence Maximum A Posteriori estimation (MAP) came in. MAP consider the posterior probability which takes account not only the likelihood function but also the prior distributions on $\nabla x, k$ as shown in (1.10) and (1.11). Given the blur image y, the MAP of image deconvolution is to find the Maximum A Posteriori estimation of both $\nabla x, k$:

$$\hat{\Theta}_{MAP} = (\hat{\nabla x}, \hat{k}) = \arg\max_{\nabla x,k} p(\nabla x, k|\nabla y) \tag{3.1}$$

$$= \arg\max_{\nabla x,k} \frac{p(\nabla y|\nabla x, k)p(\nabla x)p(k)}{p(\nabla y)} \tag{3.2}$$

$$= \arg\max_{\nabla x,k} p(\nabla y|\nabla x, k)p(\nabla x)p(k) \tag{3.3}$$

By (1.8) (1.10) and (1.11) the maximization can be rewrite as

$$(\hat{\nabla x}, \hat{k}) = \arg\max_{\nabla x,k} \left\{ \left(\frac{1}{(\sqrt{2\pi}\eta)^N} \exp^{-\frac{1}{2\eta^2}\|\nabla x \otimes k - \nabla y\|^2} \right) \left(\prod_{i=1}^{N} \prod_{l=1}^{2} \left(\sum_{j=1}^{J} \frac{\pi_j}{\sqrt{2\pi}\sigma_j} \exp^{-\frac{1}{2\sigma_j^2}|(g_l \otimes x)(i)|^2} \right) \right) \left(\lambda \exp^{-\lambda \sum_{i=1}^{M} |k(i)|} \right) \right\} \tag{3.4}$$

$$= \arg\min_{\nabla x,k} \left\{ \left(\frac{1}{2\eta^2} \|\nabla x \otimes k - \nabla y\|^2 \right) - \left(\sum_{i=1}^{N} \sum_{l=1}^{2} \log \left(\sum_{j=1}^{J} \frac{\pi_j}{\sqrt{2\pi}\sigma_j} \exp^{-\frac{1}{2\sigma_j^2}|(g_l \otimes x)(i)|^2} \right) \right) + \left(\lambda \sum_{i=1}^{M} |k(i)| \right) \right\} \tag{3.5}$$

Eq(3.5) is obtained by taking $-\log$ of eq(3.4) and discards the terms that didn't contain $\nabla x, k$. A straightforward image deconvolution solution is to solve eq(3.5) directly. Such optimization is equivalent to solving a regularized least square problem. In both Fergus and Levin works [3][7], they solved the optimization of $\nabla x, k$ using conjugate gradient search. However the result isn't ideal. An explanation is that the objective function in (3.5) tends to minimize all gradients into zeros while we expect there exist some large gradients in natural image. On the other hand if we decrease the noise level η which means increasing the weight on data fitting term, the result tends to be a non blind image y and a delta kernel $k = \delta$ since the no blur explaination has zero value in the data fitting term. Additionally in [3], they found that the MAP objective function tend to be very susceptible to poor local minima. In the work of Levin [7], they point out that no matter which kind of prior $p(\nabla x)$ that characterizes the heavy tail property is used, the most likely event of prior term in eq(3.5) is the fully flat image. This phenomenon is robust to the exact choice of prior, and replacing the gradient with higher order derivatives or changing to more accurate prior doesn't affect the result.

3.2 $\mathbf{MAP}_{x,k}$

In order to solve the problems cause by direct optimization of (3.5), Fergus et al find another way to handle it in [3] . Instead of maximizing the posterior probability $p(\nabla x, k|\nabla y)$, the author adopt a variational Bayesian approach. First they approximate $p(\nabla x, k|\nabla y)$ with $q(\nabla x, k)$ where $q(x, k)$ is chosen from some simply parametric family and then compute the kernel k

with maximum marginal probability. That is

$$\hat{k} = \arg\max_{k} \int q(\nabla x, k)\ d(\nabla x) \tag{3.6}$$

The author claim that optimizing the marginal probability as in eq(3.6) will avoid the overfitting that can occur when selecting a single best estimate image ∇x. Later in Levin's work [8], they point out that optimizing the marginal probability over ∇x is the major reason of their success which we'll discuss in next section. After getting estimated kernel $\hat{k}$, they perform a well known non blind deconvolution on y with $\hat{k}$ to find the estimated image $\hat{x}$

In practice, Fergus [3] assume $q(\nabla x, k) = q(\nabla x)q(k)$ where $q(\nabla x)$ is a gaussain distribution , and $q(k)$ is a rectified Gaussian distribution:

$$\begin{cases} q(\nabla x) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_i} \exp^{-\frac{1}{2\sigma_i^2}(\nabla x(i) - u_i)^2} \\ q(k) = \prod_{i=1}^{M} \left(\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\frac{m_i}{s_i}} e^{-\frac{t^2}{2}} dt\ \delta(k(i)) + \frac{1}{\sqrt{2\pi}s_i} \exp^{-\frac{(k(i)-m_i)^2}{2s^2}}\ U(k(i)) \right) \end{cases} \tag{3.7}$$

where $U(x)$ is the unit step function and $\delta(x)$ is the Dirac delta function. N is the number of pixels in image and M is number of pixels in kernel.

Since they assume $\nabla x, k$ are independent and by (3.7), the optimization in (3.6) becomes:

$$\begin{aligned} \hat{k} &= \arg\max_{k}\ q(k) \int q(\nabla x)\ d(\nabla x) \\ &= \arg\max_{k}\ q(k) \\ &= m = (m_1, m_2, \cdots, m_M) \end{aligned}$$

since $q(k)$ is a rectified Gaussian distribution with mean $m = (m_1, \cdots, m_M)$.

The only calculation we need is the approximation of $q(\nabla x, k)$ and $p(\nabla x, k)$. The variational algorithm minimizes the KL divergence between $q(\nabla x, k)$ and $p(\nabla x, k)$:

$$F(q) = \int q(\nabla x, k) \log \left(\frac{q(\nabla x, k)}{p(\nabla x, k)} \right) dxdk \tag{3.8}$$

To solve minimization of (3.8), the author adopt a coordinate descent method. The means $u = (u_1, \cdots, u_N), m = (m_1, \cdots, m_M)$ of $q(\nabla x)$ and $q(k)$ are set to initial values of ∇x and k respectively. The initial variances $\sigma = (\sigma_1, \cdots, \sigma_N), s = (s_1, \cdots, s_M)$ are set high, reflecting the lack of certainty in the intial estimates. First fix $q(k)$, optimize $q(\nabla x)$ and next fix $q(\nabla x)$

optimize $q(k)$. Repeat the optimization until convergence. The result mean m of $q(k)$ is the desire kernel estimation $\hat{k}$. Take the estimated kernel k to perform a non blind image deconvolution such as Richardson-Lucy algorithm to find the estimated image x.

3.3 $\mathbf{MAP}_k$

MAP_k is an blind image deconvolution brought out by Levin et al in [8]. Levin state that since the number of unknowns x, k in MAP out run the observed blur image y, the measurements may not be enough for estimation. On the other hand, the estimation theory has told us that given enough measurements, MAP do approach the true solution. Hence, the key to success is to utilize the special property of blind image deconvolution: there is a strong asymmetry between the dimensionalities of x and k. The size of x exceed k significantly. Another property is that while the dimensionalities of x grow with image size, the dimensionalities of kernel k remain the same. In Levin's work [8], they come up with a method combining the asymmetry property and the variational Bayesian method propose by Fergus in [3] MAP_k. Instead of optimizing the posterior probability of both $p(\nabla x, k|\nabla y)$, we first find the best k by optimizing the posterior probability with k along $p(k|y)$ and then perform a non blind deconvolution to find the latent image x just like Fergus have done. A MAP_k estimator selects $\hat{k}$ s.t

$$\begin{aligned}
\hat{k} &= \arg\max_k p(k|y) && (3.9)\\
&= \arg\max_k \frac{p(y|k)p(k)}{p(y)} && (3.10)\\
&= \arg\max_k \left(\int \frac{p(y,x|k)}{p(y)}\, dx \right) (p(k)) \\
&= \arg\max_k \left(\int p(x,k|y) dx \right) && (3.11)
\end{aligned}$$

Comparing (3.10) and (3.6), we can see that they are very similar. If we substitute $q(\nabla x, k)$ with original probability $p(\nabla x, k)$, both equations are the same. To see the role of marginalization, consider an scalar blind deconvolution model. Suppose a blur scalar y is observed, and, similar to (1.1), $y = kx + n$ where $k \in \mathbb{R}$ is the unknown kernel. Similar to (1.5) and (1.7), we assume the noise $n \sim N(0, \eta^2)$ and the prior $p(x)$ is a gaussian that is $x \sim N(0, \sigma^2)$. Therefore

$$p(x,k|y) = \frac{p(y|x,k)p(x)}{p(y)} \propto p(y|x,k)p(x) \propto \exp^{-\frac{1}{2\eta^2}(kx-y)^2 - \frac{x^2}{2\sigma^2}}$$

we can see that when $x \to 0$ and $k \to +\infty$ the cost of $p(x, k|y)$ becomes larger. Since the MAP compute the optimum of $p(x, k|y)$ directly, the solutions tend to be $x \to 0$ and $k \to +\infty$. In contrast, if we consider the MAP_k approach:

$$\hat{k} = \arg\max_k \int p(x, k|y)\ dx = \arg\max_k \int \exp^{-\frac{1}{2\eta^2}(kx-y)^2 - \frac{x^2}{2\sigma^2}}\ dx$$

we can see that the cost function of MAP_k is no longer maximize by $k \to \infty$ since the term $(kx - y)^2$ will be large for all $x \neq 0$ when $k \to \infty$, causing the integral to be small.
Levin et al also prove a claim in paper for image deconvolution by MAP_k:

Claim 1 *Suppose the image x is arbitrary large, sampling from the prior $p(x)$ and $y = k \otimes x + n$. Then the cost function of MAP_k (i.e $p(y|k)$) is maximized by the true kernel k^*. Moreover, if $\arg\max_k p(y|k)$ is unique, $p(k|y)$ approaches a delta function.*

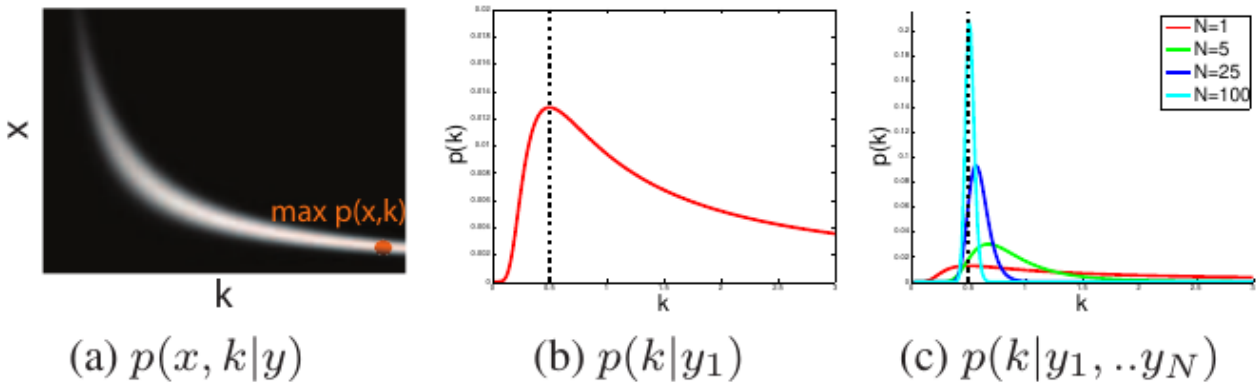


Figure 3: Comparison of 1D $\text{MAP}_{x,k}$ and MAP_k in [7]. (a) The cost function of $\text{MAP}_{x,k}$ $p(x, k|y)$ tends to be optimized by $x \to 0$ and $k \to \infty$. (b)The cost function of MAP_k $p(k|y_1)$ will optimized by the true kernel instead of trivial one in $\text{MAP}_{x,k}$ (c) While more data are included, the cost function of MAP_k $p(k|y_1 m \cdots, y_n)$ is more sharp.

Figure 3 (c) plots $p(k|y)$ for scalar deconvolution with N observation, illustrating that as N increase, the cost function $p(k|y)$ tend to be the delta function.

3.3.1 Difference between MAP_k and MAP via loss function perspective

Another way to understand the difference between MAP_k and MAP, we look into the definition of a Bayes estimator. Define $L(x, \hat{x}, k, \hat{k})$ to be the loss function. The Bayes estimator of unknown parameter $\hat{\Theta} = (\hat{x}, \hat{k})$ is:

$$(\hat{x},\hat{k}) = \arg\min_{\bar{\Theta}=(\bar{x},\bar{k})} \int\int p(\Theta|y)L(\bar{\Theta},\Theta)d\Theta$$
$$= \arg\min_{\bar{x},\bar{k}} \int\int p(x,k|y)L(x,\bar{x},k,\bar{k})dxdk \tag{3.12}$$

A simple way to derive MAP estimator is to define the loss function as:

$$L(x,\hat{x},k,\hat{k}) = 1 - \delta((\hat{x},\hat{k}) - (x,k)) \tag{3.13}$$

Taking (3.12) as the loss function in (3.11), we will derive $(\hat{x},\hat{k}) = \arg\min_{\bar{x},\bar{k}} p(\bar{x},\bar{k}|y)$ which is exactly the MAP estimator. The sensitivity of loss function (3.12) has been stress out many times in signal process literature. The all or nothing loss is too harsh for many signal processing applications since it ignores all other information around the mode.

Instead of such sensitive loss function, taking a smoother one such as mean square error may be more reasonable. Suppose that

$$L(x,\hat{x},k,\hat{k}) = |x-\hat{x}|^2 + |k-\hat{k}|^2 \tag{3.14}$$

then the Bayes estimator becomes

$$(\hat{x},\hat{k}) = \arg\min_{\bar{x},\bar{k}} \int\int p(x,k|y)(\|x-\bar{x}\|^2 + \|\bar{k}-\bar{k}\|^2)\ dxhk \tag{3.15}$$

To find the estimated image $\hat{x}$, we differentiate (3.14) with $\bar{x}$ and solve the equation

$$\int\int p(x,k|y)2(x-\bar{x})\ dxhk = 0$$

Therefore

$$\begin{aligned}
\hat{x} &= \int\int xp(x,k|y)\ dxdk \\
&= \int\int \frac{p(x,k,y)}{p(y)}x\ dxdk \\
&= \int\int \frac{p(x,k,y)}{p(k,y)}\frac{p(k,y)}{p(y)}x\ dxdk \\
&= \int\int p(x|k,y)p(k|y)x\ dxdk \\
&= \int p(k|y)\left(\int p(x|k,y)x\ dx\right)dk \\
&= \int p(k|y)\mu^k dk
\end{aligned} \tag{3.16}$$

where $\mu^k = \int p(x|k,y)x\ dx$ can be interpreted as a non blind image deconvolution with kernel k since it is the MMSE of x given k, y:

$$\mu^k = \arg\min_{\tilde{x}} E(\|x - \tilde{x}\|^2 \mid k, y) = \arg\min_{\tilde{x}} \int \|\bar{x} - \tilde{x}\|^2 p(\bar{x}|k,y) d\bar{x}$$

If $p(k|y)$ has a unique maximum k^{MAP} then, by Claim 1, $p(k|y)$ is delta function of k, and hence (3.15) becomes

$$\hat{x} = \mu^{(k^{MAP})}$$

Therefore finding the estimated image $\hat{x}$ is equivalent to solve the maximum a posterior estimation of kernel k follow by a non blind image deconvolution.

3.3.2 EM based algorithm of MAP_k

Although we have shown adopting MAP_k will yield a better solution than traditional MAP method, the marginalize term in (3.10) of MAP_k need to be evaluated. If we assume the prior $p(x)$ in (1.7) to be a gaussian instead of mixture of gaussain, the marginalize term can be calculate easily with closed form and also yield a fine solution of k. Under the gaussian prior and noise in image domain (1.5), we can find the approximated solution by using Expectation-Maximization Algorithm. To see how the algorithm work, we clear the assumptions first. Let the blur image y be the observe data $\mathcal{X} = \{y\}$, the latent image x be the hidden variable $\mathcal{H} = \{x\}$ with gaussian $p(x)$ and the unknown parameter $\Theta = k$ is waiting to be estimated. The goal of EM algorithm is finding the best estimation that maximize the likelihood function $\mathcal{L}(\mathcal{X}; \Theta) = \log(p(y|k))$ while MAP_k aim to find k that maximize $p(k|y) \propto p(y|k)p(k)$ as shown in (3.9). Hence we consider the minimization iteratively with cost function $F(k)$ s.t

$$-\log(p(y|k)) - \log(p(k)) \leq -Q(k; k_{old}) - \log(p(k)) = F(k) \tag{3.17}$$

where the first term will maximize $p(y|k)$ by EM algorithm shown in section2.2 while the second term makes the kernel to be sparse.

E-Step First we calculate the conditional probability $p(x|y,k)$. From Bayes theorem, we have

$$p(x|y,k) \propto p(x,y|k)$$

and

$$p(x, y|k) = p(y|x, k)p(x)$$

taking $-\log$ on both side of equation, by (1.5) and (1.7)

$$\begin{aligned}
-\log(p(x, y|k)) &= -\log(p(y|x, k)) - \log(p(x)) \\
&= \frac{1}{2\eta^2}\|k \otimes x - y\|^2 + N\log(\sqrt{2\pi}\eta) + \sum_{l=1}^{2}\left(\frac{1}{2\sigma_1^2}\|g_l \otimes x\|^2 - \log(\pi_1) + \log(\sqrt{2\pi}\sigma_1)\right) \\
&= \frac{1}{2\eta^2}\|k \otimes x - y\|^2 + \sum_{l=1}^{2}\frac{1}{2\sigma_1^2}\|g_l \otimes x\|^2 + c \qquad (3.18)
\end{aligned}$$

where $c = N\log(\sqrt{2\pi}\eta) + 2\log(\sqrt{2\pi}\sigma_1) - 2\log(\pi_1)$ is the constant.

Express the 2d convolution $\otimes$ as a linear transformation

$$k \otimes x = T_k x$$

Same process for $g_l \otimes x = T_{g_l} x$. Then we can rewrite (3.17) as

$$-\log(p(x, y|k)) = \frac{1}{2}x^T A_k x - b_k^T x + c \qquad (3.19)$$

where

$$A_k = \frac{1}{\eta^2}T_k^T T_k + \frac{1}{\sigma_1^2}\sum_{l=1}^{2}T_{g_l}^T T_{g_l} \qquad (3.20)$$

$$b_k = \frac{1}{\eta^2}T_k y \qquad (3.21)$$

Note that A_k is a symmetric matrix and x, y, k are vectorized along the rows.

From (3.18) we can see that $p(x|y, k)$ is also a gaussian. Suppose it has a mean μ and a covariance C, the probability $p(x|y, k)$ can be written as

$$\begin{aligned}
-\log(p(x|y, k)) &= \frac{1}{2}(x - \mu)^T C^{-1}(x - \mu) + c \\
&= \frac{1}{2}x^T C^{-1}x - \mu^T C^{-1}x + \frac{1}{2}\mu^T C^{-1}\mu + c \\
&= \frac{1}{2}x^T C^{-1}x - \mu^T C^{-1}x + c \qquad (3.22)
\end{aligned}$$

where c is sum of all constant that doesn't contain x.

Comparing the coefficient of (3.18) and (3.21)

$$C^{-1} = A_k$$

$$\mu^T C^{-1} = \mu^T A_k = b_k^T$$

Since $A_k^T = A_k$, $(\mu^T A_k)^T = A_k\mu$. Therefore we have

$$C = A_k^{-1} \tag{3.23}$$

$$\mu = A_k^{-1} b_k \tag{3.24}$$

In practice, the convolution can be calculated more easily in frequency domain. Since fourier transform of convolution matrix is diagonal, the covariance can be derive directly. Therefore μ can also be compute directly, avoid solving the linear system numerically.

M-step In this step we first need to find the form of $Q(k; k_{old})$ then minimize the cost function

$$F(k) = -Q(k; k_{old}) - \log(p(k)) \tag{3.25}$$

From (2.17) we have

$$\begin{aligned} -Q(k; k_{old}) &= E[-\log(p(x|y,k)) \mid y, k_{old}] \\ &= \int \frac{1}{2\eta^2} \|k \otimes \bar{x} - y\|^2 p(\bar{x}|y, k_{old}) \; d\bar{x} \end{aligned} \tag{3.26}$$

For given $x = \bar{x}$,

$$\|k \otimes x - y\|^2 = k^T A_x k - b_x^T k \tag{3.27}$$

where

$$A_x = T_x^T T_x$$

$$b_x = T_x^T y$$

To illustrate the matrix A_x, we give an example of 2d convolution matrix.

Suppose

$$x = \begin{bmatrix} x_{0,0} & x_{0,1} & \cdots & \cdots & x_{0,n_2-1} \\ \vdots & & \ddots & & \vdots \\ x_{n_1-1,0} & x_{n_1-1,2} & \cdots & \cdots & x_{n_1-1,n_2-1} \end{bmatrix}$$

$$k = \begin{bmatrix} k_{0,0} & k_{0,1} & \cdots & \cdots & k_{0,m_2-1} \\ \vdots & & \ddots & & \vdots \\ k_{m_1-1,0} & k_{m_1-1,2} & \cdots & \cdots & k_{m_1-1,m_2-1} \end{bmatrix}$$

where $n_1 * n_2 = N$, $m_1 * m_2 = M$. For convenience we assume m_1, m_2 to be odd integers. Expand the image x and 2d kernel k w.r.t row and yield column vectors v_x and v_k

$$v_x = [x_{0,0}, x_{0,1}, \cdots, x_{0,n_2-1}, x_{1,0}, \cdots, x_{1,n_2-1}, \cdots, \cdots, x_{n_1-1,0}, \cdots, x_{n_1-1,n_2-1}]^T$$

$$v_k = [k_{0,0}, k_{0,1}, \cdots, k_{0,m_2-1}, k_{1,0}, \cdots, k_{1,m_2-1}, \cdots, \cdots, k_{m_1-1,0}, \cdots, k_{m_1-1,m_2-1}]^T$$

Let $z = k \otimes x$, and v_z be the corresponding vectorization of z. Then for $(i_1, i_2) \in [0, n_1 - 1] \times [0, n_2 - 1]$

$$z(i_1, i_2) = (k \otimes x)(i_1, i_2) = \sum_{j_1=-\frac{m_1-1}{2}}^{\frac{m_1-1}{2}} \sum_{j_2=-\frac{m_2-1}{2}}^{\frac{m_2-1}{2}} \tilde{x}(i_1 - j_1, i_2 - j_2)\tilde{k}(j_1 + \frac{m_1-1}{2}, j_2 + \frac{m_2-1}{2})$$

where

$$\tilde{x}_{(i_1,i_2)} = \begin{cases} x_{(i_1,i_2)} & ,(i_1, i_2) \in [0, n_1 - 1] \times [0, n_2 - 1] \\ 0 & ,\text{otherwise} \end{cases}$$

and

$$\tilde{k}_{(j_1,j_2)} = \begin{cases} k_{(j_1,j_2)} & ,(j_1, j_2) \in [0, m_1 - 1] \times [0, m_2 - 1] \\ 0 & ,\text{otherwise} \end{cases}$$

we can write $z = k \otimes x$ by a linear transformation between vectors v_z, v_k with matrix $T_x \in \mathbb{R}^{N \times M}$

$$v_z = T_x v_k$$

Let $\bar{m_1} = \frac{m_1-1}{2}$ and $\bar{m_2} = \frac{m_2-1}{2}$. And define

$$\begin{cases} h_1(n) = n - \left[\frac{n}{n_2}\right] n_2 + \bar{m_1} \\ h_2(n) = n - \left[\frac{n}{m_2}\right] m_2 + \bar{m_2} \end{cases}$$

where $[\cdot]$ is the floor function Then we can write

$$T_x(i, j) = \tilde{x}_{(i-\left[\frac{i}{m_2}\right]m_2+\bar{m_1}, j-\left[\frac{j}{n_2}\right]n_2+\bar{m_2})} = \tilde{x}_{(h_1(i),h_2(j))} \tag{3.28}$$

Hence for $(j_1, j_2) \in [0, m_1 - 1] \times [0, m_2 - 1]$

$$\begin{aligned} A_x(j_1, j_2) = (T_x^T T_x)(j_1, j_2) &= \sum_{i=0}^{N-1} T_x^T(j_1, i) T_x(i, j_2) \\ &= \sum_{i=0}^{N-1} T_x(i, j_1) T_x(i, j_2) \\ &= \sum_{i=0}^{N-1} \tilde{x}_{(h_1(i),h_2(j_1))} \tilde{x}_{(h_1(i),h_2(j_2))} \end{aligned} \tag{3.29}$$

From (3.29), we can see that the (j_1, j_2)-th element of matrix A_x is the autocorrelation of columns $h_2(j_1)$ and $h_2(j_2)$ in $\tilde{x}$.

Now combining (3.26) and (3.27), we derive

$$\begin{aligned} -Q(k; k_{old}) &= \frac{1}{2\eta^2}\left[k^T\left(\int A_x p(x|y, k_{old})\ dx\right)k - \left(\int b_x^T p(x|y, k_{old})\right)k\ dx\right] \\ &= \frac{1}{2\eta^2}\left(k^T\bar{A}k - \bar{b}k\right) \end{aligned} \tag{3.30}$$

Observe that from (3.29)

$$\begin{aligned} \bar{A}(j_1, j_2) &= \int A_x(j_1, j_2)p(x|y, k_{old})\ dx \\ &= \int \sum_{i=0}^{N-1} \tilde{x}_{(h_1(i),h_2(j_1))}\tilde{x}_{(h_1(i),h_2(j_2))}p(x|y, k_{old})\ dx \\ &= E\left[\sum_{i=0}^{N-1} \tilde{x}_{(h_1(i),h_2(j_1))}\tilde{x}_{(h_1(i),h_2(j_2))} \mid y, k_{old}\right] \\ &= \sum_{i=0}^{N-1} E[\tilde{x}_{(h_1(i),h_2(j_1))}\tilde{x}_{(h_1(i),h_2(j_2))} \mid y, k_{old}] \\ &= \sum_{i=0}^{N-1} \left(\tilde{\mu}_{(h_1(i),h_2(j_1))}\tilde{\mu}_{(h_1(i),h_2(j_2))} + Cov(\tilde{x}_{(h_1(i),h_2(j_1))}, \tilde{x}_{(h_1(i),h_2(j_2))})\right) \end{aligned} \tag{3.31}$$

where

$$\tilde{\mu}_{i,j} = \begin{cases} \mu(in_2 + j) & , (i, j) \in [0, n_1 - 1] \times [0, n_2 - 1] \\ 0 & , \text{otherwise} \end{cases}$$

and

$$Cov(\tilde{x}_{i_1,j_1}, \tilde{x}_{i_2,j_2}) = \begin{cases} C(i_1n_2 + j_1, i_2n_2 + j_2) & , (i_c, j_c) \in [0, n_1 - 1] \times [0, n_2 - 1],\ c = 1, 2 \\ 0 & , \text{otherwise} \end{cases}$$

For $\bar{b}$, we have

$$\begin{aligned} \bar{b}(j) &= \int b_x^T(j)p(x|y, k_{old})\ dx = \int \sum_{i=1}^{N} T_x(i, j)y(i)\ dx \\ &= \int \sum_{i=1}^{N} \tilde{x}_{(h_1(i),h_2(j))}y(i)\ dx \\ &= E[\sum_{i=1}^{N} \tilde{x}_{(h_1(i),h_2(j))}y(i) \mid y, k_{old}] \\ &= \sum_{i=1}^{N} \tilde{\mu}_{(h_1(i),h_2(j))}y(i) \end{aligned} \tag{3.32}$$

From (3.30) (3.31) and (3.32) we can obtain the explicit form of $-Q(k; k_{old})$. The next step is to minimize the cost function $F(k)$. From the sparse kernel prior assumption (1.11),

$$F(k) = \frac{1}{2\eta^2}\left(k^T\bar{A}k - \bar{b}^T k\right) + \beta\|k\|_0 \tag{3.33}$$

We implemented a method in the research [12, 13]. The minimization could be solved by adding auxiliary value r to the objective function and modify the cost function to

$$(\hat{k}, \hat{r}) = \arg\min_{k,r} \frac{1}{2\eta^2}\left(k^T\bar{A}k - \bar{b}^T k\right) + \frac{1}{2}\beta\|k - r\|^2 + \lambda\|r\|_0 \tag{3.34}$$

where r is the auxiliary value, and λ is the weight respect to r. The minimization is solved by coordinate descending. First we fix k and solve for r. According to [13], it could be represented as

$$r = \begin{cases} k & \text{, if } |k|^2 \geq \dfrac{\lambda}{\beta} \\ 0 & \text{, otherwise} \end{cases} \tag{3.35}$$

the weight β will increase double in each iteration of coordinate decsend method, as the kernel become more sparse. Next we solve for k

$$\hat{k} = \arg\min_{k} \frac{1}{2\eta^2}\left(k^T\bar{A}k - \bar{b}^T k\right) + \frac{1}{2\beta}\|k - r\|^2 \tag{3.36}$$

The 2-norm in the second term can also express in a quadratic form

$$\|k - r\|^2 = k^T k - 2r^T k r + r^T r$$

Hence

$$\hat{k} = \arg\min_{k>0, \|k\|_1=1} \frac{1}{2\eta^2}\left(k^T(\bar{A} + \frac{\eta^2}{\beta}\mathbf{1}_M)k - (\bar{b}^T + \beta r^T)k\right) \tag{3.37}$$

where $\mathbf{1}_M$ is the identity matrix with size $M \times M$. The objective function is of quadratic form with constraint $k > 0, \|k\|_1 = 1$. We implemented matlab function *quadprog* to find the numerical solution for $\hat{k}$

4 Non blind deconvolution

After obtaining the estimated kernel $\hat{k}$ from MAP_k algorithm, we will need to perform a non blind deconvolution on the blur image y. The most intuitive way is finding the wiener filter w that minimize the mean square error:

$$\hat{x} = \arg\min_{\bar{x}} E\left[\|x - \bar{x}\|_2^2\right]$$

where

$$y = \hat{k} \otimes x + n$$

$$\bar{x} = w \otimes y$$

with the noise n follows a normal distribution $\mathcal{N}(0, \eta^2)$. Since the cost function is of 2-norm, by parseval theorem we can take the minimization into frequency domain. Considering the fourier transform $Y, \hat{K}, X, N$of $y, \hat{k}, x, n$, the minimization becomes

$$\hat{X} = \arg\min_{\bar{X}} E\left[\|X - \bar{X}\|_2^2\right]$$

where

$$Y = \hat{K} \cdot X + N$$

$$\bar{X} = W \cdot Y$$

As Gonzalez shows in book [5], under the independent assumption of random variables X, N, the solution of W is

$$W(u, v) = \frac{K^*(u, v)}{|K(u, v)|^2 + \text{NSR}(N, X)}$$

where the constant NSR interpreted as the Noise to Signal ratio between N and X,

$$\text{NSR}(N, X) = \frac{\sum_u \sum_v |N(u, v)|^2}{\sum_u \sum_v |X(u, v)|^2}$$

since we doesn't has the exact value of NSR, we'll need to estimated by our own.

Although using weiner filter to deconvolute the blur image seem to be reasonable, but in many cases the deblurred image will affect by ring effect significantly. The research [13] had pointed out that it is not a good way to set the noise to be a random variable following a Gaussian

distribution. In order to achieve high robustness, avoid increasing the noise of the blur image and the ring effect, the researchers proposed a model with sparse prior on noise. In this section, we will explain more details about Fast TV-l_1 Deconvolution brought out in [11][13][14].

4.1 Fast TV-l_1 Deconvolution

Different from the Gaussian noise for deblurring the image y. The objective function in the research is proposed as

$$\hat{x} = \arg\min_{\bar{x}} \|\hat{k} \otimes \bar{x} - y\|_1 + \lambda \|\nabla \bar{x}\|_1 \tag{4.1}$$

where $\|\hat{k} \otimes \bar{x} - y\|_1$ is the data fitting term with l_1 norm, $\|\nabla \bar{x}\|_1$ is the regulariztion term, and λ is the weight. To solve the l_1 norm minimization, we adopted alternating minimization method basing on half-quadratic splitting.

The method uses auxiliary values which make the objective function turn into l_2 norm problem. Therefore, the objective function becomes more solvable. To be explicit, the auxiliary values v and w are used for modifying, and the modified function can be represented as

$$(\hat{x}, \hat{w}, \hat{v}) = \arg\min_{\bar{x}, w, v} \frac{1}{2\beta} \|\hat{k} \otimes \bar{x} - y - v\|_2^2 + \frac{1}{2\theta} \|\nabla \bar{x} - w\|_2^2 + \|v\|_1 + \lambda \|w\|_1 \tag{4.2}$$

where $v = \mathbb{R}^{n_1 \times n_2}$, $w = (w^x_{i_1,i_2}, w^y_{i_1,i_2}, \cdots, w^x_{n_1,n_2}, w^y_{n_1,n_2})$ with $w_{i_1,i_2} = (w^x_{i_1,i_2}, w^y_{i_1,i_2})$ and $\theta, \beta \in \mathbb{R}$. Note that the modified objective function (4.2) approach to solution of original objective function (4.1) when $v \to 0$ and $w \to 0$.

To solve the modified objective function, the researchers utilize the coordinate descend method that is iteratively finding the solutions of $\hat{x}, w, v$ one at a time while fixing other until convergence. Since solving $\hat{x}$ is our main target, we need to minimize the values of v and w first [11][14].

- **Solve for $\hat{v}$:** The objective function w.r.t v separated from (4.2) is

$$\hat{v} = \arg\min_{v} \frac{1}{2} \|v - (\hat{k} \otimes \hat{x} - y)\|_2^2 + \beta \|v\|_1 \tag{4.3}$$

  Let $F(v) = F(v_{1,1}, \cdots, v_{n_1,n_2}) = \frac{1}{2}\|v - (\hat{k} \otimes \hat{x} - y)\|_2^2 + \beta \|v\|_1$ be the objective function. The optimization of objective function F can be categorized to a single-variable optimization

problem because the variables for different pixels are not spatially coupled. Thus we only need to solve for the critical points s.t:

$$\frac{\partial F}{\partial v_{i_1,i_2}} = 0 \quad \Rightarrow \quad v_{i_1,i_2} - ((\hat{k} \otimes \hat{x})_{i_1,i_2} - y_{i_1,i_2}) + \beta \frac{\partial |v_{i_1,i_2}|}{\partial v_{i_1,i_2}} = 0 \tag{4.4}$$

Thus we have

$$\hat{v}_{i_1,i_2} = \text{sign}((\hat{k} \otimes \hat{x})_{i_1,i_2} - y_{i_1,i_2}) \max(|(\hat{k} \otimes \hat{x})_{i_1,i_2} - y_{i_1,i_2}| - \beta, 0) \tag{4.5}$$

- **Solve for $\hat{w}$:** Similarly for w, we have the objective function

$$\hat{w} = \arg\min_{w} \frac{1}{2}\|w - \nabla \hat{x}\|_2^2 + \theta\lambda\|w\|_1 \tag{4.6}$$

Let $F(w) = F(w^x_{i_1,i_2}, w^y_{i_1,i_2}, \cdots, w^x_{n_1,n_2}, w^y_{n_1,n_2}) = \frac{1}{2}\|w - \nabla \hat{x}\|_2^2 + \theta\lambda\|w\|_1$ be the objective function, similar to (4.5), the critical points satisfies

$$\frac{\partial F}{\partial w^*_{i_1,i_2}} = 0 \quad \Rightarrow \quad (w^*_{i_1,i_2} - \partial_* \hat{x}_{i_1,i_2}) + \theta\lambda \frac{\partial |w^*_{i_1,i_2}|}{\partial w^*_{i_1,i_2}} = 0 \tag{4.7}$$

hence solution can be written as

$$\begin{cases} \hat{w}^x_{i_1,i_2} = \dfrac{(\partial_x \hat{x})_{i_1,i_2}}{|(\partial_x \hat{x})_{i_1,i_2}|} \max(|(\partial_x \hat{x})_{i_1,i_2}| - \theta\lambda, 0) \\ \hat{w}^y_{i_1,i_2} = \dfrac{(\partial_y \hat{x})_{i_1,i_2}}{|(\partial_y \hat{x})_{i_1,i_2}|} \max(|(\partial_y \hat{x})_{i_1,i_2}| - \theta\lambda, 0) \end{cases} \tag{4.8}$$

where we followed the convention $0 \cdot (0/0) = 0$.

- **Solve for $\hat{x}$:** Last, for solving $\hat{x}$, we have the objective function

$$\hat{x} = \arg\min_{\bar{x}} \|\hat{k} \otimes \bar{x} - y - v\|_2^2 + \frac{\beta}{\theta}\|\nabla \bar{x} - w\|_2^2 \tag{4.9}$$

We can see that the cost function is of 2-norm, by Parseval theorem we can solve the minimization more easily in frequency domain. Therefore the objective function becomes

$$\hat{X} = \arg\min_{\bar{X}} \|\hat{K}\bar{X} - Y - V\|_2^2 + \frac{\beta}{\theta}\|\nabla \bar{X} - W\|_2^2 \tag{4.10}$$

Let $F(\bar{X}) = \|\hat{K}\bar{X} - Y - V\|_2^2 + \frac{\beta}{\theta}\|\nabla \bar{X} - W\|_2^2$. The critical points can be solved from

$$\begin{aligned} \frac{\partial F}{\partial \bar{X}} = 0 \quad &\Rightarrow \quad 2\hat{K}^*(\bar{X}\hat{K} - (Y + V)) + \frac{2\beta}{\theta}\tilde{\nabla}^*(\tilde{\nabla}\bar{X} - W) = 0 \\ &\Rightarrow \quad \left(\hat{K}^*\hat{K} + \frac{\beta}{\theta}\tilde{\nabla}^*\tilde{\nabla}\right)\bar{X} = \hat{K}(Y + V) + \frac{\beta}{\theta}\left(\tilde{\nabla}^* W\right) \end{aligned} \tag{4.11}$$

where $\hat{K}, \bar{X}, Y, V, \tilde{\nabla}$ are fast fourier transform of $\hat{k}, \bar{x}, y, v, \nabla$ respectively.

Thus we have

$$\hat{X} = \frac{\hat{K}(Y+V) + \frac{\beta}{\theta}\left(\tilde{\nabla}^* W\right)}{\left(\hat{K}^*\hat{K} + \frac{\beta}{\theta}\tilde{\nabla}^*\tilde{\nabla}\right)} \tag{4.12}$$

After finding $\hat{X}$, we apply inverse fourier transform to get the estimated image $\hat{x}$.

5 Numerical Results

In this chapter, we will show the numerical results of deblurring.

The environment of process:

CPU : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz

RAM : 8.0GB

OS : Ubuntu 20.04.3 LTS

Programming Language : Python 3.7 & MATLAB R2020a

We have three images and five blur kernels for estimation.

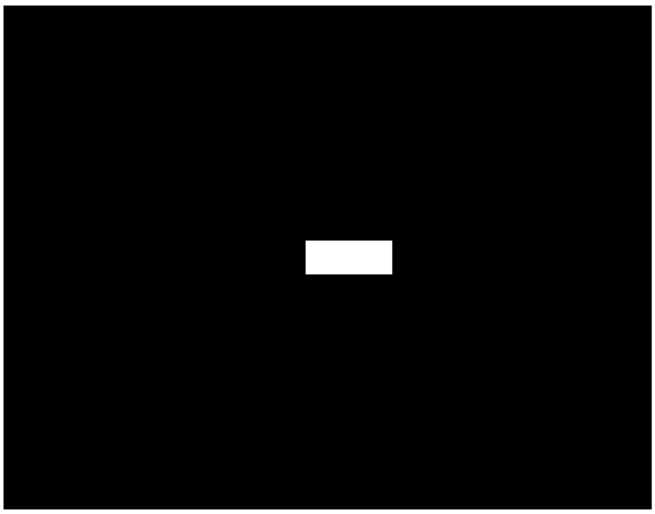
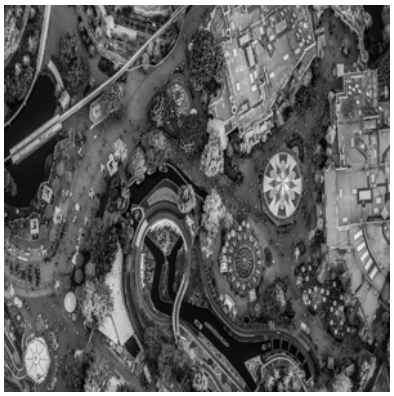


Figure 4: The initial kernel will always set to be a delta kernel

The three test images are:

(a) The origin image (1)



(b) The origin image (2)

(c) The origin image (3)

Figure 5: The test images

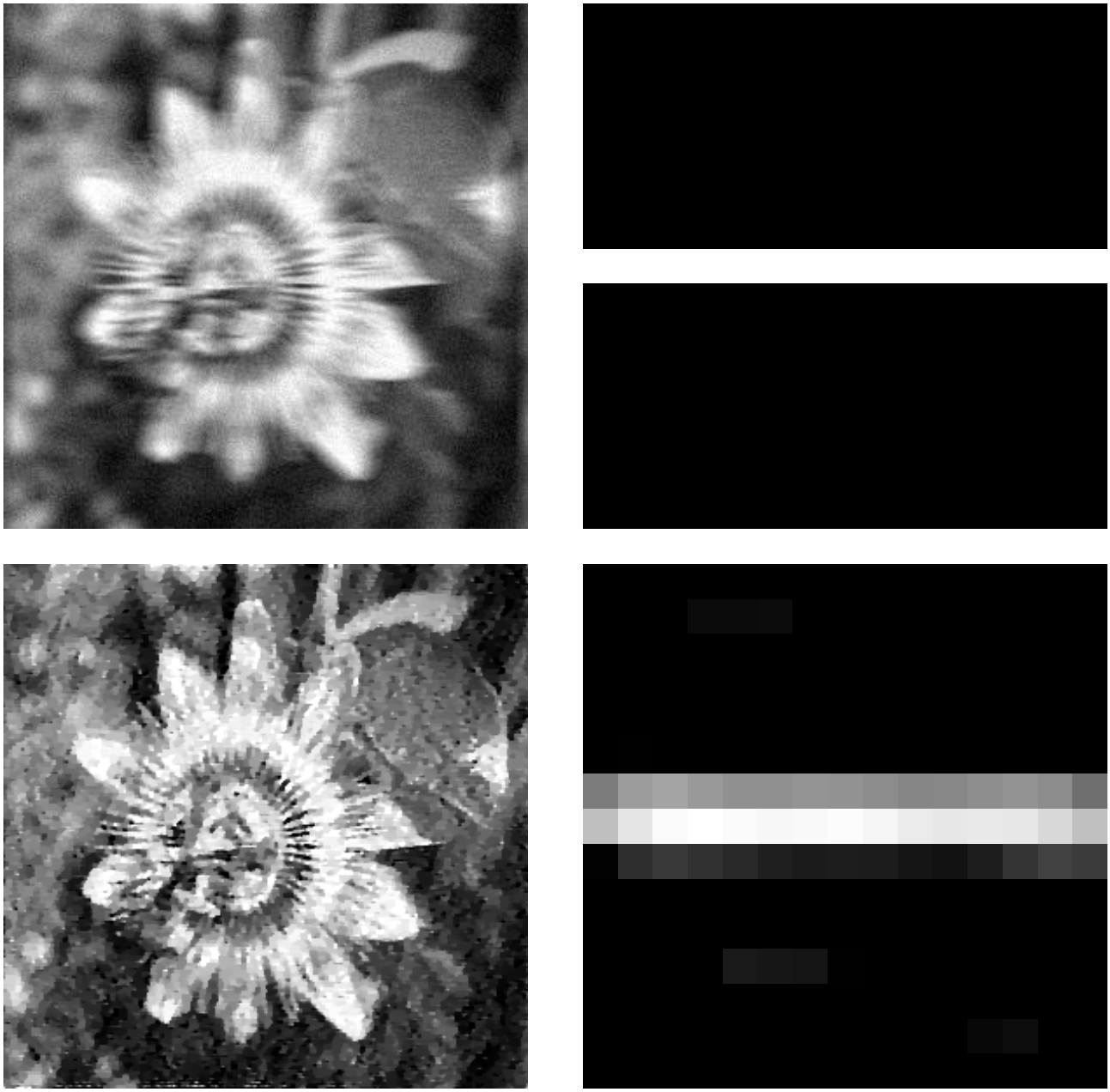


Figure 6: **(a)(b)**. Blur image(1) with the horizontal kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

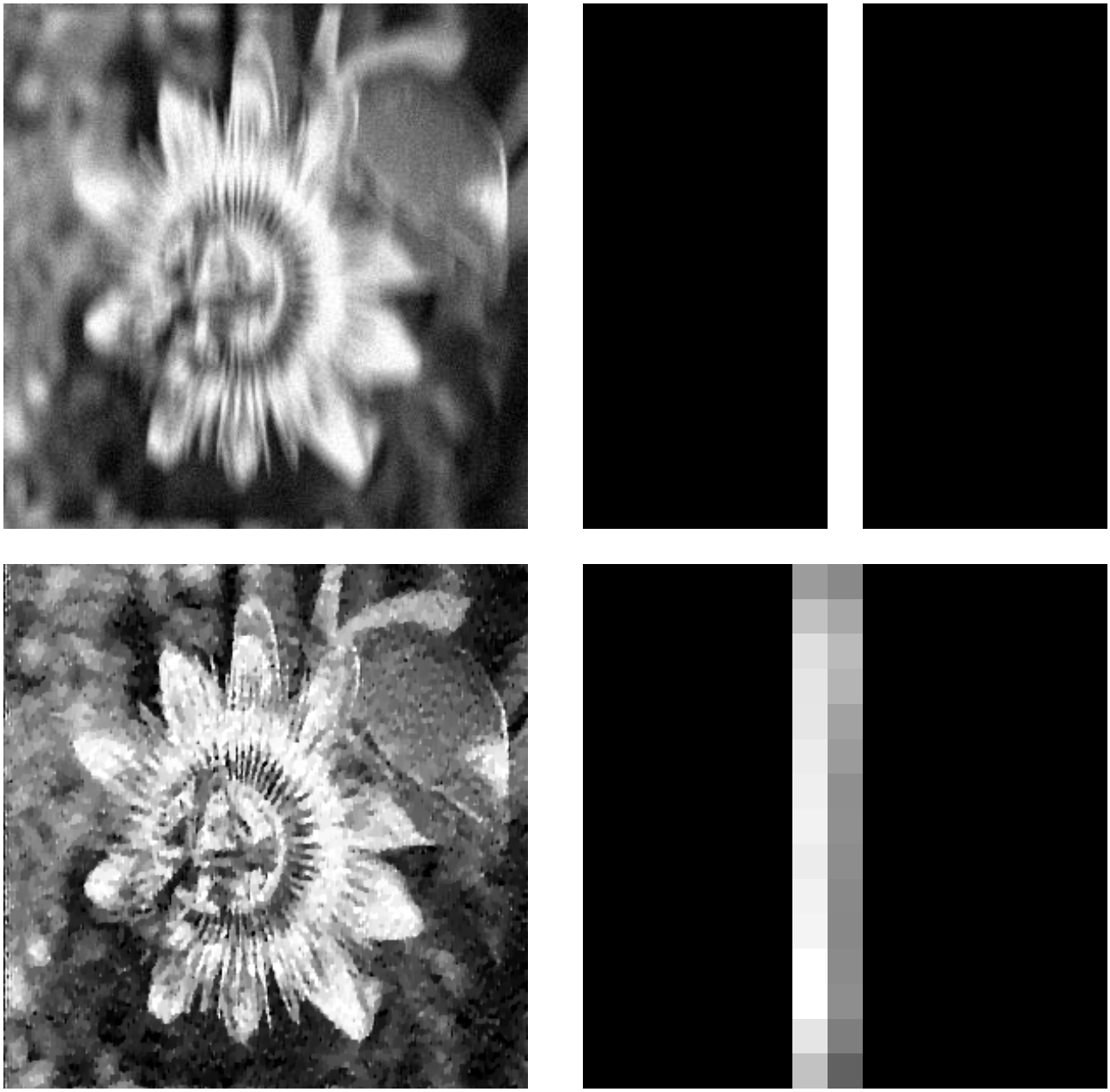


Figure 7: **(a)(b)**. Blur image(1) with the vertical kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

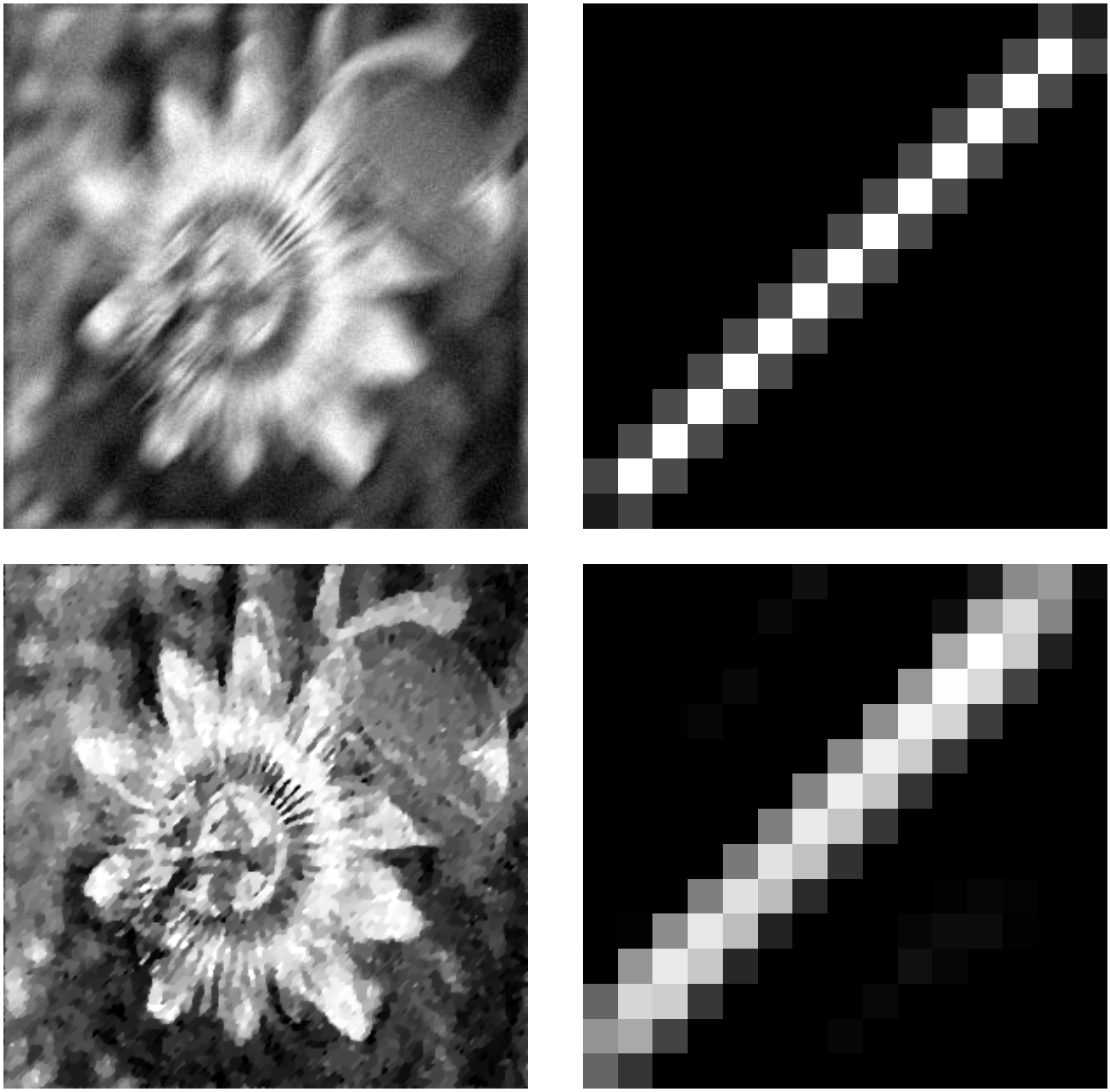


Figure 8: **(a)(b)**. Blur image(1) with the 45 degree line kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

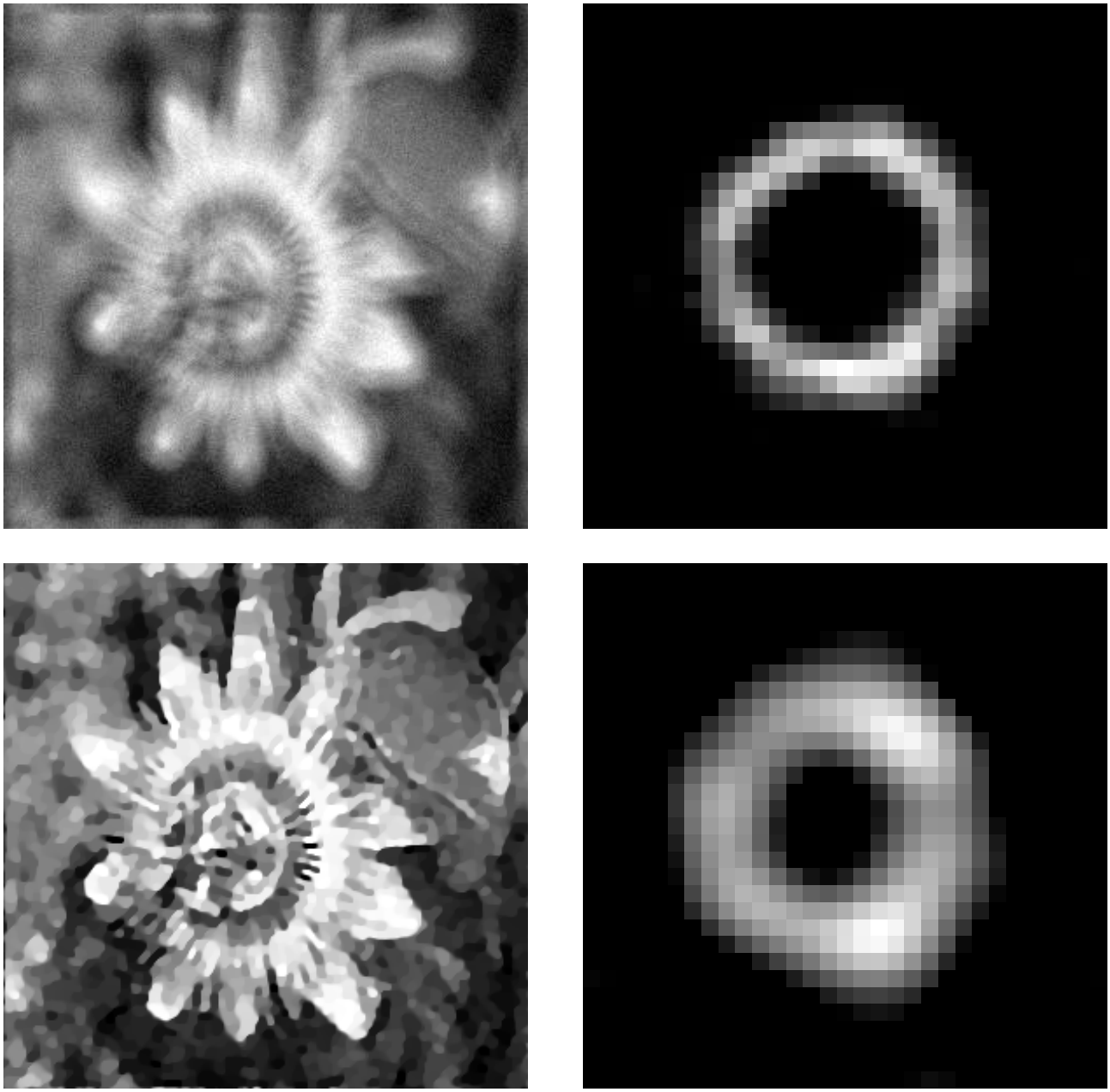


Figure 9: **(a)(b)**. Blur image(1) with the ring kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

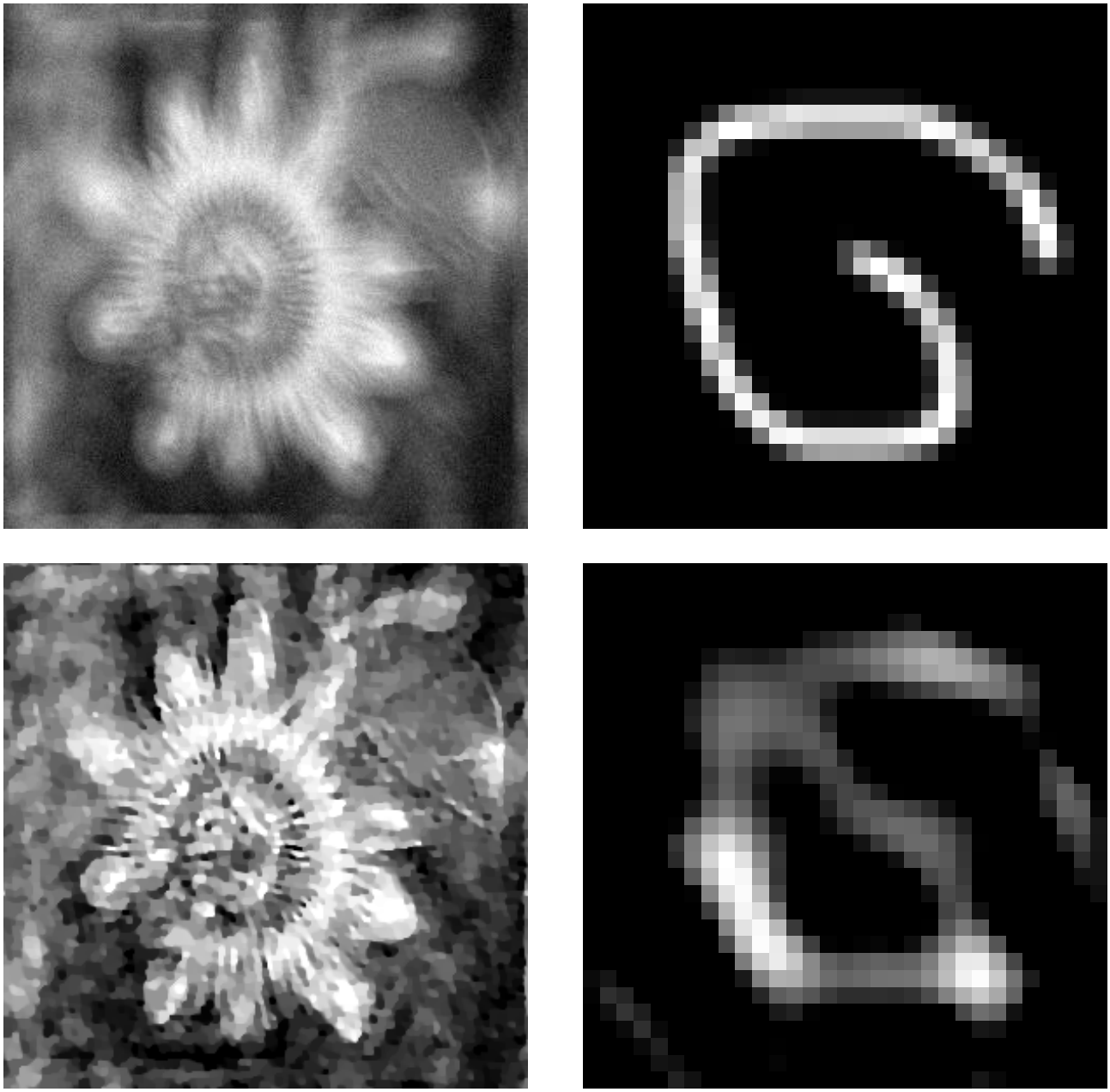


Figure 10: **(a)(b)**. Blur image(1) with a curve like kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

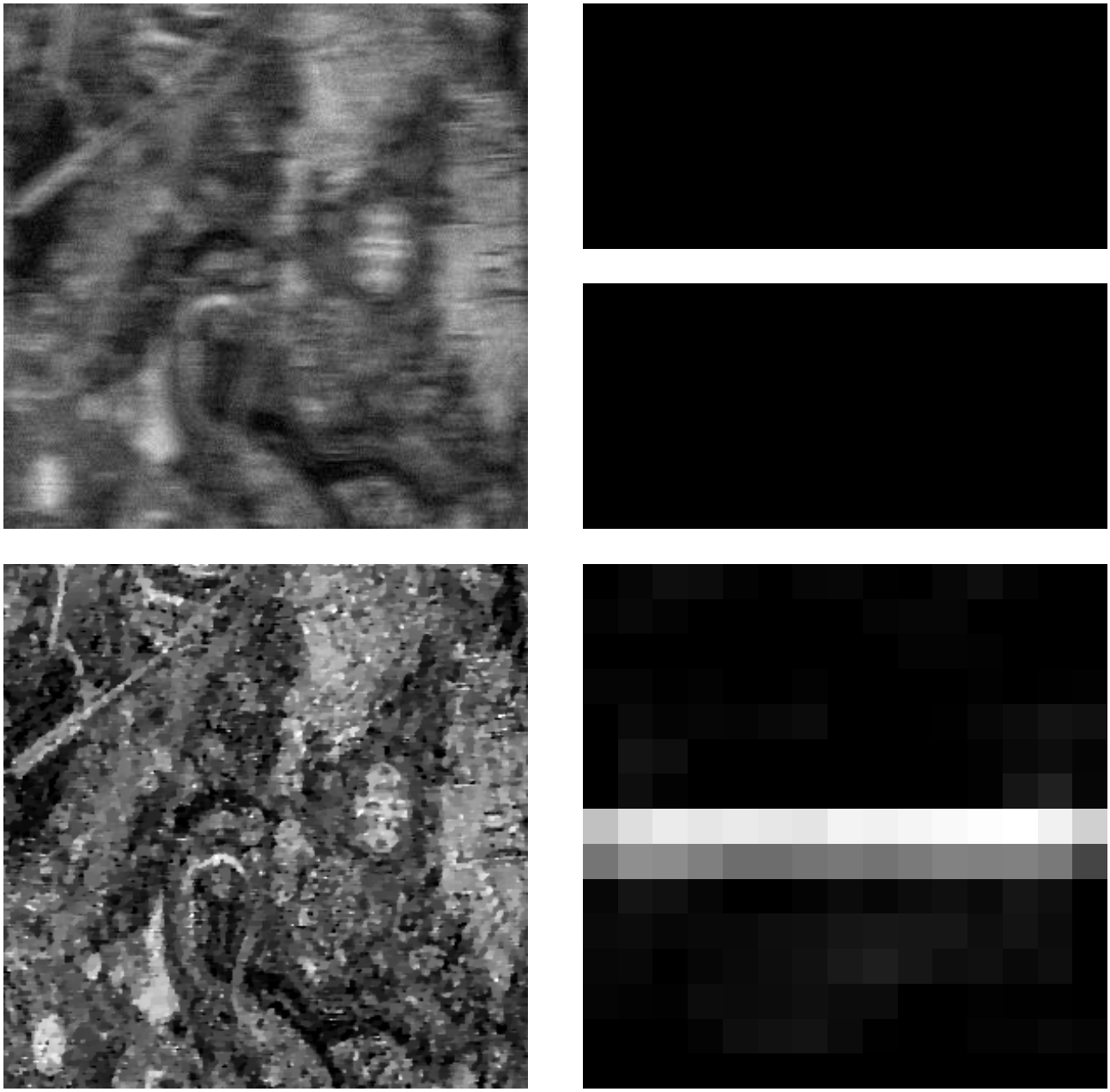


Figure 11: **(a)(b)**. Blur image(2) with the horizontal kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

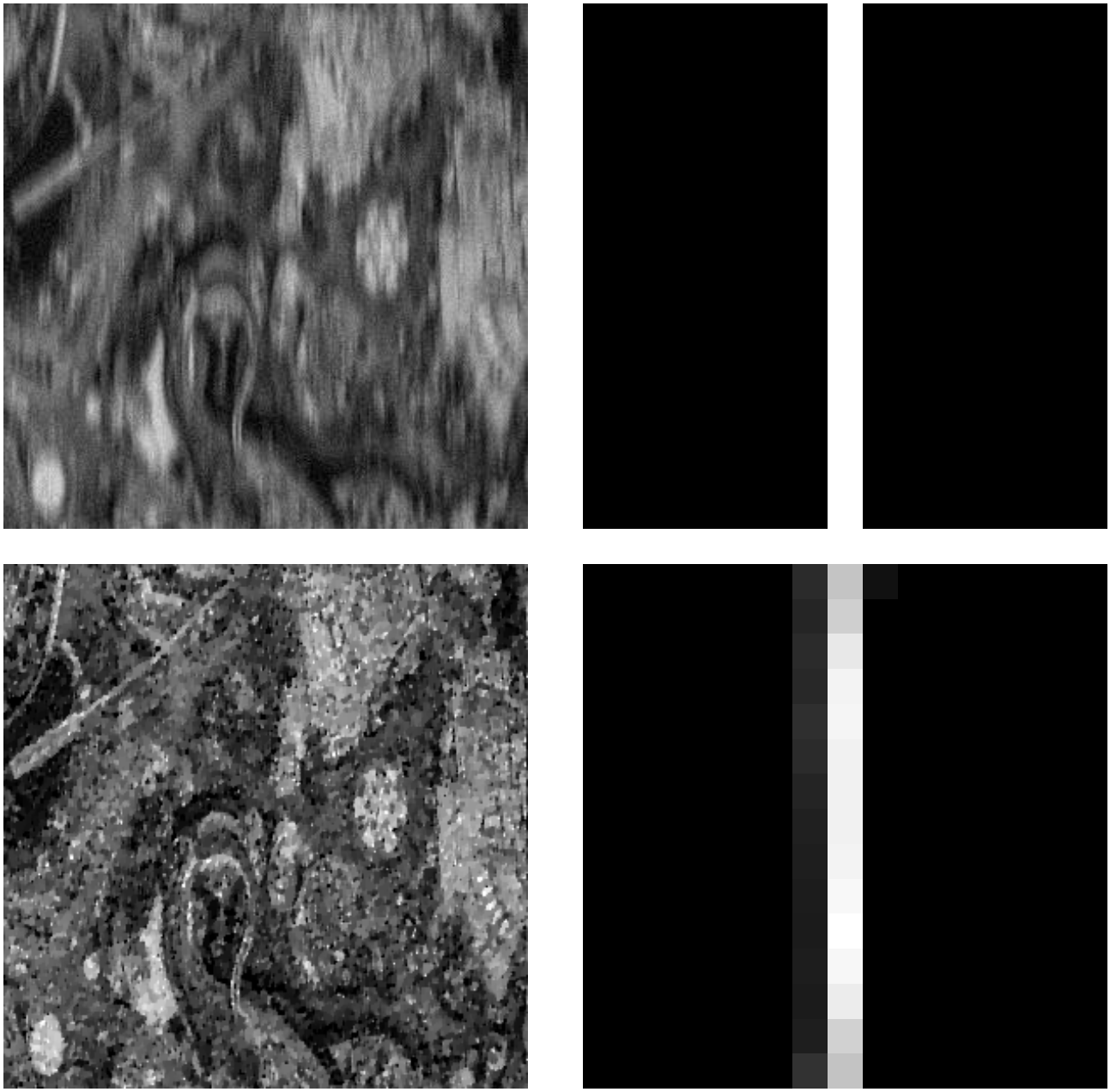


Figure 12: **(a)(b)**. Blur image(2) with the vertical kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

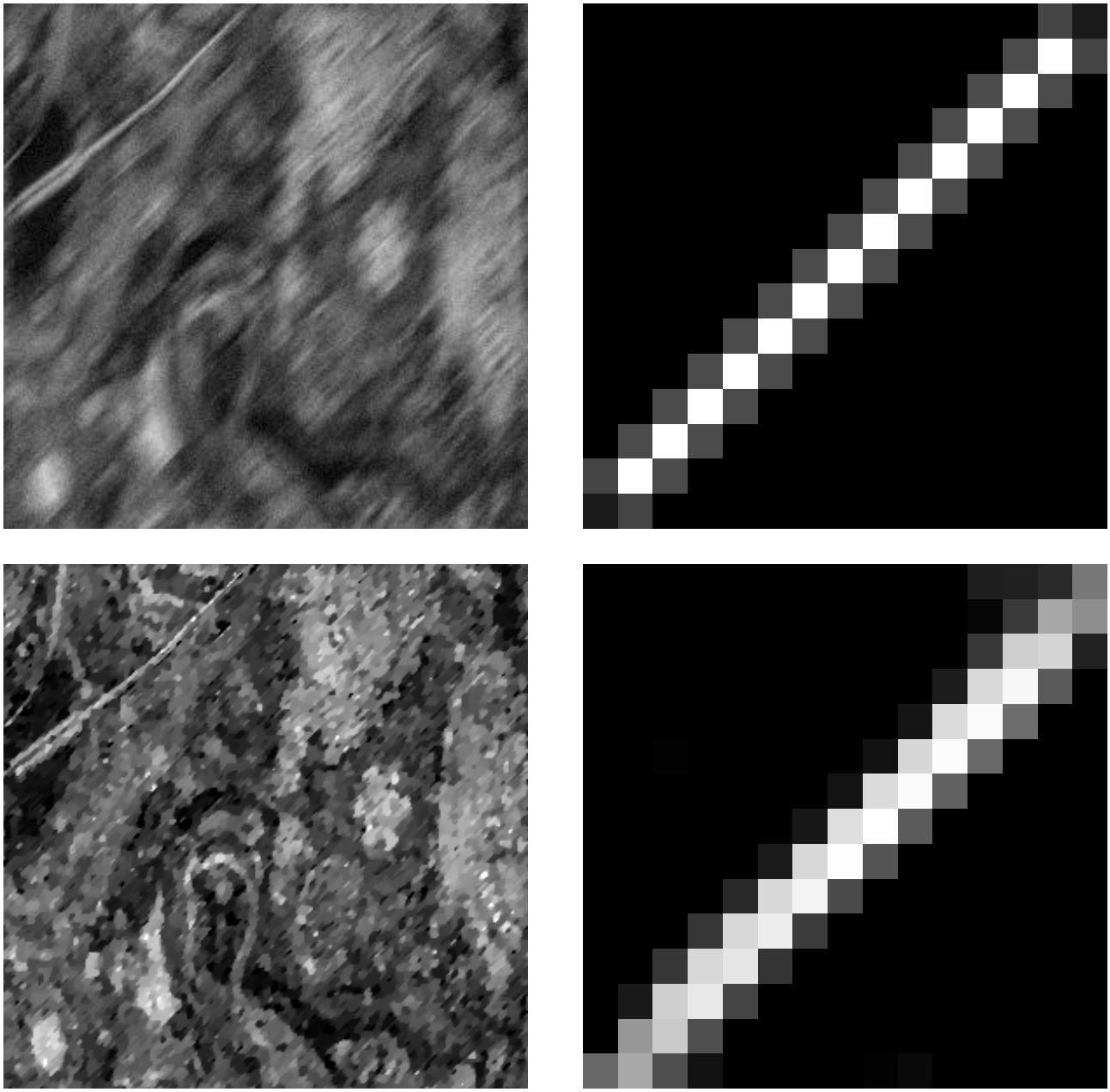


Figure 13: **(a)(b)**. Blur image(2) with the 45 degree line kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

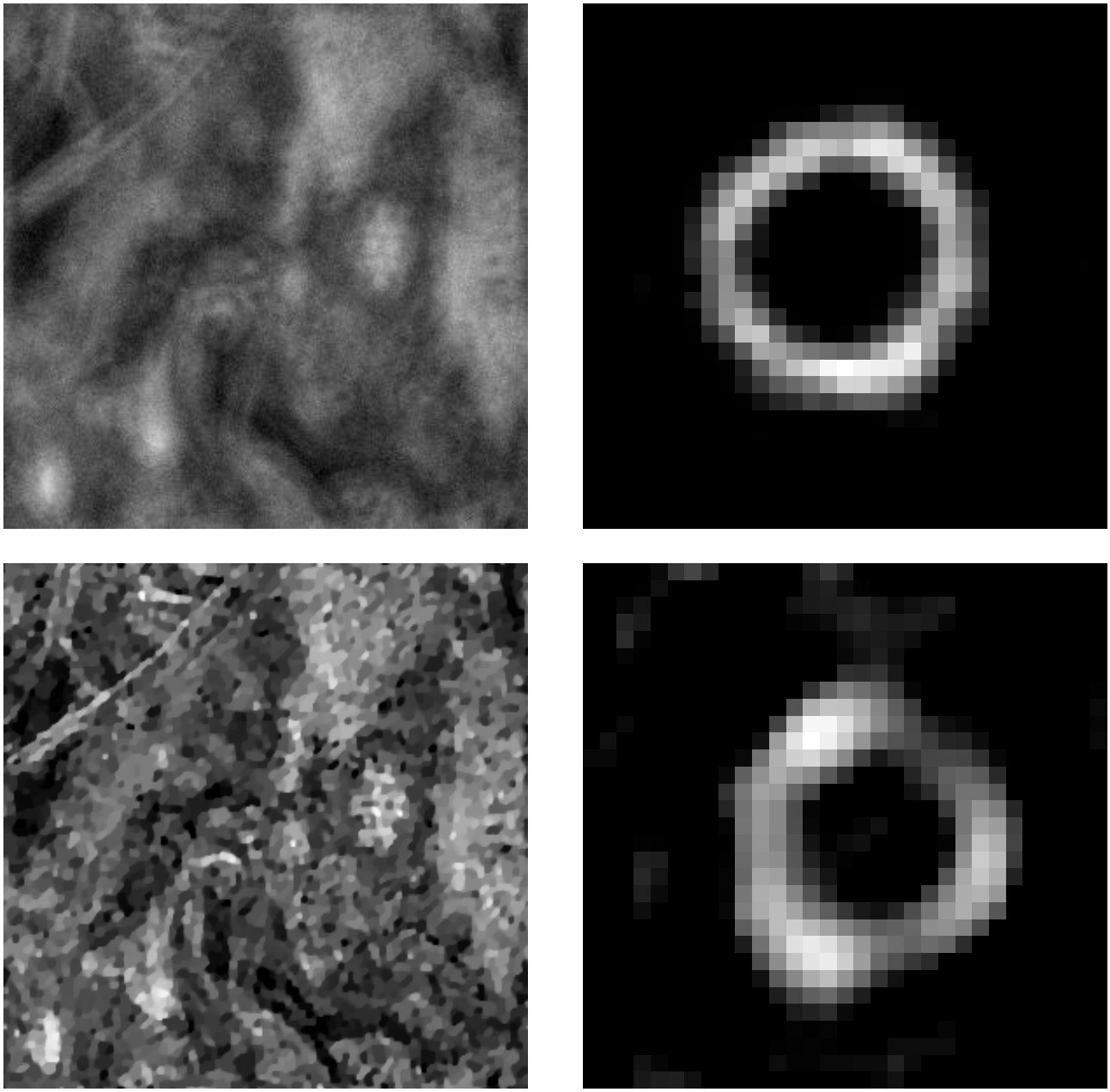


Figure 14: **(a)(b)**. Blur image(2) with the ring kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

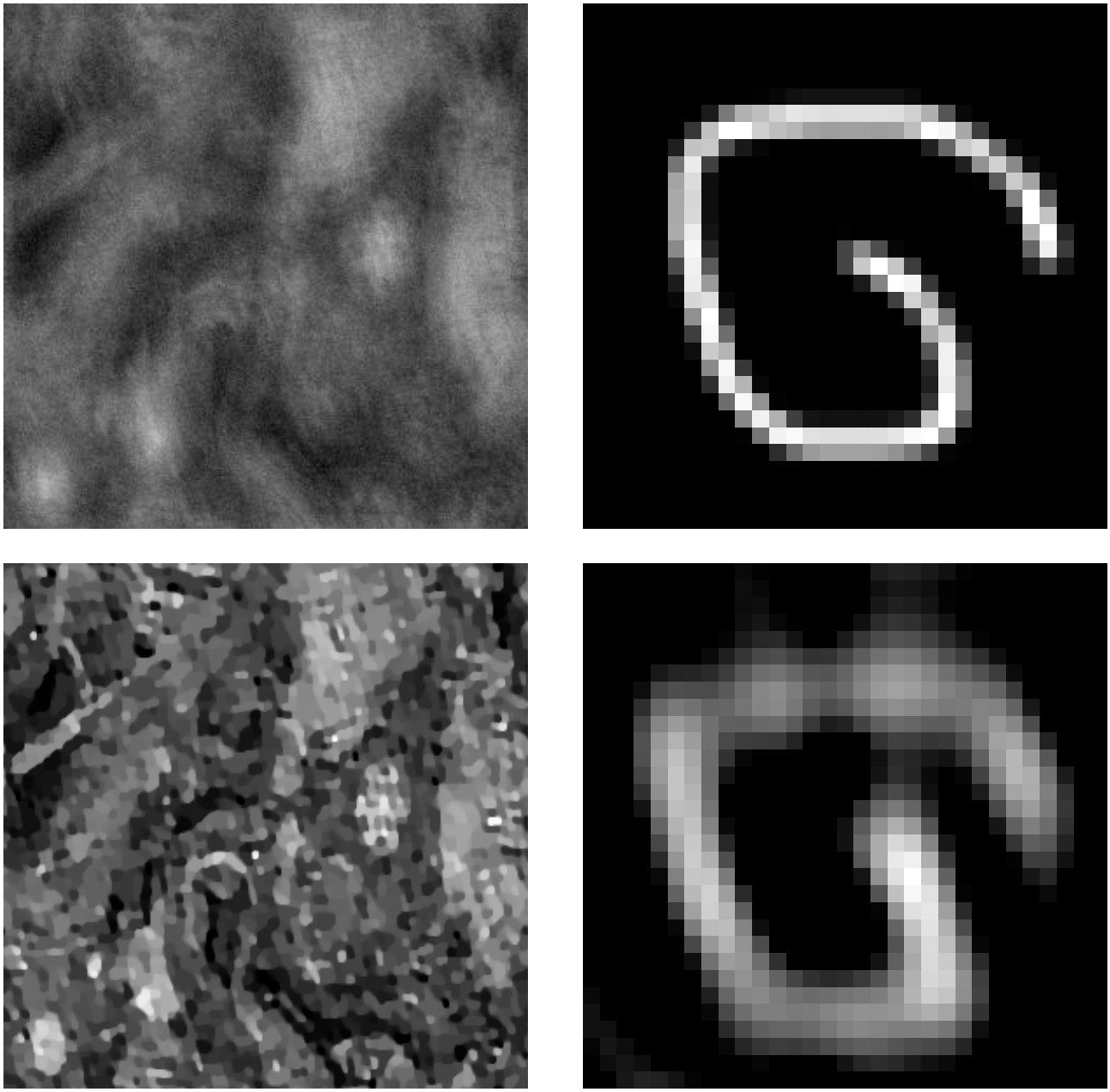


Figure 15: **(a)(b)**. Blur image(2) with a curve like kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

Figure 16: **(a)(b)**. Blur image(3) with the horizontal kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

Figure 17: **(a)(b)**. Blur image(3) with the vertical kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

Figure 18: **(a)(b)**. Blur image(3) with the 45 degree line kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

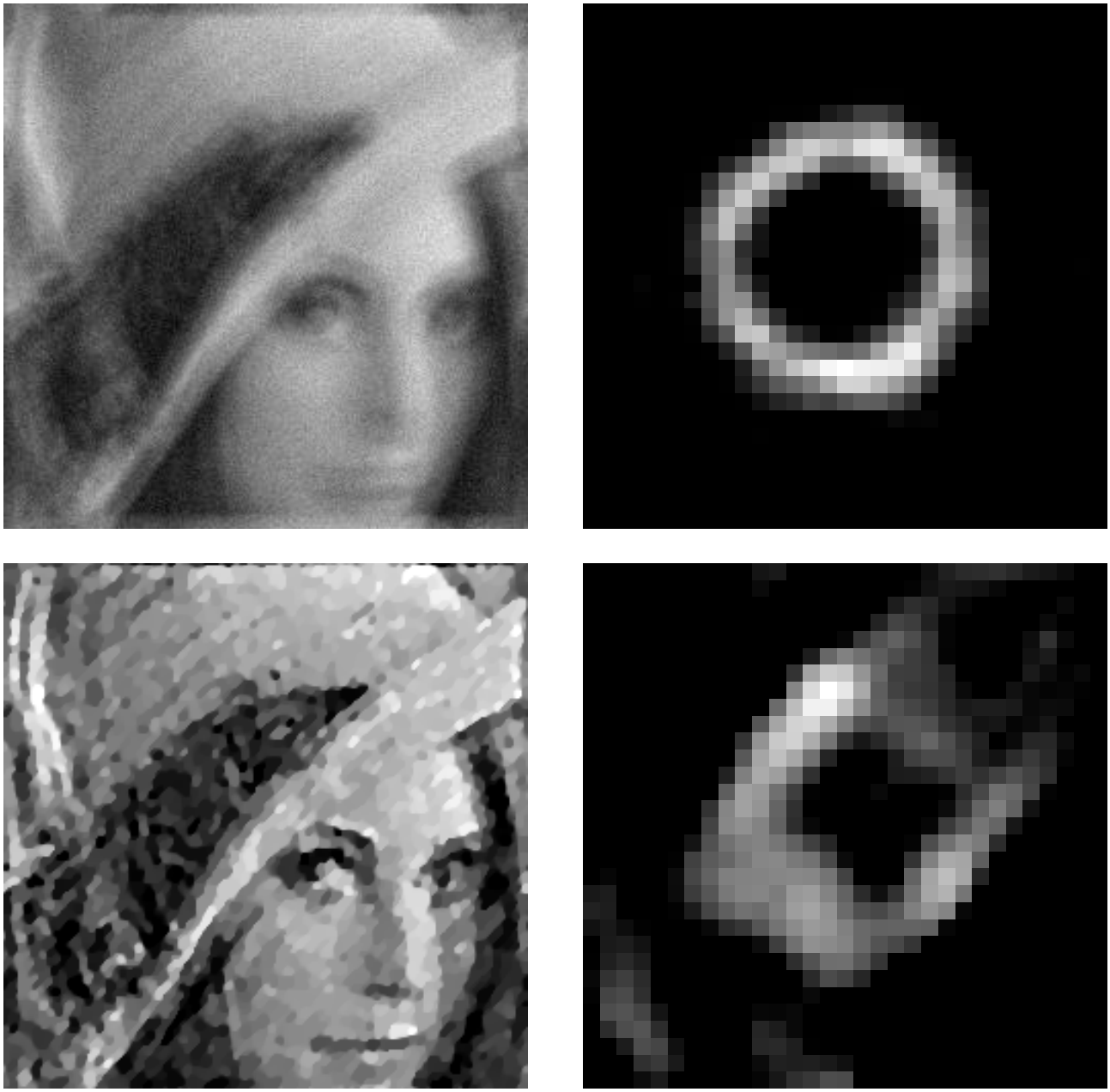


Figure 19: **(a)(b)**. Blur image(3) with the ring kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

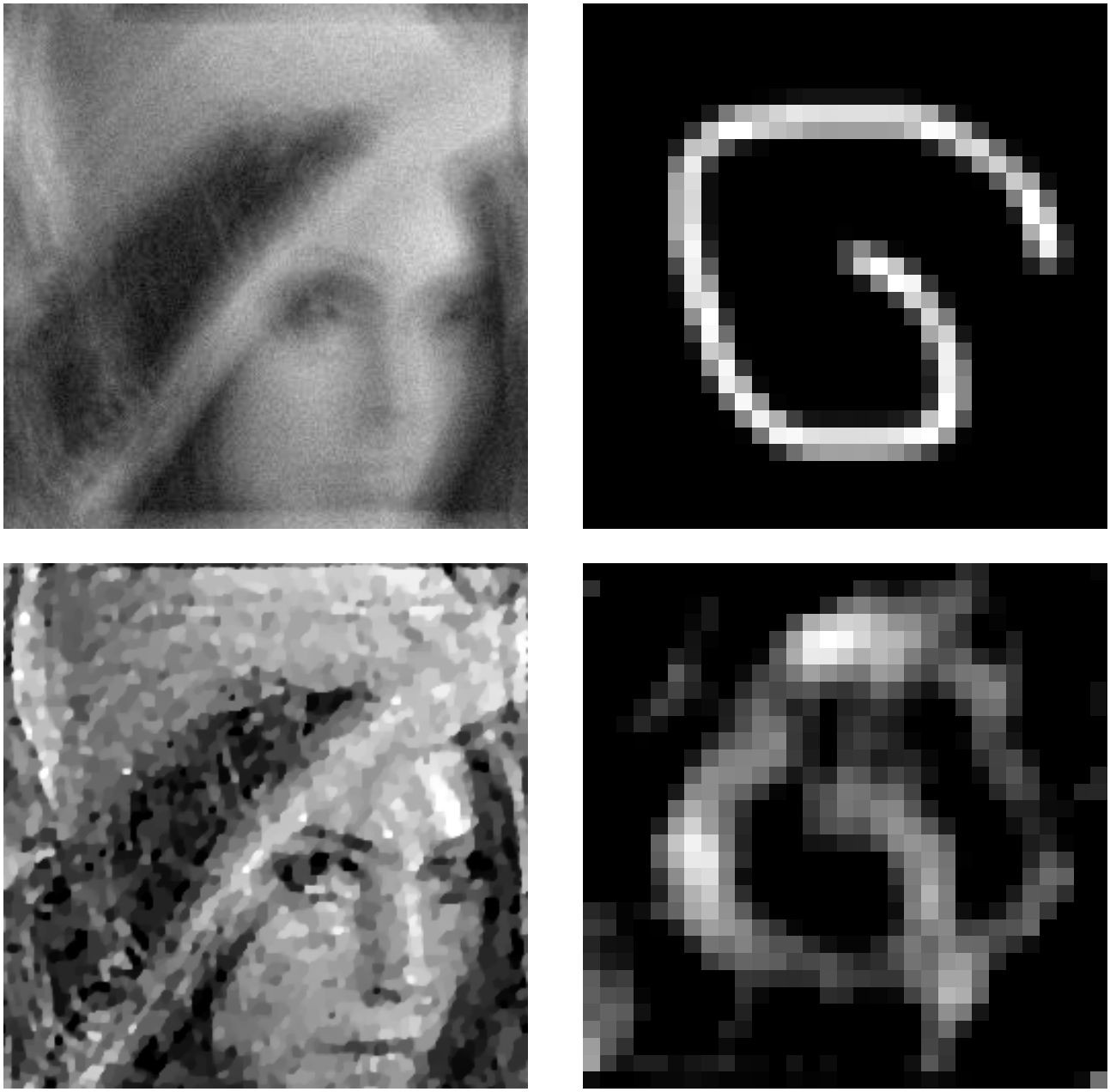


Figure 20: **(a)(b)**. Blur image(3) with a curve like kernel and add gaussian noise. **(c)(d)**. Corresponding deblur image of (a) with the estimated kernel.

6 Conclusions

This thesis introduces a method to estimate the blur kernel and deblur the image. The main method is closed related to maximum a posterior with the prior being sparsity of gradient histogram. We start from delta-like kernel and get a reasonable estimated kernel. The hard part is deteeminig the noise variance and the variance of gradient histogram. These two variables greatly influence the result. And we also found that different region in a same image may need different variances. Another observation is that when the image is rich in edges, the estimated kernel tend to be more similar to the true kernel, while when an image contain large part of saturated region, the result may not be satisfying. In the future we hope to combine the power of this method with recent mechine learning research. Our goal is to contruct an estimated kernel from a naive initial kernel. And use the estimated kernel as a constraint in deblurGAN[6] or CycleGan[15]. Since most mechine learning method for deblurring suffer from lacking of data, we think that the estimated kernel will provide great information and guide the network to better result without worrying about the efficiency of data. There are also network based EM algorithm for blind deconvolution [10][9], they convert the process of MAP_k to a variational based network. This may also shed light on our future work. Hopefully we can discover more in the future.

References

[1] C. M. Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[3] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. pages 787–794, 2006.

[4] D. J. Field. What is the goal of sensory coding? *Neural computation*, 6(4):559–601, 1994.

[5] R. C. Gonzalez and R. E. Woods. Digital image processing 4th edition, global edition. 2018.

[6] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8183–8192, 2018.

[7] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. pages 1964–1971, 2009.

[8] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. pages 2657–2664, 2011.

[9] Y. Li, M. Tofighi, V. Monga, and Y. C. Eldar. An algorithm unrolling approach to deep image deblurring. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7675–7679. IEEE, 2019.

[10] Y. Nan, Y. Quan, and H. Ji. Variational-em-based deep learning for noise-blind image deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3626–3635, 2020.

[11] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[12] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. pages 157–170, 2010.

[13] L. Xu, S. Zheng, and J. Jia. Unnatural l0 sparse representation for natural image deblurring. pages 1107–1114, 2013.

[14] J. Yang, Y. Zhang, and W. Yin. An efficient tvl1 algorithm for deblurring multichannel images corrupted by impulsive noise. *SIAM Journal on Scientific Computing*, 31(4):2842–2865, 2009.

[15] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.