

Depth Estimation Using Segmentation in Natural Images

Shrayan Roy, Roll No: MD2220

Project Supervisor: Dr. Deepayan Sarkar*

Abstract

Estimating depth map from a single photograph is an intriguing problem with many interesting practical applications. Levin et al [4] proposed a method which requires a modified camera with a specialized coded aperture. However, this approach is not applicable in practice since it requires modifying the camera before capturing the image, which may not always be feasible. We aim to explore whether similar tasks can be done using standard cameras. We have formulated simple parametric models for depth blurring, which we hope will lead to more efficient estimates.

Keywords: Auto Regressive Prior, Blur Kernel, Depth Map, Image Segmentation

1 Introduction

Traditional photographs are two dimensional projections of a three dimensional scene. The third dimension is *depth*, which represents the distance between camera lens and objects in the image. Depth perception is crucial for understanding spatial relationships, with various applications in computer vision tasks. Additionally, photography and cinematography benefit from depth perception, aiding in the creation of visually compelling compositions.

Most methods to estimate depth involve analyzing multiple images of the same scene to measure pixel sharpness across the stack and determine depth based on the distance from the sharpest pixel [2]. Hardware-based solutions are also available, usually involving extra apparatus such as light emitters.

Depth estimation based on single image is a more challenging problem because we have single observation for each pixel of the scene. Depth estimation from defocus blur exploits the phenomenon where objects appear more blurred depending on their distance from the camera lens, serving as a depth surrogate. Levin et al. [4] utilized this idea with a *sparse gradient prior* on natural images to estimate the *level of blur* per pixel. However, this method requires a modified

*Theoretical Statistics and Mathematics Unit, ISI Delhi < deepayan@isid.ac.in >

camera with a special coded aperture. Zhu et al. [11] employed Gabor filters for local frequency component analysis and utilized a *simple gradient prior*, without the need for a special coded aperture. After estimating depth for each pixel, an energy minimization technique based on Markov Random Field over the image is used to generate a smooth depth map.

In this project, we will explore whether a somewhat different approach can be used for the same problem. Instead of using a MRF approach for depth segmentation, we plan to start with modern high-performance segmentation algorithms such as Segment Anything [3]. Our hope is that if blurring is uniform within each segmented portion, it will be easier to estimate the blur kernel without a special coded aperture. We follow the approach developed by Nandy [6] to estimate the blur kernel, using in particular the locally dependent gradient prior proposed by them. However, instead of their nonparametric approach, we formulate and use simple parametric models for depth blurring, which we hope will lead to more efficient estimates for smaller image segments.



Figure 1: Example of Depth Map from [4]. Original image (Left panel) and corresponding Depth map (right panel)

2 Mathematical Formulation

When light rays spread from a single point source and hit the camera lens, they should ideally get refracted and converge on the pixel corresponding to the original scene. However, if the source is out of focus, the refracted rays spread out over neighboring pixels as well. This spreading pattern, determined by the object's distance from the lens or camera movement, is called the *Point Spread Function* (PSF) or *Blur Kernel*. The blurred image can be viewed as the result of convolving the original sharp image using the PSF. If we assume that the scene remains static for the duration of the photograph and there is no significant camera shake or rotation, then the observed blurred image \mathbf{b} of dimension $M \times N$ can be modeled as:

$$\mathbf{b} = \mathbf{k} \otimes \mathbf{l} + \epsilon \quad (1)$$

Where \mathbf{k} is an $m \times n$ blur kernel, \mathbf{l} is the $(M+m) \times (N+n)$ *true latent image* which we want to

estimate, ϵ is an $M \times N$ matrix of noise, and \otimes denotes the *convolution* operator (By *convolution* we mean *valid convolution*). Reconstructing the latent image from an observed blurred image is called the **image deconvolution problem**. Depending on whether the blur kernel is known or unknown, we classify it as *non-blind deconvolution* or *blind deconvolution* problem. In either case, it is an ill-posed problem because the number of parameters is larger than the number of observations MN . One solution to this is to assume a prior for the latent image \mathbf{l} .

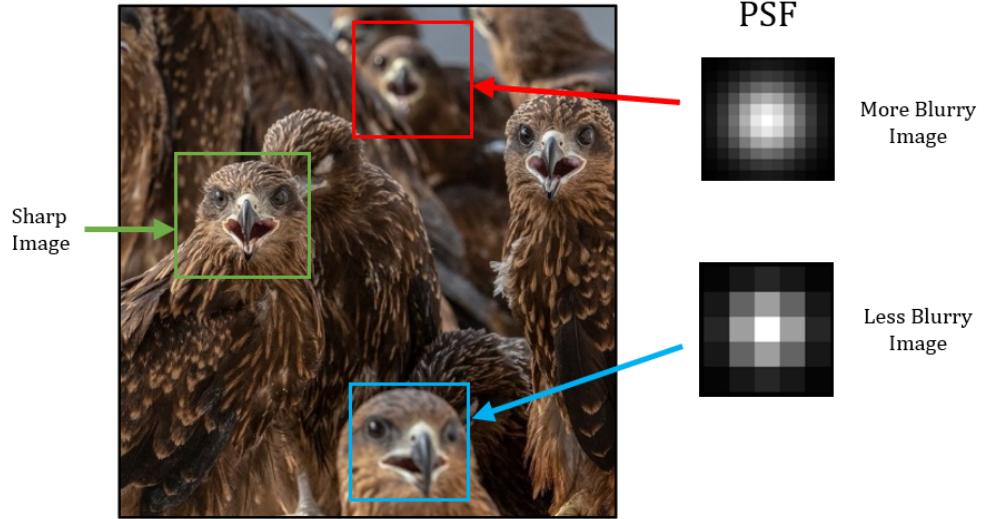


Figure 2: Example of Spatially Varying Blur: Each patch exhibits a distinct level of blur.

Note that, the model defined above assumes that the associated PSF is *shift invariant*, meaning the same PSF applies to all pixels of the underlying latent image. However, this may not be the case (Figure 2). In the context of depth from defocus blur, the PSF function is *not* shift invariant i.e. it is *spatially varying*. Therefore, (1) will not hold. We redefine it as:

$$\mathbf{b}[\mathbf{t}] = (\mathbf{k}_t \otimes \mathbf{l})[\mathbf{t}] + \epsilon[\mathbf{t}] \quad (2)$$

Where $[\mathbf{t}]$ indicates the corresponding elements at pixel location \mathbf{t} and \mathbf{k}_t is the spatially varying blur kernel at pixel location \mathbf{t} . Now, the problem of estimating blur kernel and latent image becomes more ill-posed because for each pixel, we need to estimate a blur kernel. However, if the blurring is only due to the objects being away from the plane of focus, we can assume special structures of the associated blur kernels. We model the blur kernel \mathbf{k}_t as some probability density over a square grid. For each pixel location \mathbf{t} , we characterize the blur kernel by the parameter θ_t that determines the *scale* or *spread* of the associated probability distribution (e.g. scale parameters in bivariate normal distribution). This parameter θ_t encompasses information regarding the level of blur, hence providing insight into depth. Our objective is to estimate this parameter θ_t based on the observed blurred image \mathbf{b} for each pixel location \mathbf{t} .

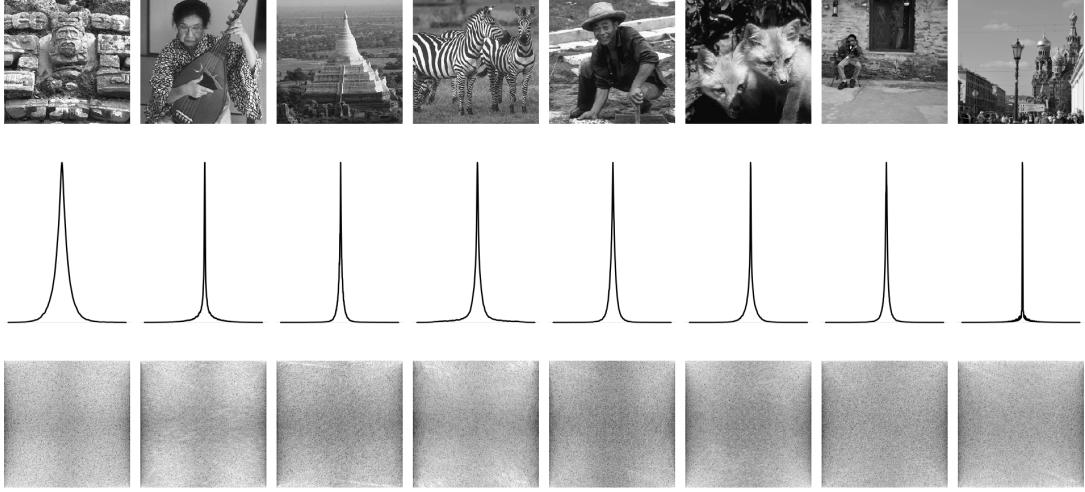


Figure 3: Eight sharp images (top row) and their density plot of horizontal gradients (middle row) and plot of log modulus DFT coefficients of horizontal gradients (bottom row). Similar observations are found for vertical gradients as well.

3 Priors on Natural Images

3.1 Choice of Prior and Basic Ideas

The prior family used for the latent *natural image*¹ is motivated by the observation that when a gradient filter is applied to image, the distribution of the output has a consistent and distinctive form across a wide range of scene types, with the distribution sharply peaked at zero and relatively heavier tails than the Gaussian distribution and Laplace distribution (Figure 3). Priors with these features are often referred to as *sparse priors* and a useful parametric family to model this is the so called **Hyper-Laplacian Distribution** given by

$$f_\alpha(z) = \frac{\alpha}{2\Gamma(\frac{1}{\alpha})} \exp(-|z|^\alpha), z \in \mathbb{R} \text{ and } \alpha > 0 \quad (3)$$

For $\alpha = 2$ we have Gaussian distribution and for $\alpha = 1$ we have Laplace distribution. Levin et al. [4] used $\alpha = 0.8$, while Zhu et al. [11] utilized $\alpha = 2$, assuming IID gradients. Nandy [6] showed that the assumption of independent image gradients is incorrect, and modeled the dependency structure of gradients using a simple first-order AR Model.

To apply these priors, we must express the blur model in terms of image gradients, specifically in frequency domain. We will do this for (1). If $\delta_h = [-1, 1]$ and $\delta_v = [-1, 1]^T$, then the horizontal and vertical gradients are given by

$$\delta_h \otimes b = \delta_h \otimes (k \otimes l) + (\delta_h \otimes \epsilon) = k \otimes (\delta_h \otimes l) + (\delta_h \otimes \epsilon) \quad (4)$$

¹By *natural*, we refer to typical scenes captured in amateur digital photography, excluding specialized contexts like astronomy or satellite imaging.

$$\delta_v \otimes b = \delta_v \otimes (k \otimes l) + (\delta_v \otimes \epsilon) = k \otimes (\delta_v \otimes l) + (\delta_v \otimes \epsilon) \quad (5)$$

To keep the notations simple, we will henceforth represent both (4) and (5) in the generic form

$$y = k \otimes x + n, \quad (6)$$

where y is the horizontal (or vertical) gradient of observed blurred image, and x and n are the same for latent image and noise. By virtue of the *Convolution Theorem*, we can rewrite (6) in the frequency domain as

$$Y = K \odot X + N, \quad (7)$$

where Y, K, X and N are the *Discrete Fourier Transform*'s of y, k, x and n respectively, and \odot indicates the *element wise product* operator. For all $\omega = (\omega_1, \omega_2)$ we have

$$Y_\omega = K_\omega X_\omega + N_\omega. \quad (8)$$

Remark: In practice the size of the blur kernel k is much smaller compared to the latent image l and x . But for (7), the size of K must be the same as the size of X . Thus, we pad k symmetrically with zeros to make of same size as x and then take DFT.

3.2 Prior on Fourier coefficients

If the elements of x are zero mean IID random variables with pdf (3), then by orthogonality of DFT, X_ω 's must be IID complex normal when $\alpha = 2$. Even for $\alpha \neq 2$, X_ω 's are uncorrelated and asymptotically complex normal by the CLT of Peligrad and Wu [8]. When elements of x are correlated, X_ω 's are still asymptotically independent and complex normal; however, depending on the correlation structure, the variance is no longer constant and depends on the specific frequency ω . We will use either the IID prior, or the simple AR prior of Nandy [6], for which $Var(X_\omega) = g_\omega \sigma^2$ with the form of g_ω known explicitly.

For spatially varying blur, it is not immediately clear how we can express (2) in terms of image gradients. If we assume that the blur kernel k_t is shift invariant in a neighborhood η_t of size $p_1(t) \times p_2(t)$ containing t , then we get equation (9) as blur model.

$$y[t'] = (k_t \otimes x)[t'] + n[t'] \quad \forall t' \in \eta_t \quad (9)$$

In practice, we expect this assumption to be approximately true if the patch size is small, because we expect objects in small local patches to have same depth and hence same level of blur (Figure 2). We can then apply the priors discussed above to these specific patches of the image.

4 Parametric Models for Blur Kernel

From a single point source, light rays emit in different directions and fall on the lens of camera. The diffracted rays by the lens then form a circular shape on the camera sensor plane, which is called the *Circle of Confusion* or *Blur Circle* (Fig 4). The diameter of the blur circle (c_{diam}) and the depth of an object (d) in a given camera setting are related as follows [1]:

$$c_{diam} = a_{diam} f \left| \frac{d - d_{focus}}{d(d_{focus} - f)} \right| \approx a_{diam} f \left| \frac{1}{d_{focus}} - \frac{1}{d} \right| \quad (10)$$

In a given camera setting (i.e., fixed a_{diam} and f),² $c_{diam} \propto \left| \frac{1}{d_{focus}} - \frac{1}{d} \right|$. It is evident from (10) that, the support of PSF in our case must be circular rather than square. Due to *diffraction* caused by the camera lens and the boundaries of the circular aperture, we expect the intensity distribution of light to be spherical symmetrically distributed over the circular support. Keeping all these in mind we propose the following models for blur kernel.

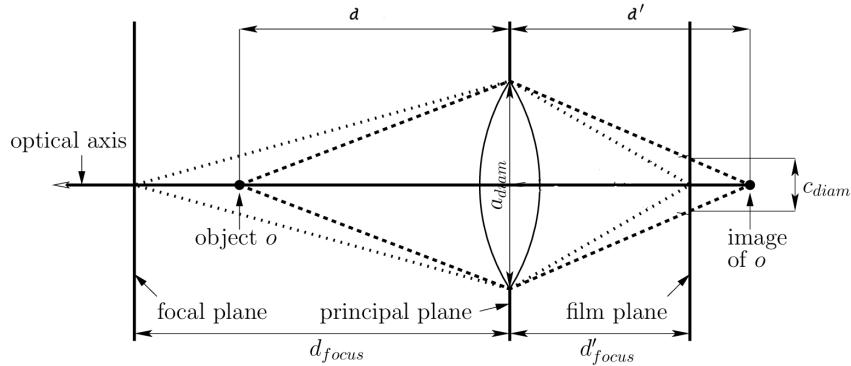


Figure 4: Thin Lens Model of Camera and Circle of Confusion c_{diam} in a given camera setting

1. Disc Kernel: This is the simplest blur model, which assumes uniform spread of light over a circular disc. We characterize the kernel using the parameter r , which corresponds to $c_{diam}/2$ in (10).

$$k(x, y) = \frac{1}{\pi r^2} \times I_{\{x^2 + y^2 \leq r^2\}}$$

2. Circular Gaussian Kernel: Another standard choice of kernel the widely used Gaussian/Normal distribution. Here, we will consider a truncated version of it over a circular region. We characterize the kernel using the radius of circle r and scale parameter h .

$$k(x, y) = \frac{C_{h,r}}{2\pi h^2} e^{-\frac{x^2+y^2}{2h^2}} \times I_{\{x^2 + y^2 \leq r^2\}}$$

3. Circular Cauchy Kernel: We know that the intercept on the x-axis of a beam of light coming from the point $(0, h)$ under certain assumption is distributed as a $\text{Cauchy}(0, h)$ distribution.

²For most cameras, $d_{focus} \gg f$, hence the approximation.

Extending this concept to the two-dimensional case, we have a bivariate Cauchy distribution over a circular support. We characterize this using the radius of the circle r and the scale parameter h .

$$k(x, y) = \frac{C_{h,r}}{2\pi} \frac{h}{(x^2 + y^2 + h^2)^{3/2}} \times I_{\{x^2 + y^2 \leq r^2\}}$$

Remark: The parameter r in both the Circular Gaussian and Circular Cauchy kernel characterizes the radius of the blur circle (i.e., $c_{diam}/2$). For a given camera setting, there exists a relation between h and r , namely $h = \kappa \times r$ with κ depending upon particular camera. This implies that we cannot change h and r independently.

Note that each value of c_{diam} corresponds to two possible values of the depth d , either closer to or further away from the camera relative to the focal plane. Thus, from the diameter of the blur circle, which is the parameter we estimate, it is impossible to distinguish between these two possibilities.

5 Maximum Likelihood Estimation of Blur Kernel Parameters

Our main focus in this section is to develop *Maximum Likelihood Estimation* procedure for the parameter θ_t of blur kernel for each pixel location t . For the first three kernels $\theta_t = (r_t, h_t)$, and for the Gaussian kernel $\theta_t = h_t$. We need to estimate these parameters based on the observed image \mathbf{b} or equivalently \mathbf{y} . For that we need joint distribution of elements of \mathbf{y} .

As our priors for \mathbf{x} are defined in terms of Fourier coefficients, we move to the frequency domain. In the case of generalized prior we have $\mathbf{X}_\omega \sim \mathcal{CN}(0, \sigma^2 g_\omega) \quad \forall \omega$ independently. In addition, if the errors in the original image, given by ϵ , are assumed to be IID Gaussian, then its gradient \mathbf{n} will have correlated elements and successive elements in the direction of the gradient will have correlation 0.5, while all other pairs will be uncorrelated. This will induce a non-constant variance for \mathbf{N}_ω , given by a function h_ω such that $Var(\mathbf{N}_\omega) = \eta^2 h_\omega$. Hence, $\mathbf{N}_\omega \sim \mathcal{CN}(0, \eta^2 h_\omega)$ independently. An explicit expression for h_ω can be found similarly to that of g_ω .

From (8) we have, $\mathbf{Y}_\omega \sim \mathcal{CN}(0, \sigma^2 |K_\omega|^2 g_\omega + \eta^2 h_\omega) \quad \forall \omega$. For a given choice of model for blur kernel, we estimate parameters θ based on \mathbf{Y}_ω 's using maximum likelihood principle. For ease of calculation we have used likelihood of $|\mathbf{Y}_\omega|^2$'s.

Result: If $Z \sim \mathcal{CN}(0, \sigma^2)$. We know that $Re(Z)$ and $Im(Z)$ follows $\mathcal{N}(0, \frac{\sigma^2}{2})$ independently. Then, $|Z|^2 = Re^2(Z) + Im^2(Z) \sim \frac{\sigma^2}{2} \chi_2^2 \equiv Exp(\lambda = \frac{1}{\sigma^2})$.

Using last result, $|\mathbf{Y}_\omega|^2 \sim Exp(\lambda_\omega = \frac{1}{\sigma^2 |K_\omega|^2 g_\omega + \eta^2 h_\omega}) \quad \forall \omega$ independently. If $f_{\theta, \omega}(\cdot)$ denotes the pdf of $Exp(\lambda_\omega = \frac{1}{\sigma^2 |K_\omega|^2 g_\omega + \eta^2 h_\omega})$ for given parameters θ (say,) of blur kernel model. Then, likelihood of $|\mathbf{Y}_\omega|^2$ is given by -

$$f_\theta(|Y_\omega|^2, \forall \omega) = \prod_{\omega} f_{\theta, \omega}(|Y_\omega|^2) \quad (11)$$

The above considerations are only for gradients in a particular direction, i.e., *horizontal* or *vertical*. To incorporate both directions in the estimation procedure, we assume for simplicity that they are independent. The joint likelihood is then given by

$$L(\boldsymbol{\theta}) = L_h(\boldsymbol{\theta}) \times L_v(\boldsymbol{\theta}) = f_{\theta}(|Y_{h,\omega}|^2, \forall \omega) \times f_{\theta}(|Y_{v,\omega'}|^2, \forall \omega') \quad (12)$$

Our goal is to find the $\boldsymbol{\theta}$ maximizing $L(\boldsymbol{\theta})$, or equivalently $\log L(\boldsymbol{\theta}) = \log L_h(\boldsymbol{\theta}) + \log L_v(\boldsymbol{\theta})$. We do this by numerically optimizing the objective (log-likelihood) function. For the spatially varying case, we can apply this procedure to local patches η_t for all pixel locations t in the image domain.

6 Choice of Tuning Parameters in ML Estimation

The log-likelihood, defined in (12), is a complicated function of the blur kernel parameter $\boldsymbol{\theta}$. This parameter is involved in the expression for λ_{ω} through $|K_{\omega}|^2$, which itself is a complicated function of $\boldsymbol{\theta}$. Therefore, before we start using any optimization method to find the maximizer of the log-likelihood, we should investigate the behavior of $L(\boldsymbol{\theta})$ as a function of $\boldsymbol{\theta}$. For this, we conducted simulated experiments as shown in Figure 5. For now, we will focus on the disc kernel. We considered a sequence of values for r ranging from 1 to 4 with a step size of $\Delta r = 0.05$, and for σ ranging from 0.01 to 0.20 with a step size of $\Delta \sigma = 0.01$, while keeping $\eta = 0.005$ constant.

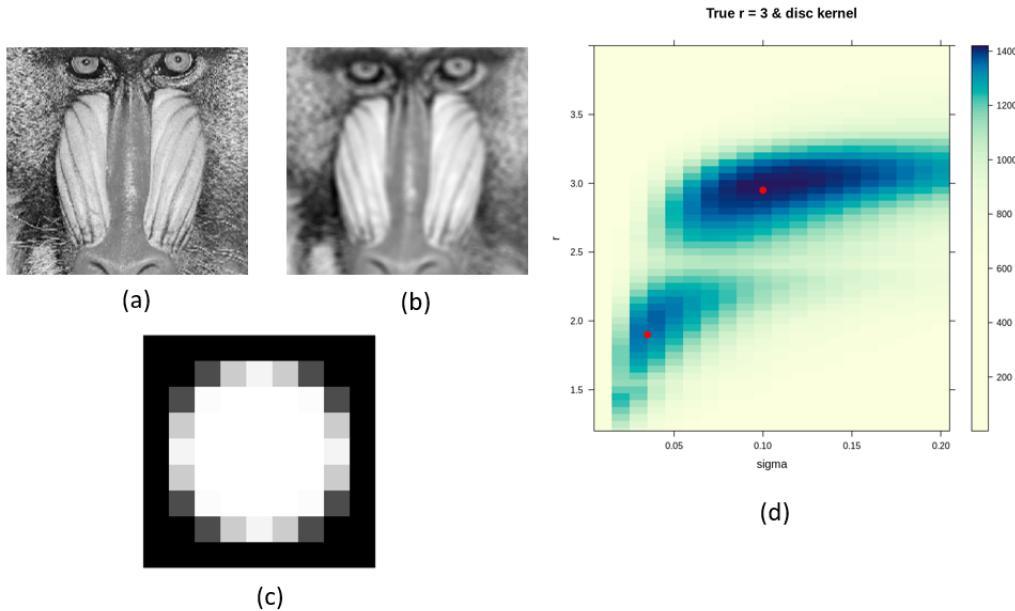


Figure 5: (a) 101×101 Sharp Image, (b) Blurred Image Using disc kernel with $r = 3$, (c) Disc Kernel with $r = 3$, (d) Levelplot of log likelihood as a function of σ and r

In Figure 5, (a) shows the original sharp image of size 101×101 . We use a disc kernel with $r_{\text{true}} = 3$ to simulate defocus blur as shown in (b). The log likelihood is plotted as a function of r

and σ in (d). Two local maxima points are visible in the plot, indicated by red dots. The global maximum is located near $r = 3$, with the corresponding value of σ around 0.10. Maximizing the log likelihood over a grid of values of (r, σ) gives a correct estimate of r in this case. However, it is not guaranteed that the global maximum always indicates the true parameter r . Figure 6 shows another example, where we implement the same simulation with $r_{true} = 2$. But this time the global maximum is attained at $r = 1.48$ with corresponding $\sigma = 0.03$, which is different from the true parameter value. In this case, we have three local maxima, with one near the true parameter value and corresponding σ near 0.10.

Simply looking for global maxima is not the solution. Choice of the prior parameter σ significantly affects the performance of ML Estimation. In the Bayesian Paradigm, this is quite common. If we can choose σ such that the maximizer of the log-likelihood for that given σ closely matches the true parameter, then we are done. To find a reasonable value of σ we use simulation. We consider six sharp images of size 255×255 and true parameter values $r_{true} = 1, 3, 5$ to simulate defocus blur. For each r_{true} value and fixed patch size, we randomly select five patches from each image and plot the estimated r as a function of σ for each random patch. We have considered patch sizes 51×51 , 101×101 , and 201×201 .

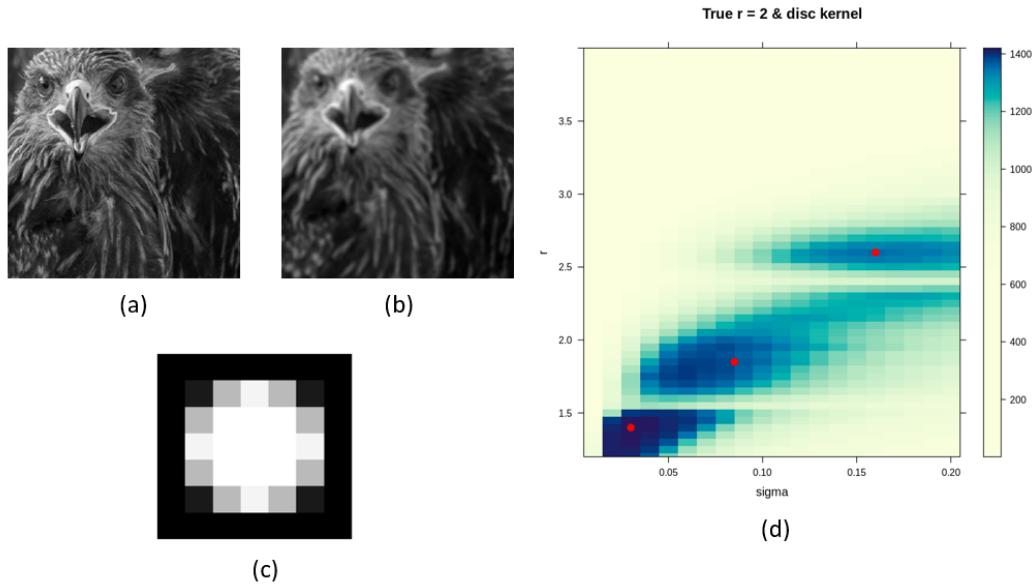


Figure 6: (a) 101×101 Sharp Image, (b) Blurred Image Using disc kernel with $r = 2$, (c) Disc Kernel with $r = 2$, (d) Levelplot of log likelihood as a function of σ and r

From Figure 7, 8 and 9, we observe that as patch size increases, the estimated r becomes more stable as a function of σ . Most importantly for $\sigma = 0.10$, the estimated r closely matches with the true value r_{true} across all cases, indicating that $\sigma = 0.10$ is a suitable choice for the prior parameter. This observation is consistent with Figures 5 and 6. It is also clear that the maximum likelihood estimation does not perform well for small patch sizes as expected. Because we have less number of pixels for small patch sizes.

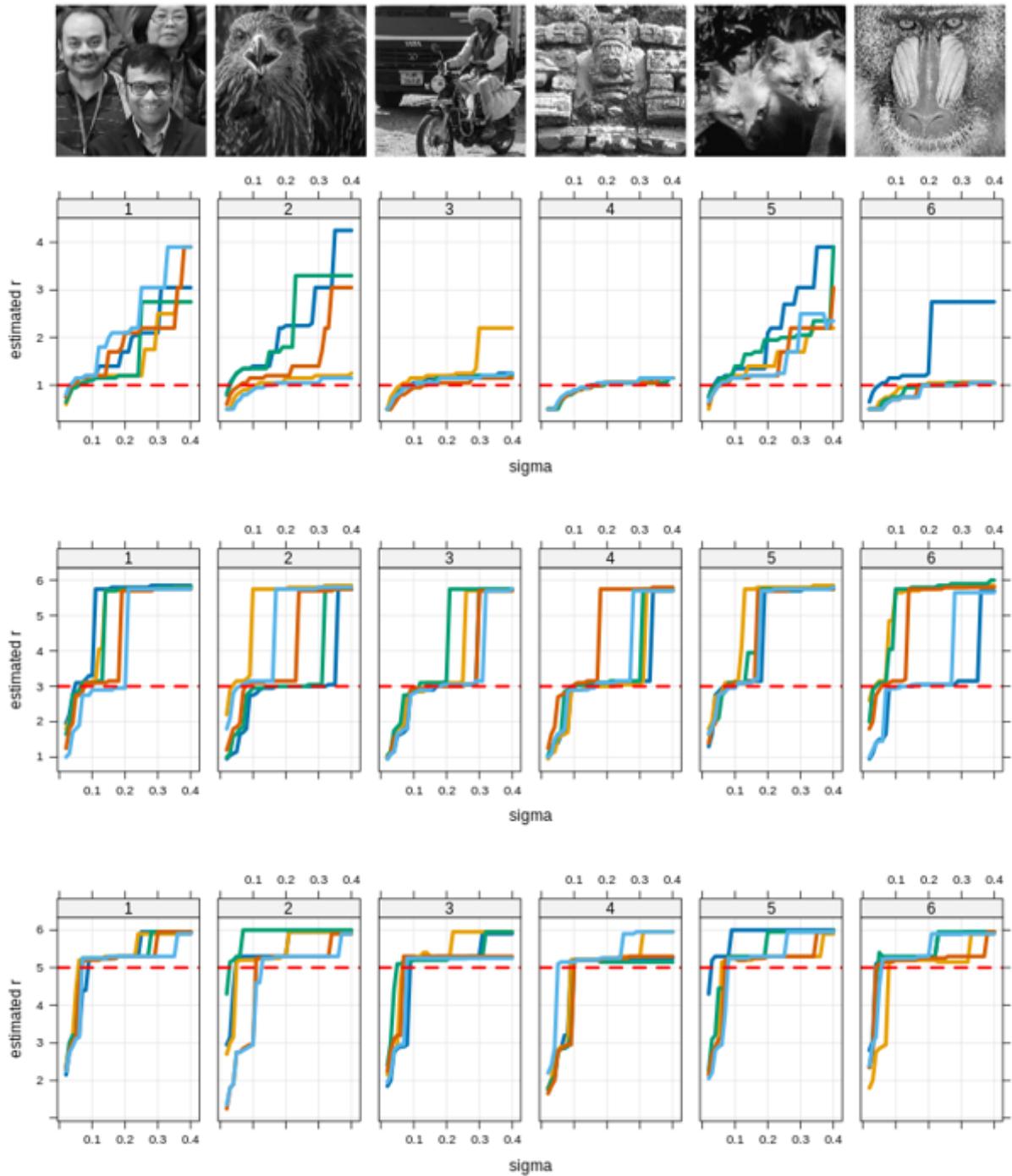


Figure 7: 51×51 Patch and Disc Kernel: Top row indicates the sharp images used for simulation and second, third and bottom row corresponds to $r_{true} = 1, 3, 5$ respectively.

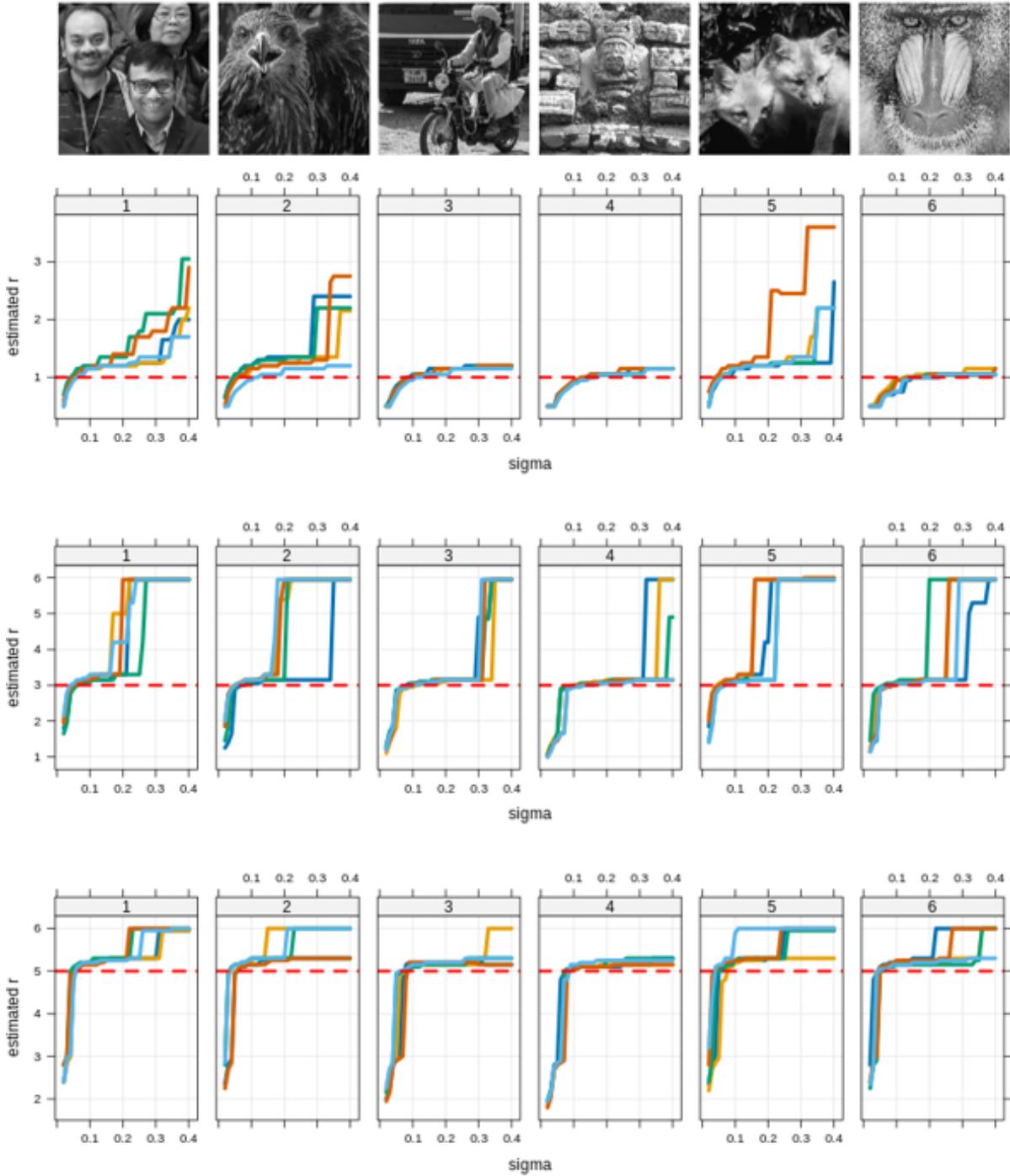


Figure 8: 101×101 Patch and Disc Kernel: Top row indicates the sharp images used for simulation and second, third and bottom row corresponds to $r_{true} = 1, 3, 5$ respectively.

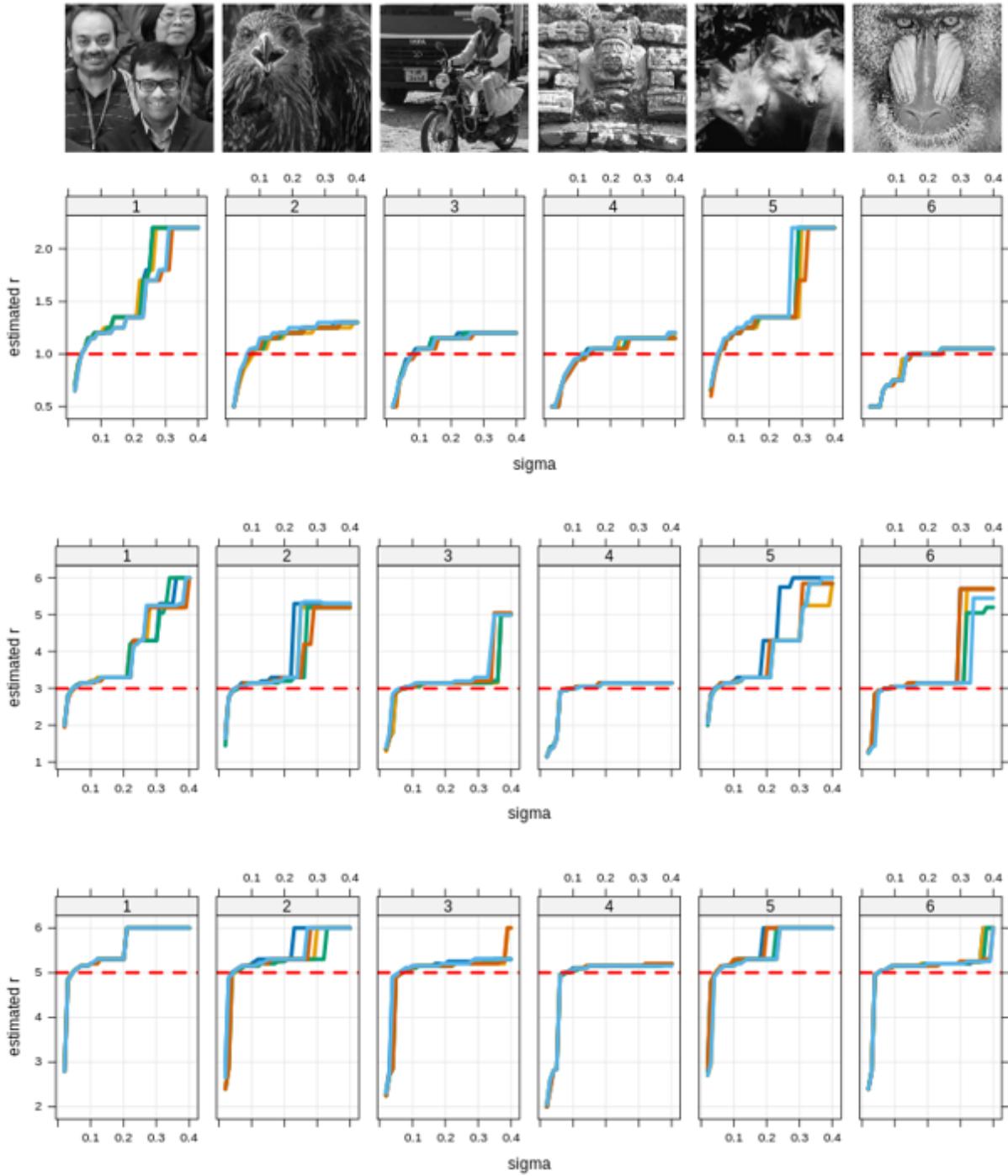


Figure 9: 201×201 Patch and Disc Kernel: Top row indicates the sharp images used for simulation and second, third and bottom row corresponds to $r_{true} = 1, 3, 5$ respectively.

7 Considerations in Spatially Varying Blur Estimation

We have developed a procedure for estimating blur kernel parameters for images with uniform blur. However, real-world images often exhibit spatially varying blur, where the level of blur varies from pixel to pixel. To address this, one reasonable assumption is that the level of blur remains locally constant within a small patch around each pixel. By applying our estimation procedure to each local patch, we can obtain the blur level for each pixel, as described in Equation (9). This results in an estimated blur map for the image.

In Figure 11, we have tried to estimate spatially varying blur for image of moon. But, instead of estimating the blur level for each pixel, we divided the image into 56 equal parts and for each part, we estimated the blur kernel using the maximum likelihood procedure assuming circular Cauchy kernel with $\kappa = 0.5$ to estimate the blur radius. The estimated blur kernels are overlaid on the image, part by part, for visualization. As expected, the blur radius map is spherically symmetric with minimum value at the center and it increases as we move away from center.

In Figure 10, we have conducted a simple experiment using spatially varying blur with a disc kernel on a 255×255 image. We have used local patches of size 51×51 for estimation of blur radius for each pixel. Both the original and estimated blur maps are shown. It is evident that the estimated blur map closely resembles the original blur map except for the transition regions.

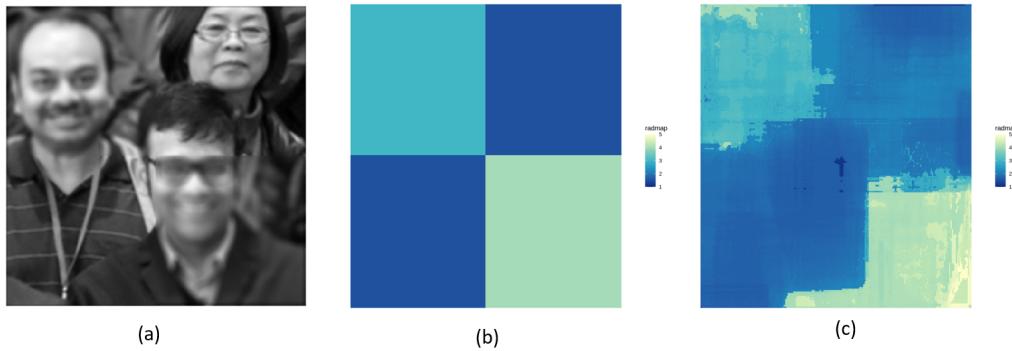


Figure 10: Spatially Varying Blur Estimation Under Locally Constant Assumption: (a) Spatially Varying Blurred Image, (b) Latent Blurmap, (c) Estimated Blurmap

The estimated blur map is clearly not smooth, particularly for transition regions between two blur levels. To obtain a smoother blur map, we need to employ *Markov Random Field Smoothing*. The idea is to solve a penalized problem, penalizing the difference between blur levels on neighboring pixels. An adaptive weighting, primarily based on RGB distance between pixels, is used in literature to obtain a smoothed blur map. But, this entire procedure is time-consuming and fails in situations where a local patch contains pixels with very different depth values.

The assumption of locally constant level of blur is not always valid. In real images, level of blur tends to remain constant for different objects within the image, as it is the objects themselves that vary in distance from the camera, not the individual pixels. Figure 2 provides an example illustrating this phenomenon, where different birds in the image exhibit varying levels of blur.

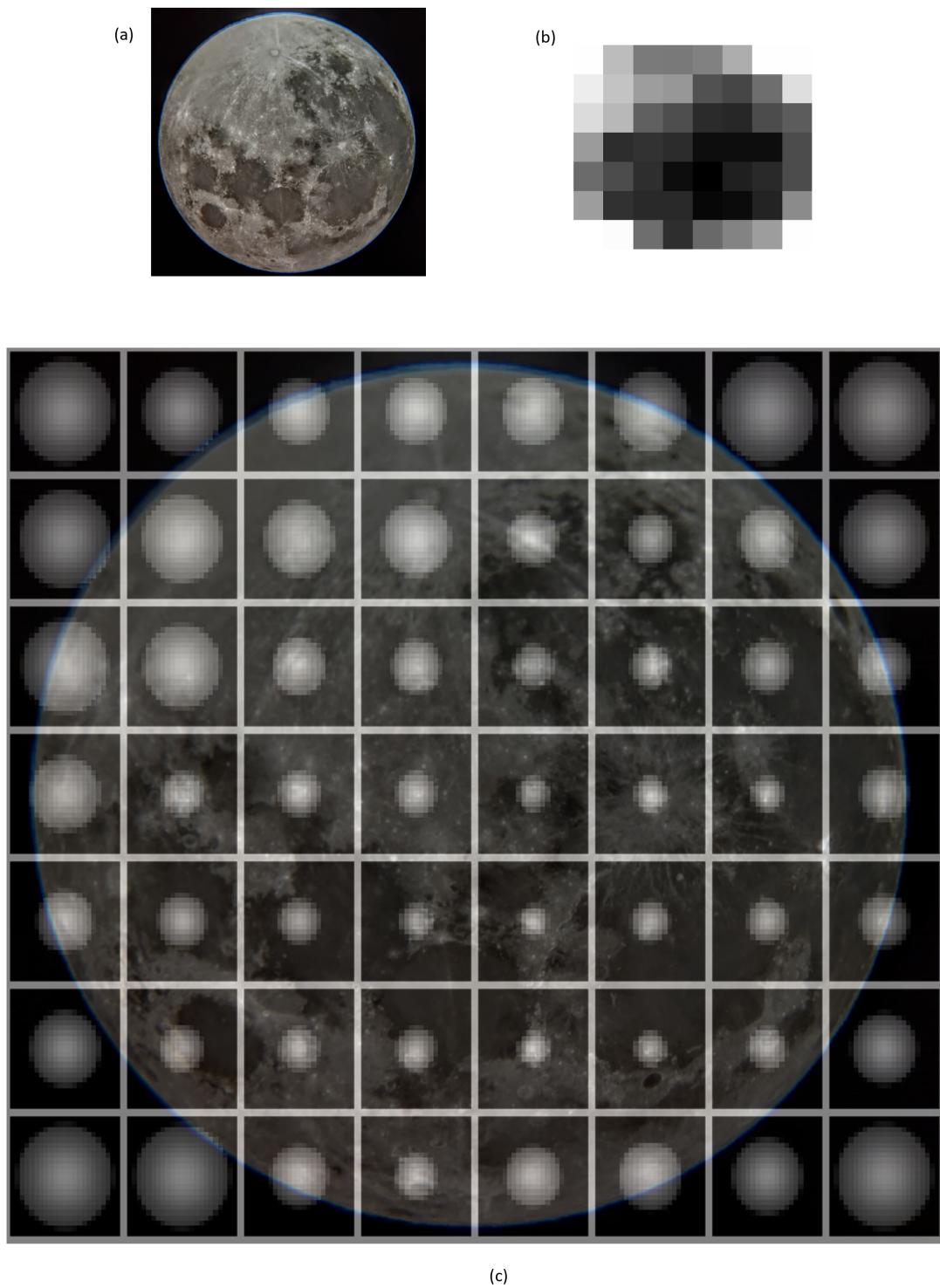


Figure 11: Spatially Varying Blur for Moon: (a) Image of Moon, (b) Blur radius map using circular cauchy kernel with $\kappa = 0.5$ (black for small radius and white for large), (c) Estimated blur kernels are overlaid on the image part by part

Therefore, instead of attempting to estimate the blur level for each pixel, a more effective approach would be to estimate the blur level for each distinct object in the image. By segmenting the image into meaningful objects (such as birds in Figure 2), and estimating the blur level for each segment, we can generate a more accurate blur map easily.

8 Segment Anything

We used Segment Anything (SAM), developed by Meta AI in 2023, for image segmentation. SAM offers versatile modes of use, including automatic and guided segmentation (Figure 12). It employs a unique network architecture akin to language models like *ChatGPT* and *Google Bard*, utilizing an image encoder for embedding computation, a prompt encoder for prompt embedding, and a lightweight decoder for mask prediction. SAM’s data engine collected over 1.1 billion high-quality masks resulting in the *SA-1B dataset*, a significant contribution to image processing field. SAM’s promptable architecture enables zero-shot learning and effectively handles various downstream tasks. Several domain specific modifications of SAM are available such as Ma et al. [5] and Osco et al. [7].



Figure 12: Automatic and Guided Segmentation Using Segment Anything Model

We used SAM for automatic segmentation. In this mode, SAM considers bounding boxes aimed at identifying and segmenting objects within an image efficiently. After identifying the bounding boxes, SAM produces three possible masks for each bounding box along with a confidence score. We require both the bounding boxes and masks for our purpose. However, in the presence of a small number of observations (i.e., pixels), our ML estimation procedure may produce poor results. Therefore, we filtered out the small segments based on the size of the bounding box and the confidence score.

9 Challenges in Post-Segmentation ML Estimation

For post-segmentation estimation of the blur map, we need to apply the maximum likelihood procedure to each image segment identified by SAM. Unfortunately, this is not straightforward, because the maximum likelihood procedure as outlined above requires rectangular images, whereas the image segments obtained are of irregular shape. The obvious solution is to fill / pad

the segments with zeros to make it a rectangular array; however, this approach typically leads to bias towards selecting smaller values of the radius r . The issue arises because the discrete Fourier transform (DFT) combines frequency content from across all spatial locations, implicitly assuming stationarity of the gradients, which no longer holds.

Some approaches we tried to address this issue are the following.

- Instead of utilizing the irregular segments identified by SAM, we tried using the corresponding bounding boxes as inputs. However, bounding boxes may encompass objects with vastly different depths, leading to significantly varied blur levels within the same box. In such scenarios, we've observed that the sub-area of the box with the least level of blur tends to dominate the Fourier spectrum.
- Another strategy is to decompose irregularly shaped segments into smaller, rectangular regions and apply the estimation procedure to these sub regions. However, this approach possibly discards a large proportion of the available data depending on how irregular the segment is.
- Instead of working in the frequency domain, we can work in the spatial domain. As convolution is a linear operator, we can express (6) as $\text{vec}(\mathbf{y}) = \mathbf{A}_k \text{vec}(\mathbf{x}) + \text{vec}(\mathbf{n})$, where $\text{vec}(\cdot)$ represents the corresponding vectorized versions of image segments, and \mathbf{A}_k is a matrix determined by the blur kernel \mathbf{k} . We can then formulate the maximum likelihood procedure as previously done in the frequency domain. However, this approach is computationally infeasible. To understand why, we first consider the simpler IID prior on image gradients \mathbf{x} , assuming further that elements of \mathbf{n} are also independent. Then, $\text{vec}(\mathbf{y})$ follows a multivariate normal distribution $\text{MVN}(\mathbf{0}, \sigma^2 \mathbf{A}_k \mathbf{A}_k^T + \eta^2 \mathbf{I})$. To form a maximum likelihood estimation procedure, we need to calculate the likelihood (given \mathbf{y}) as a function of $\boldsymbol{\theta}$ (which characterizes the blur kernel \mathbf{k}). For observed image gradient \mathbf{y} the likelihood is given by

$$L(\boldsymbol{\theta}) \propto \frac{1}{\sqrt{\det(\sigma^2 \mathbf{A}_k \mathbf{A}_k^T + \eta^2 \mathbf{I})}} \exp\left(-\frac{1}{2} \text{Vec}(\mathbf{y})^T (\sigma^2 \mathbf{A}_k \mathbf{A}_k^T + \eta^2 \mathbf{I})^{-1} \text{Vec}(\mathbf{y})\right) \quad (13)$$

The parameter $\boldsymbol{\theta}$ determines \mathbf{A}_k . To evaluate the likelihood function using (13), we need to calculate the inverse and determinant of the matrix $\sigma^2 \mathbf{A}_k \mathbf{A}_k^T + \eta^2 \mathbf{I}$. Although this matrix is sparse, it is still potentially large, and evaluating the likelihood for multiple values of θ for each segment is impractical.

In the next section, we describe an ad hoc alternative loss function that is simpler to evaluate.

10 Decorrelation Loss Function

For an observed blurred image \mathbf{y} , if the true blur kernel is \mathbf{k}_{r_0} , it is evident from Equation (6) that deconvolving \mathbf{y} using \mathbf{k}_{r_0} would yield a quantity \mathbf{x}_{r_0} closely resembling the latent sharp image gradient \mathbf{x} . Consequently, the autocorrelation function of the horizontal and vertical gradients of \mathbf{x}_{r_0} would exhibit smaller values compared to the blurred image gradient. To empirically validate this, we simulated defocus blur on a 255×255 image using a disc kernel of radius $r_{\text{true}} = 3$. We have used a deconvolution procedure similar to Wiener filtering to obtain \mathbf{x}_r . For comparison, we have plotted ACF for blurred image gradients, deconvoluted gradients using $r = 2, 3, 4$ and also of sharp images in Figure 13.

For $r = 2$, we can see that ACF values are much smaller compared to the blurred image, and for $r = 4$, at some lags, the ACF takes on large values. But for the true value of r , we observe a substantial decrease in ACF values in all directions as expected. It is to be noted that for small values of radius, we have a similar type of decrease in along direction ACF values.

Based on these observations, we consider sum of squared ACFs in all directions as loss function. It captures the amount of correlation present in image gradients after deconvoluting using \mathbf{k}_r . We call it *Decorrelation Loss*. For true value of r , we expect this to be small. We have used the following theorem for deconvolution step.

Theorem: *If $\mathbf{Y}_\omega = \mathbf{K}_\omega \mathbf{X}_\omega + \mathbf{N}_\omega$ with $\mathbf{X}_\omega \sim \mathcal{CN}(0, \sigma^2 g_\omega)$ and $\mathbf{N}_\omega \sim \mathcal{CN}(0, \eta^2 h_\omega)$ independently. Then, the conditional expectation of \mathbf{X}_ω given $\mathbf{Y}_\omega, \mathbf{K}_\omega$ is given by*

$$\mathbb{E}[\mathbf{X}_\omega | \mathbf{Y}_\omega, \mathbf{K}_\omega] = \frac{\sigma^2 g_\omega \mathbf{K}_\omega \mathbf{Y}_\omega}{\eta^2 h_\omega + \sigma^2 g_\omega |\mathbf{K}_\omega|^2} \quad (14)$$

Using Equation (14) we have obtained the deconvolved versions of both the horizontal and vertical gradients in the frequency domain, and then performed an Inverse Discrete Fourier Transform (IDFT) to obtain their spatial domain counterparts, denoted by $\mathbf{x}_{h,r}$ and $\mathbf{x}_{v,r}$ respectively. Then, we calculate the autocorrelation function (ACF) along and across the gradient directions for both the horizontal and vertical deconvoluted gradients. Typically, considering only the ACFs along the direction is sufficient. However, we have observed that including both along and across direction ACFs results in more stable outcomes. We select the value of r that minimizes the decorrelation loss.

For image segments, we slightly modify the procedure. We perform deconvolution of bounding boxes corresponding to the image segments and instead of calculating the ACF for all pixels within that bounding box, we only consider the pixels that belong to the image segment corresponding to that bounding box.

The choice of the parameters σ and η are also important. We have conducted an experiment similar to maximum likelihood case to obtain optimal choice of parameters and the results are similar. Choice of ρ_1 and ρ_2 are also important. The default choices are $\rho_1 = 0.5$ and $\rho_2 = 0.5$.

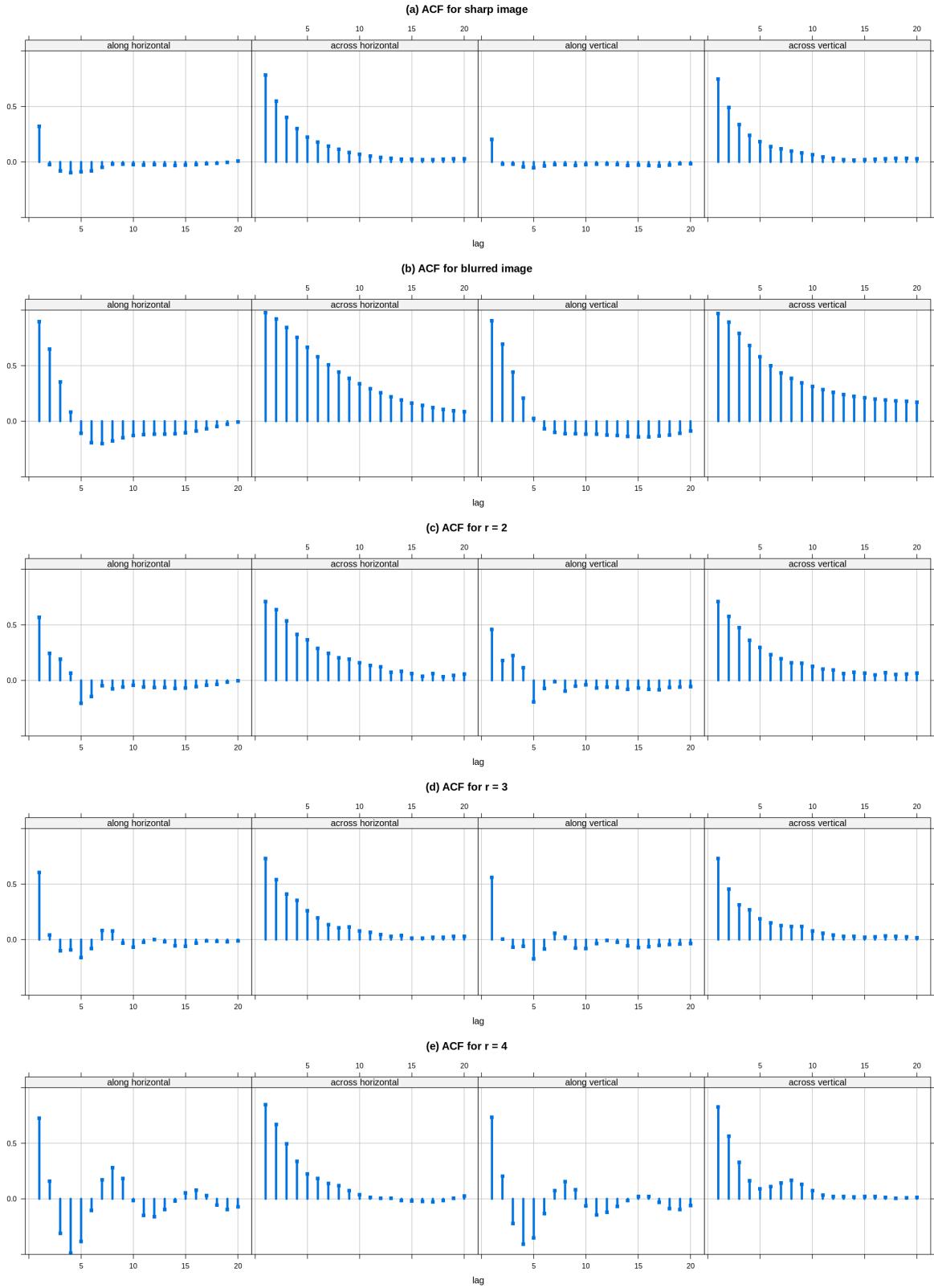


Figure 13: Motivation for Decorrelation loss: Top row indicates ACF for sharp image gradients, second row for blurred image gradients and last three rows for deconvoluted gradients. The correct value of the radius is 3.

11 Simulated Experiments

A simulated experiment is illustrated in Figure 14, which allows us to quantitatively test the performance of the proposed method. We start with a sharp original RGB image and apply SAM to identify the image segments. SAM considers different bounding boxes and produces three possible masks for each such bounding box along with their confidence scores. We discard small segments based on the size of the bounding box and the confidence score of the corresponding mask. Afterward, we simulate defocus blur based on these masks using a disc kernel, employing the model in (9) and considering the segments as neighbors of the pixel they belong to. The latent blur map and generated blurred image are shown in Figure 14 (a) and (c). We apply the above-discussed method to the gray scale version of the blurred image with $\sigma = 0.10$, $\eta = 0.005$, and $\rho = (0.5, 0.5)$ as the choice of tuning parameters. The estimated blur map is shown in Figure 14 (d). We have used a grid-search procedure with step size $\Delta r = 0.1$ to search for optimal r .

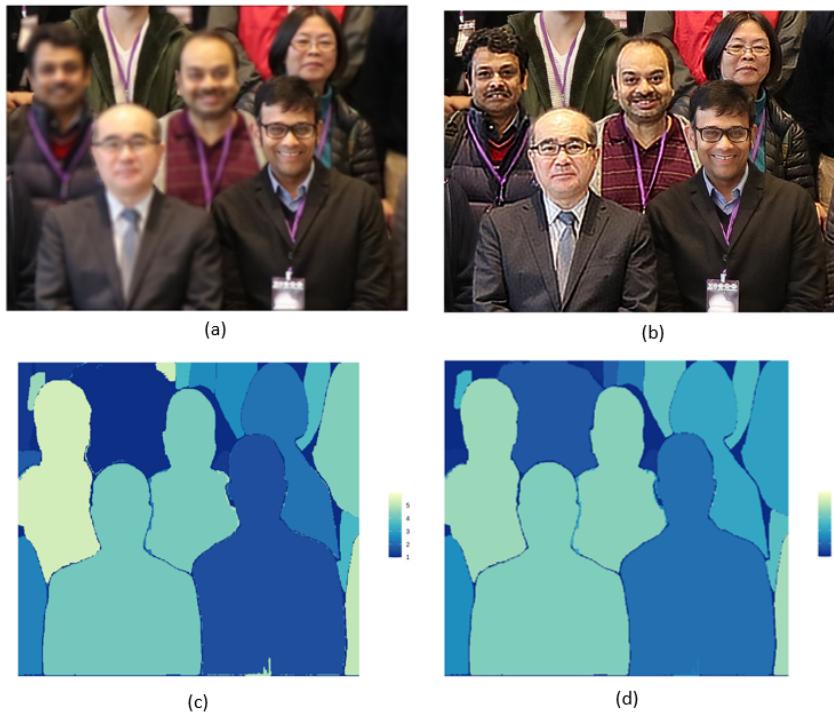


Figure 14: Simulated Experiment: (a) Simulated input image with spatially varying blur, (b) Deblurred image based on estimate in (d), (c) Latent blur map, (d) Estimated blur map

Notice that except for some regions, the estimated blur map closely resembles the latent blur map. The discrepancy is mainly for small segments. Using the estimated blur map and blur kernels generated by the disc function, we performed spatially varying deconvolution using Richardson Lucy algorithm with number of iterations 30. The deblurred image is shown in Figure 14 (b). We can see that the spatially varying blur is successfully removed from the image. The PSNR of the estimate $\hat{\mathbf{x}}$ of the sharp image \mathbf{x} is 54.47 dB, which is quite high.

12 Application on Real Images

We applied the procedure described above to real images, and the results are shown in Figure 15,16 and 17. We have also included the deblurred images obtained using *Richardson-Lucy Algorithm* [9]. Since these images are sourced externally, calibrated blur kernels corresponding to them are unavailable. While blur from an ideal lens with a circular aperture can be modeled by a disc kernel in the absence of diffraction effects, real-world scenarios involve diffraction. In our experience, disc kernels can produce sufficiently accurate blur maps for real images, but the loss curves may not be stable. Alternatively, circular kernels perform well in producing stable blur maps suitable for deconvolution. For the results presented here, we used a circular Cauchy kernel with $\kappa = 0.5$ to estimate the radius map (i.e., r). We have used $\sigma = 0.10$, $\eta = 0.010$, and $\rho = (0.5, 0.5)$ for blur radius estimation in all cases, along with 30 iterations for deblurring using Richardson-Lucy algorithm.

Figure 15 (a) illustrates an example demonstrating varying levels of blur. The two persons appear blurred because they are closer to the camera. Moving away from the persons towards the merry-go-round, the level of blur decreases, reaching a minimum near the merry-go-round. Therefore, the plane of focus of the camera in this scene is the merry-go-round. Our method effectively captures the differences in blur levels. Although the ranking is not entirely accurate, it produces a sufficiently good blur map for deblurring the image. The deblurred image demonstrates successful removal of spatially varying blur. Figure 15 (b) illustrates another example with only two levels of blur. Our method captures the differences in blur levels. In the deblurred image, we can clearly see the patterns on the ground that were not clear in input image.

Figure 15 (c) presents an example with multiple blur levels. The woman closer to the camera appears more blurred, and the level of blur decreases as we move towards the back until the fourth person. After that, it increases again. The estimated blur map also reflects these variations. Although the wall in the background should be the most blurred, our blur map does not show this. However, the resulting deblurred image is sufficiently sharp. Figure 15 (d) presents another example of multiple blur levels, and our method is able to identify them nearly accurately.

The input images in Figure 16 are from Levin et al. [4]. In each case, the blur maps depict the nearly correct blur level estimation, and the deblurred images also show improvement. Figure 17 (a),(b) presents close-ups of faces from 16 (a),(c) respectively before and after deblurring, demonstrating the improvement achieved. The deblurred images are not smooth, but we can achieve smoothness using other deconvolution algorithms, For example putting L_p penalty on deblurred image gradients [6]. But those are much slower compared to Richardson-Lucy.

Finally, in Figure 17 we show some comparisons from examples given in Zhu et al. [11]. A notable improvement is the accurate identification of boundaries between objects in the blur maps, which is a consequence of using SAM to perform pre-segmentation. Our method clearly depicts the differences in blur levels, such as between the cattle in (e), the individuals and buildings in (d), whereas these details are absent in [11]. The ranking is not entirely correct, but it provides a better representation of 3D geometric information compared to [11].



Figure 15: Spatially Varying Blur Estimation: First Column for input blurry image, second column for estimated blur radius map and third column for deblurred image using estimated blur radius map in second column

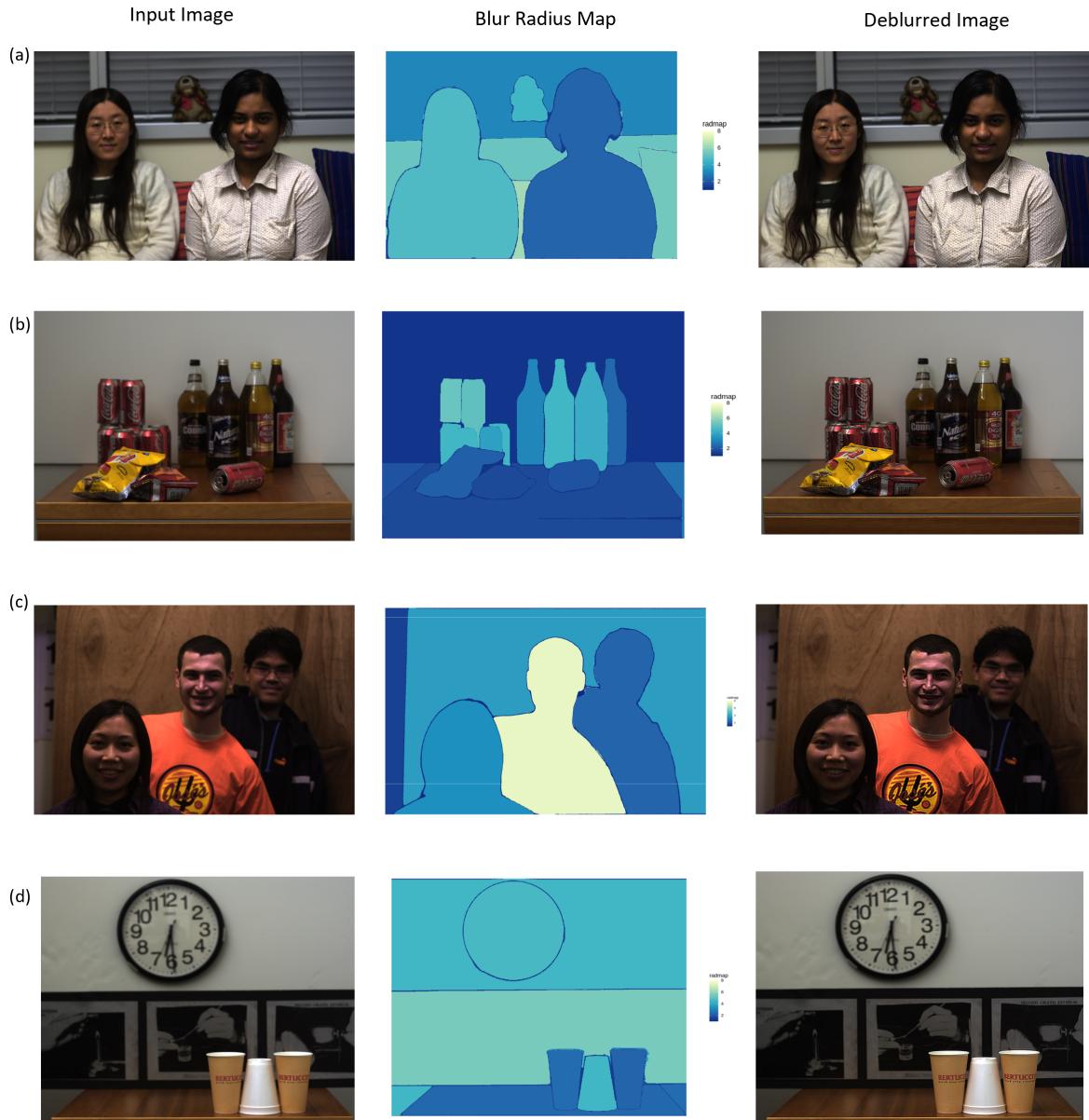


Figure 16: Spatially Varying Blur Estimation: First Column for input blurry image, second column for estimated blur radius map and third column for deblurred image using estimated blur radius map in second column

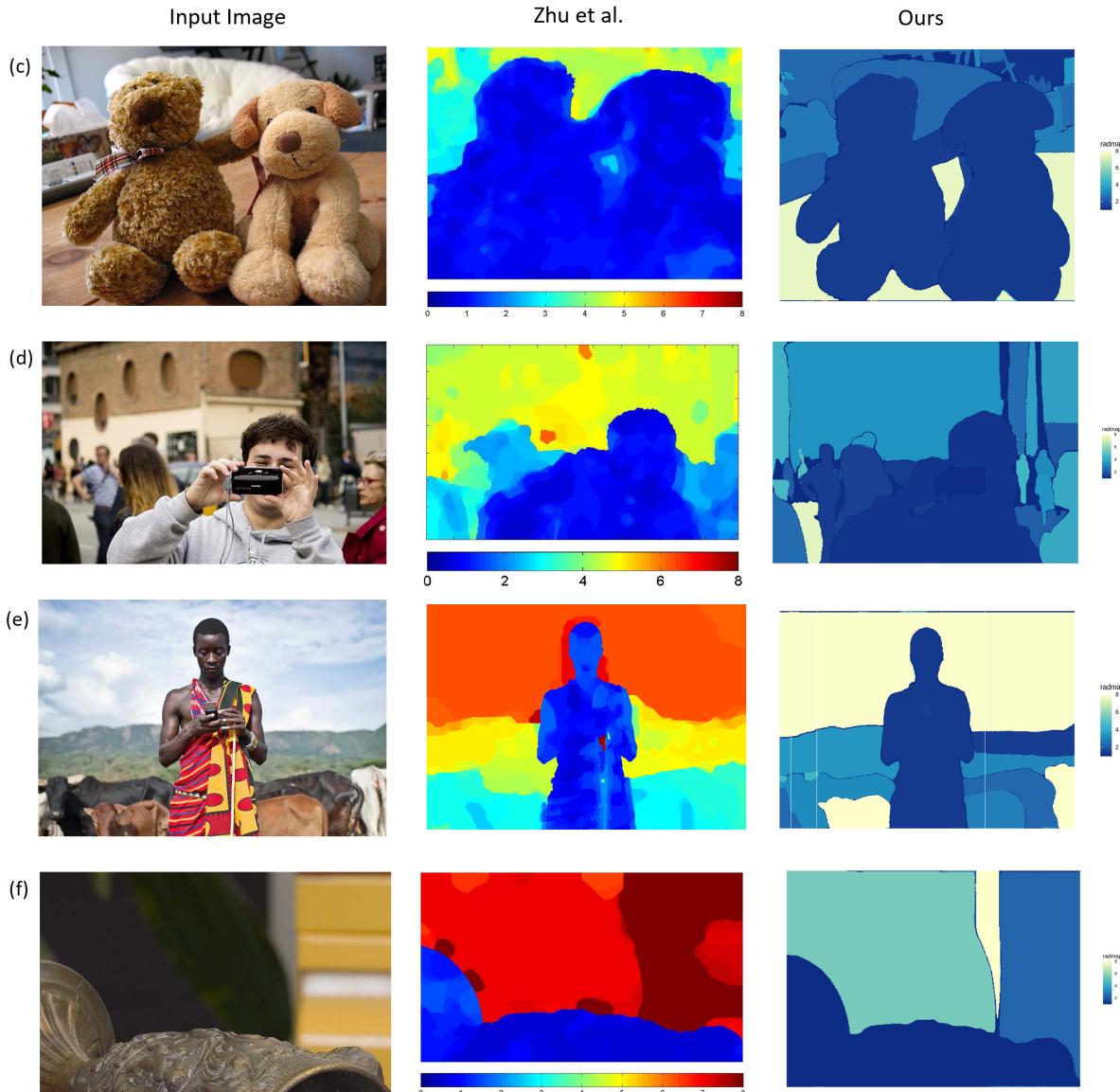
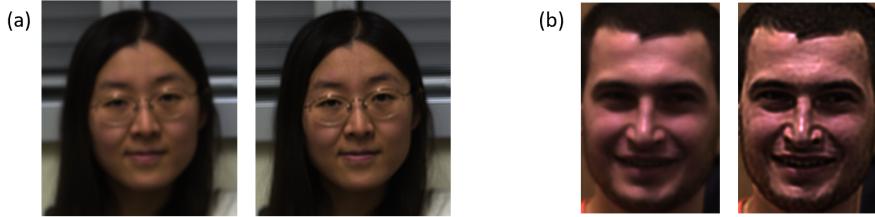


Figure 17: Closeups and Comparison with our method with [11]: (a), (b) shows improvement after deblurring and (c), (d), (e), (f) are comparison of blurmaps

13 Discussion

In this project, we proposed a new method to estimate spatially varying blur from a single image. We initially aimed to estimate the blur level for segments identified by SAM using the maximum likelihood procedure. However, due to the issue of zero padding with ML estimation, we developed an ad hoc procedure based on the idea of *Decorrelation Loss*.

Our initial goal was to estimate depth using the blur level as a surrogate. However, this is an ill-posed problem because, for a given value of r (i.e., $c_{diam}/2$), there can be two different values of d , as seen from Equation 10. However, the blur maps can be considered as depth maps if the object closest to the camera in the image is the sharpest in the scene. The images in Figure 16 are examples of this, where the corresponding blur map can be viewed as depth map.

The ranking in blur maps is not entirely correct. One common issue is identifying the blur level of background objects. Additionally, segments with very little texture often result in underestimation of the blur radius. This issue is common to most passive algorithms but can be mitigated using a post-segmentation approach with neighboring pixel information. In [11], the initial blur map estimate also had ambiguities, but after MRF smoothing, they have obtained a reasonable blur map. We need to think whether a post-estimation modification can remove the ambiguities in our blur maps.

In this project, we only considered the disc kernel for simulation and the circular cauchy kernel with $\kappa = 0.5$ for real image applications. However, we can generalize it for any spherically symmetric kernel. The choice of κ is based on empirical observations, but we can also treat κ as a parameter and incorporate into estimation procedure. For a single image, the choice of κ should be the same for all segments (pixels), and one needs to consider this while estimating κ .

The choice of tuning parameters σ , η , and ρ is crucial. We based our selection of these parameters on a simulation study. However, when dealing with real images, we may need to examine them more closely. It's true that selecting the right tuning parameters can improve our blur map estimation process. But it is hard to isolate the effects of each tuning parameter individually because of their confounding effect.

The blur maps estimated by our method clearly depict the differences in blur levels and boundaries between objects in image, unlike those in [11, 4]. This improvement is mainly due to segmenting the image into meaningful objects and using that for estimation. It reduces computation time significantly. Because, instead of estimating for each pixel separately, we estimate only for segments, which is significantly fewer than the number of pixels. Currently our R implementation takes around 7 minutes to perform the estimation step for a 1000×1000 image on CPU run time and additional 3 minutes for segmentation using SAM on GPU run time in Google Colab Platform. In contrast, methods based on estimating the blur level for each pixel separately take around 1 hour and 30 minutes only for the initial blur map estimation on the same platform and run time. This demonstrates a significant improvement in computation time.

14 Supplementary Materials: DepthR

A R package implementing our estimation procedure is available on Github named `DepthR` (<https://github.com/ShrayanRoy/DepthR>). It has implementation of both maximum likelihood procedure and decorrelation loss. The code for segmentation using SAM is also available.

15 Acknowledgement

First and foremost, I would like to express my heartfelt gratitude to my supervisor Dr. Deepayan Sarkar, for his unwavering support and deep involvement throughout my project. His exemplary mentorship and expertise were instrumental in the successful completion of my work.

I am indebted to the faculty members at ISI Delhi and Kolkata for their continuous support throughout my M.Stat Programme, nurturing my growth at every step. Their guidance, both in and out of the classroom, provided the foundation upon which this thesis stands.

Finally, I express my gratitude to my family and friends for their unwavering support and encouragement throughout this endeavor. Their belief in me fueled my determination to persevere.

References

- [1] Brian A. Barsky, Daniel R. Horn, and Klein. “Camera Models and Optical Systems Used in Computer Graphics: Part I, Object-Based Techniques”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2003. URL: http://dx.doi.org/10.1007/3-540-44842-X_26.
- [2] P. Grossmann. “Depth from focus”. In: *Pattern Recognition Letters* (1987). ISSN: 0167-8655. URL: [http://dx.doi.org/10.1016/0167-8655\(87\)90026-2](http://dx.doi.org/10.1016/0167-8655(87)90026-2).
- [3] Alexander Kirillov et al. *Segment Anything*. 2023. URL: <https://arxiv.org/abs/2304.02643>.
- [4] Anat Levin et al. “Image and depth from a conventional camera with a coded aperture”. In: *ACM transactions on graphics (TOG)* 26.3 (2007), 70–es. URL: <http://dx.doi.org/10.1145/1276377.1276464>.
- [5] Jun Ma et al. “Segment anything in medical images”. In: *Nature Communications* (2024). DOI: <10.1038/s41467-024-44824-z>. URL: <http://dx.doi.org/10.1038/s41467-024-44824-z>.
- [6] Kaustav Nandy. “Locally Dependent Natural Image Priors for Non-blind and Blind Image Deconvolution”. PhD thesis. Indian Statistical Institute, 2021. URL: <https://digitalcommons.isical.ac.in/doctoral-theses/7/>.
- [7] Lucas Prado Osco et al. *The Segment Anything Model (SAM) for Remote Sensing Applications: From Zero to One Shot*. 2023. URL: <https://arxiv.org/abs/2306.16623>.
- [8] Magda Peligrad and Wei Biao Wu. “Central Limit Theorem for Fourier Transforms of Stationary Processes”. In: *The Annals of Probability* (2010). ISSN: 0091-1798. URL: <http://dx.doi.org/10.1214/10-AOP530>.
- [9] William Hadley Richardson. “Bayesian-Based Iterative Method of Image Restoration*”. In: *J. Opt. Soc. Am.* (1972). DOI: <10.1364/JOSA.62.000055>. URL: <https://opg.optica.org/abstract.cfm?URI=josa-62-1-55>.
- [10] Deepayan Sarkar and Kaustav Nandy. *rip: Image Processing in R*. New Delhi, India, 2021. URL: <https://github.com/deepayan/rip>.
- [11] Xiang Zhu et al. “Estimating Spatially Varying Defocus Blur From A Single Image”. In: (2013). ISSN: 1941-0042. URL: <http://dx.doi.org/10.1109/TIP.2013.2279316>.