Richardson-Lucy Deblurring for Scenes under Projective Motion Path

Yu-Wing Tai Ping Tan Long Gao Michael S. Brown

***Abstract*— This paper addresses the problem of modeling and correcting image blur caused by camera motion that follows a projective motion path. We introduce a new *Projective Motion Blur Model* that treats the blurred image as an integration of a clear scene under a sequence of projective transformations that describe the camera's path. The benefits of this motion blur model is that it compactly represents spatially varying motion blur without the need for explicit blurs kernels or having to segment the image into local regions with the same spatially invariant blur. We show how to modify the Richardson-Lucy (RL) algorithm to incorporate our Projective Motion Blur Model to estimate the original clear image. In addition, we will show that our *Projective Motion RL* algorithm can incorporate state-of-the-art regularization priors to improve the deblurred results. Our Projective Motion Blur Model along with the Projective Motion RL is detailed together with statistical analysis on the algorithm's convergence properties, robustness to noise, and experimental results demonstrating its overall effectiveness for deblurring images.**

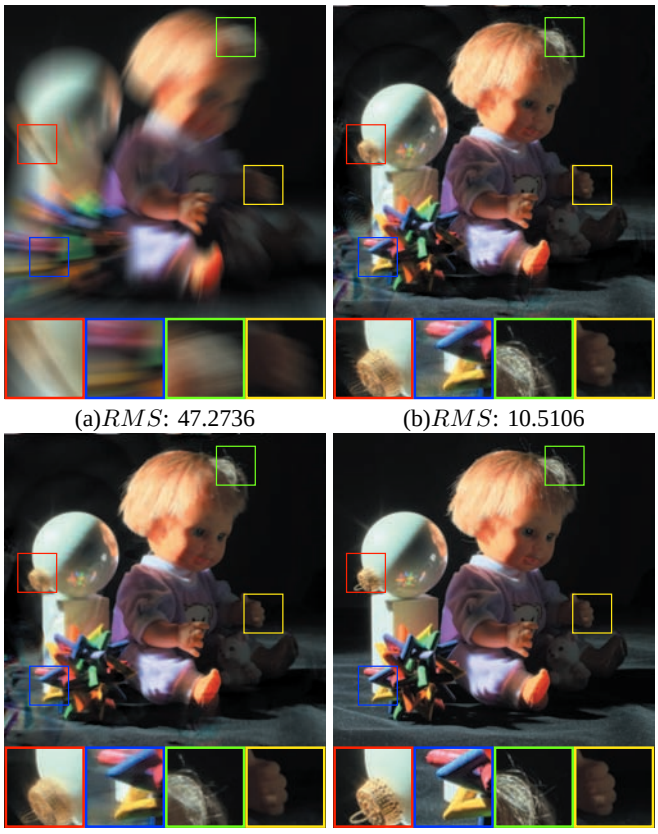
I. Introduction

Motion blur is an artifact in photography caused by the relative motion between the camera and an imaged scene during exposure. Ignoring the effects of defocusing and lens abbreviation, each point in the blurred image can be considered the result of convolution by a point spread function (PSF), i.e. a motion blur kernel, that describes the relative motion trajectory at that pixel position. The aim of image deblurring is to reverse this convolution process in order to recover a clear image of the scene from the captured blurry image.

A common assumption in motion deblurring is that the motion PSF describing the motion blur is spatially invariant. This implies that the motion blur effect is caused by camera ego motion that can be described by a global PSF. Artifacts arising from moving objects and depth variations in the scene are ignored. As recently discussed by Levin *et al.* [18], this global PSF assumption for blurred images caused by camera motion is invalid for practical purposes. In their experiments, images taken with cameras undergoing hand shake exhibited notable amounts of camera rotation that causes spatially varying motion blur within the image. As a result, Levin *et al.* [18] advocated the need for better motion blur models as well as image priors to impose on the deblurred results. In this paper, we address the former issue by introducing a new and compact motion blur model that is able to describe spatially varying motion blur caused by a camera undergoing a projective motion path. An example of the type of blur caused by a projective motion path and that of results obtained from our algorithm are shown in Figure 1.

We refer to our model as the *Projective Motion Blur Model*, as it represents the degraded image as an integration of the clear scene under a sequence of projective motions (Figure 2). The benefits of our model is that we do not explicitly require a PSF to describe the blur at any point in the image, thus our approach can be considered "kernel free". Likewise, we do not need to segment the image into regions of locally similar spatially invariant blur, as done in several previous approaches [15], [2], [7]. This makes our our approach compact in terms of its ability to describe the underlying motion blur without the need for local blur kernels. However, because our approach is not based on convolution with an explicit PSF, it has no apparent frequency domain equivalent. Therefore, one of the key contribution of this paper is to show how our blur model can be used to extend the conventional pixel-domain *Richardson-Lucy* (RL) deblurring algorithm to correct spatially varying motion blur.

We refer to this modified RL algorithm as the *Projective Motion Richardson-Lucy* algorithm. Similar to the conventional RL algorithm, regularization derived from image priors can be incorporated into our algorithm. We note that in this paper,



(a)RMS: 47.2736 (b)RMS: 10.5106

(c)RMS: 10.3517 (d) Ground truth

Fig. 1. (a) An image degraded by spatially varying motion blur. (b) Our result from our basic algorithm. (c) Our result with added regularization. (d) Ground truth image. The RMS errors are also shown below each image.

Fig. 2. Our input image is considered the integration of an image scene under projective motion .

our work is focused on developing the projective motion blur model and the associated RL algorithm. We will assume in our examples that the motion path of the camera is known and that the camera's motion satisfies our Projective Motion Blur Model. Potential methods for accurately estimating the projective motion path are discussed briefly in Section VI. As with other deblurring approaches, we assume that the scene is distance and does not contain moving objects.

The remainder of this paper is organized as follows: Section II discusses related work; Section III details our motion blur model; Section IV derives the Projective Motion Richardson-Lucy deconvolution algorithm; Section V describes how to incorporate regularization into our Projective Motion RL deconvolution algorithm with additional implementation details; Section VI discusses methods to estimate projective motion; Section VII provides an analytical analysis of the converge properties of our algorithm and its sensitivity to noise. Results and comparisons with other approaches are also presented. A discussion and summary of this work is presented in Section VIII.

II. Related Work

Motion deblurring is a well studied problem that is known to be ill-posed given that multiple unblurred images can produce the same blurred image after convolution. Nonetheless, this problem has received intense study given its utility in photography.

The majority of previous work models the blur using a spatially invariant PSF, i.e. each pixel undergoes the same blur. Traditional non-blind deblurring algorithms include the well-known Richardson-Lucy algorithm [24], [20] and Wiener filter [30]. While these algorithms can produce unblurred results, they often suffer from artifacts such as ringing due to unrecoverable frequency loss or due to poor PSF estimation. Recent work focuses on how to improve deblurring results by imposing various image priors to produce either better estimation of the PSF or to suppress ringing. Dey *et al.* [9] included total variation regularization in the RL algorithm to reduce ringing. Fergus *et al.* [10] demonstrated how to use natural image statistics to estimate a more accurate PSF. Jia [13] computed the PSF from the alpha matte of a motion blurred object. Levin *et al.* [16] introduced the gradient sparsity prior to reduce ringing. Yuan *et al.* [32] proposed a multiscale approach to progressively recover blurred details. Shan *et al.* [26] introduced regularization based on high order partial derivatives to constrain image noise. Evaluations on state-of-the-art deblurring algorithms based on different regularization was presented in Levin *et al.* [18]. This work noted that the assumption of a spatially invariant PSF was invalid in practice.

Other recent methods improve deblurring results by relying on multiple images or special hardware setups. For example, Yuan *et al.* [31] used a pair of noisy and blurry images, while Rav-Acha and Peleg [23] and Chen *et al.* [6] used a pair of blurry images. Raskar *et al.* [22], [1] coded the exposure to make the PSF more suitable for deconvolution. Ben-Ezra and Nayar [3], [4] proposed a hybrid camera to capture a high resolution blurry image together with a low resolution video which is used for estimation of an accurate PSF.

All the previously mentioned work assume the blur is spatially constant. In many cases, however, the PSF spatially varies over the degraded image. Early work by [25] addressed spatially varying motion blur by transforming the image (e.g using log-polar transform) such that the blur could be expressed as a spatial invariant PSF for which traditional global deconvolution could be applied. The range of motion, however, was limited to combined translation and rotation. Another strategy is to segment the blurry image into multiple regions each with a constant PSF. This approach is exploited by Levin [15], Bardsley *et al.* [2], Cho *et al.* [7] and Li *et al.* [19]. Such segmented regions, however, should be small to make the constant PSF assumption valid. This could be difficult even in simple cases of camera rotation or zoom. There are also methods do not require such segmentation. Shan *et al.* [27] handled spatially varying blur by restricting the relative motion to be a global rotation. Joshi *et al.* [14] estimated a PSF at each pixel for images with defocus blur. Special hardware setups have also been used to handle spatially varying blur. Levin *et al.* [17] designed a parabolic camera for deblurring images with 1D motion. During exposure, the camera sensor (CCD/CMOS) moves in a manner to make the PSF spatially invariant. Tai *et al.* [28], [29] extended the hybrid camera framework in [3], [4] to estimate a PSF at each pixel using optical flow.

While previous approaches are related in part, our work is unique for handling motion caused by projective transforms. While projective motion can be handled using existing approaches that use special hardware to estimate per-pixel PSF estimation (e.g. [28], [29]), our approach is a kernel free representation in the sense we do not need to store or compute the local PSF for each pixel. In addition, our deblurring algorithm processes the image as a whole instead of deblurring each pixel with a different PSF or resorting to segmenting into regions that have been blurred with a similar PSF.

III. Projective Motion Blur Model

In this section, we describe our *Projective Motion Blur Model*. This model will be used in section IV to derive the deblurring algorithm.

In photography, the pixel intensity of an image is determined by the amount of light (photons) received by the imaging sensor over the exposure time:

$$I(\mathbf{x}) = \int_0^T \Delta I(\mathbf{x}, t)dt \approx \sum_{i=1}^{N} \Delta I(\mathbf{x}, t_i), \tag{1}$$

where $I(\mathbf{x})$ is the image recorded after exposure; $\Delta I(\mathbf{x}, t)$ is an image captured by the sensor within infinite small time interval dt at time instance t; $[0, T]$ is the total exposure time and $\mathbf{x}$ is a 3×1 vector indicating the homogenous pixel coordinate. In our model, we assume N (the discrete sampling rate over exposure time) is large enough so that the difference between continuous integration

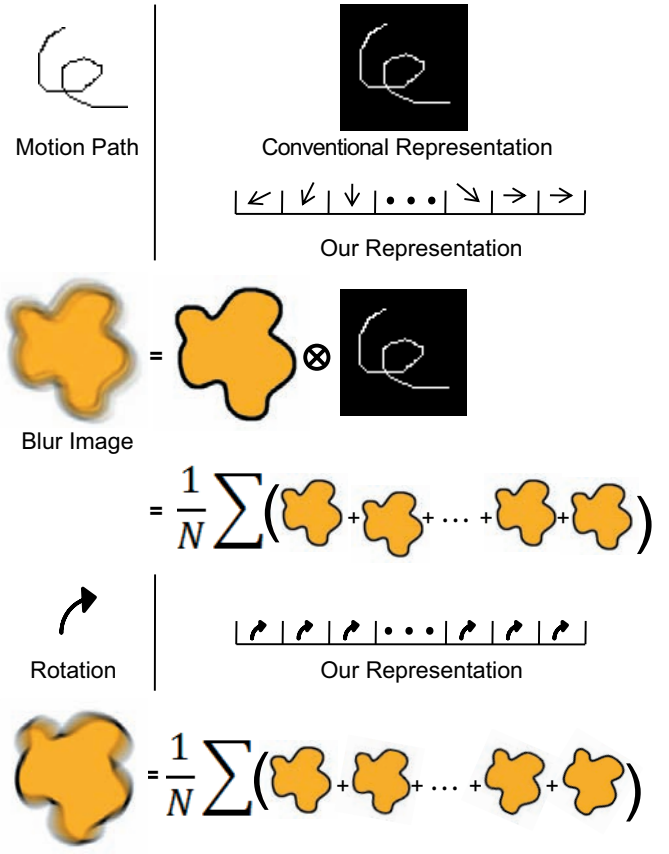


Fig. 3. This figure compares our blur model and conventional model. Given the motion path (PSF), a conventional model uses an image patch (i.e. kernel) to represent the PSF. In comparison, our model uses a sequence of transform matrices. For rotational motions, our representation encodes the rotation in homographies naturally, while conventional approach needs to store pixelwise PSF.

of light (photons) and discrete integration of light (photons) can be neglected.

When there is no relative motion between the camera and the scene, assuming the effects of photons noise is small, $\Delta I(\mathbf{x}, t_1) \cong \Delta I(\mathbf{x}, t_2) \cong \cdots \cong \Delta I(\mathbf{x}, t_N)$ and $I(\mathbf{x}) \cong N\Delta I(\mathbf{x}, t_0) \triangleq I_0(\mathbf{x})$ is a clear image. When there is relative motion, $I(\mathbf{x})$ is the summation of multiple *unaligned* images $\Delta I(\mathbf{x}, t_i)$. For a static distant scene, the relative motion causes a projective transform in the image plane, i.e. $\Delta I(\mathbf{x}, t_i) = \Delta I(\mathbf{h}_i\mathbf{x}, t_{i-1})$. Here, $\mathbf{h}_i$ is a homography[1] defined by a 3×3 non-singular matrix up to a scalar. Suppose we know all $\mathbf{h}_i$, we can then express $\Delta I(\mathbf{x}, t_i)$ by $I_0(\mathbf{x})$ using the following formulation:

$$\Delta I(\mathbf{x}, t_i) = \Delta I(\prod_{j=1}^{i} \mathbf{h}_j\mathbf{x}, t_0) = \frac{1}{N} I_0(\mathbf{H}_i\mathbf{x}), \tag{2}$$

where $\mathbf{H}_i = \prod_{j=1}^{i} \mathbf{h}_j$ is also a homography. Hence, we obtain our *projective motion blur model*:

$$B(\mathbf{y}) = \sum_{i=1}^{N} \Delta I(\mathbf{x}, t_i) = \frac{1}{N} \sum_{i=1}^{N} I_0(\mathbf{H}_i\mathbf{x}), \tag{3}$$

where $B(\mathbf{y})$ is the motion blurred image, and $I_0(\mathbf{x})$ is the clear image we want to estimate. According to our model, the blur image is generated by averaging multiple clear images, each of which is a projective transformed version of the clear image $I_0(\mathbf{x})$. This model inherently assumes that the spatially varying motion PSF within each small time interval dt can be described by a projective transform. This assumption is valid for capturing a static distant scene under camera handshaking motion. Our model fully describes the camera motion which not only models the translational motion of handshaking like a conventional PSF representation, but also models the camera in-plane/out-of-plane rotation which cannot be modeled using a conventional PSF representation.

[1]We use homography for its generality to model all planar transformation. We can also use affine transformation matrix or rotational matrix by making further assumption about the motion path.

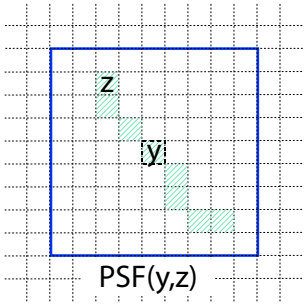


Fig. 4. For each pixel location y, we can define a local point spread function. $PSF(y, z)$ is defined to be the value at pixel location z of a local PSF, defined at pixel location y. Notice that $\sum_z PSF(y, z) = 1$.

Figure 3 illustrates the relationship between our blur model and the conventional representation. The conventional spatial invariant PSF representation is a special case of our model which every $\mathbf{h}_i$ is a translation. The conventional PSF can have intensity variations, which is modeled by the different magnitude of translations in $\mathbf{h}_i$. Note that for in-plane translational motion, the conventional kernel-based model is a significantly more compact representation of the motion blur than our model. However, in the cases of other motions, e.g. in-plane/out-of-plane rotation, our projective motion model is compact and intuitive. Further discussion with regards to the conventional representation are discussed Section VIII.

IV. Projective Motion Richardson-Lucy

In this section, we describe how to modify the Richardson-Lucy algorithm to incorporate our blur model. To do so, we first give a brief review of the Richardson-Lucy algorithm [24], [20] and then derive our algorithm in a similar manner. To make the expression simple, we use I instead of I_0 to represent the clear image to be estimated.

A. Richardson-Lucy Deconvolution Algorithm

The derivation in this section of the Richardson-Lucy Deconvolution Algorithm [24], [20] is based on the original paper from Richardson [24]. According to [24], given the motion blurred image B, the clear image I is computed by a Bayesian estimation. The pixel value $I(\mathbf{x})$ is computed according to pixels values $B(\mathbf{y})$ in the blurry image by the following formula:[2]

$$P(I_x) = \sum_y P(I_x|B_y)P(B_y). \tag{4}$$

[2]We will use I_x, B_y to represents $I(\mathbf{x}), B(\mathbf{y})$ in this section for clarity.

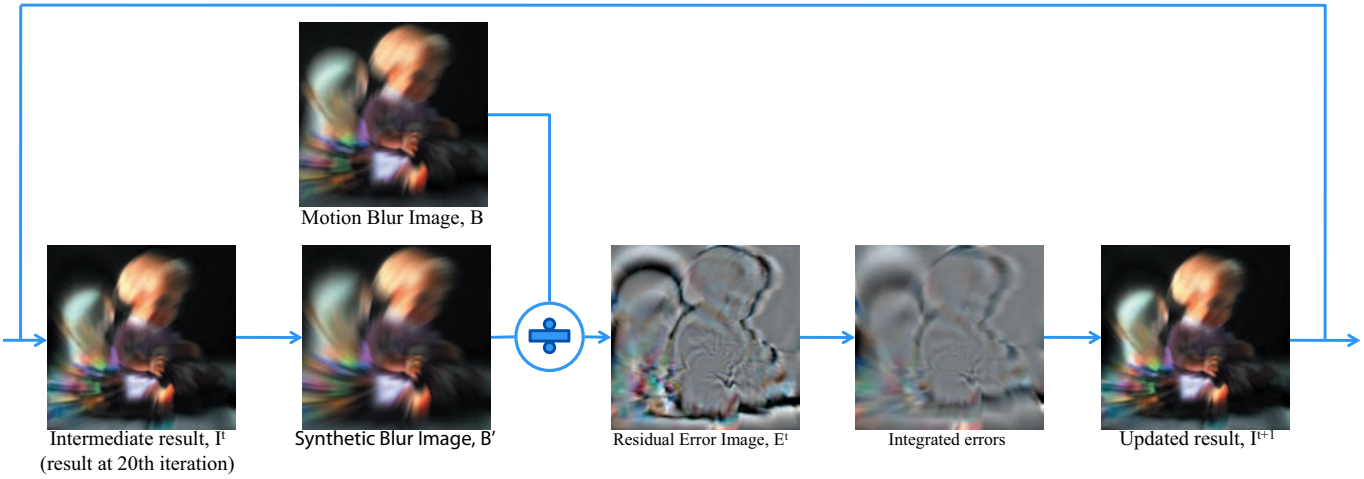


Fig. 5. Overview of our Projection Motion RL algorithm. Given the current estimation I^t, we compute a synthetic blur image B' according to the given motion in terms of H_i. The residual error image $E^t = B/B'$ is computed by pixel-wise division (substraction for Gaussian noise model). The residual errors are then integrated according to H_i^{-1} to produce an updated estimation I^{t+1}. The I^{t+1} is then used as initial guess for the next iterations. This process is iterated until convergence or after a fixed number of iterations. In our implementation, the number of iterations is 500.

where $P(I_x|B_y)$ can be computed by Bayes's rule, and $P(I_x)$ can be written as:

$$P(I_x) = \sum_y \frac{P(B_y|I_x)P(I_x)}{\sum_z P(B_y|I_z)P(I_z)} P(B_y). \tag{5}$$

Both side of this equation contains $P(I_x)$, which $P(I_x)$ on the right side is also the desired solution. To break the dependency, a general and acceptable practice [21] is to make the best of a bad situation and use a current estimation of $P(I_x)$ to approximate $P(I_x|B_y)$. Hence, we can define an iterative update procedural algorithm:

$$\begin{aligned} P^{t+1}(I_x) &= \sum_y \frac{P(B_y|I_x)P^t(I_x)}{\sum_z P(B_y|I_z)P^t(I_z)} P(B_y) \\ &= P^t(I_x) \sum_y P(B_y|I_x) \frac{P(B_y)}{\sum_z P(B_y|I_z)P^t(I_z)} \end{aligned} \tag{6}$$

where t is an iteration index. Equation (6) can be reduced to a more easily workable form by defining $P^t(I_z) = I_z^t$ to be the value of a clear image at pixel location z at t-th iteration. We can further define $P(B_y|I_z) = PSF(y,z)$ and $\sum_z PSF(y,z) = 1$, where $PSF(y,z)$ is the value at pixel location z of a point spread function, defined locally at pixel location y. Figure 4 illustrates the relationship between y and z within a local window of a point spread function. Note that for spatially invariant PSF, $PSF(y,z)$ is the same for all y.

To understand Equation (6), we can consider that $B' = \sum_z P(B_y|I_z)P^t(I_z)$ is the prediction of a blurry image according to the current estimation of clear image I^t and the given point spread function. We can also define $E_y = B_y/B'_y$ as the residual errors (by pixel-wise division) between the real blurry image B and the predicted blurry image B'. Hence, $\sum_y P(B_y|I_x)\frac{P(B_y)}{\sum_z P(B_y|I_z)P^t(I_z)} = \sum_y P(B_y|I_x)E_y^t$ is the amendment term according to the residual, which integrate the residual errors distributed within the local window of the PSF according to $P(B_y|I_x)$. Note that the index of summation is different with that of the generation of predicted blurry image, its summation index is z, while for the integration of errors, its summation index is y. The update rule in Equation (6) essentially computes a clear image I^∞ that would generate the blurry image B given a known point spread function. Typically, the algorithm start with an initial guess of $I^0 = B$.

When there is a spatially invariant PSF K, B' can be computed as $I^t \otimes K$, and $\sum_y P(B_y|I_x)E_y$ will be $K * E_y$ where $*$ and $\otimes$ are the correlation and convolution operators respectively. Hence, Equation (6) can then be presented as[3]:

$$I^{t+1} = I^t \times K * \frac{B}{I^t \otimes K} = I^t \times K * E^t, \tag{7}$$

where $E^t = \frac{B}{I^t \otimes K}$.

The Richardson-Lucy algorithm is well-known to minimize the difference between the original blurry image B and the predicted blurry image $B' = I \otimes K$ according to the Poisson noise model, i.e.

$$\arg\min_I n(B', B) \tag{8}$$

where $n(\lambda, k) = e^{-\lambda}\lambda^k/k!$, $\lambda = B'$, $k = B$ is the probability distribution function (pdf) of a Poisson distribution. Note that image noise are assumed to be independent and identically distributed (i.i.d.). Hence, $n(B', B)$ is accessed per-pixel. In [20], Lucy shows that based on the Poisson noise distribution, $K * \frac{B}{I^t \otimes K}$ converges to 1 and hence $\frac{I^{t-1}}{I^t} \to 1$ as $t \to \infty$ which is taken as a proof that the Richardson-Lucy algorithm converges.

If we suppose that the image noise follow a Gaussian distribution instead of a Poisson distribution, we can model $n(B', B) = \exp(-\frac{||B-B'||^2}{2\sigma^2})$, where σ is the standard deviation of the Gaussian noise model. To maximize $P(I_x)$ under a Gaussian noise model, we can minimize $-\log P(I_x)$ which converts the problem into a least square minimization problem: $\arg\max_I n(B', B) = \arg\min_I ||B - B'||^2$. Taking log function on both side of Equation (7) and we re-define the variables, we obtain:

$$I^{t+1} = I^t + K * (B - I^t \otimes K) = I^t + K * E^t. \tag{9}$$

where $E^t = B - I^t \otimes K$ is the residual errors computed by pixel-wise substraction. The convergence of Equation (9) has been proved in [12].

[3]Details to the rearrangement of the index coordinates of Equation (6) for a global PSF can be found in [24]

B. Projective Richardson-Lucy algorithm

With the basic Richardson-Lucy algorithm, we can derive our Projective Motion Richardson-Lucy algorithm based on the conventional Richardson-Lucy algorithm. In fact, Equation (6) does not assume that the PSF needs to be the same at each pixel. When the PSF varies spatially and if the variations satisfied our projective motion blur model, we have:

$$P(B_y|I_x) = \begin{cases} \frac{1}{N}, & \mathbf{y} = \mathbf{H}_i\mathbf{x} \\ 0, & otherwise \end{cases} \tag{10}$$

where $\sum_x P(B_y|I_x) = \sum_{i=1}^{N} P(\mathbf{y} = \mathbf{H}_i\mathbf{x}) = 1$. Note that according to this definition, $P(B_y|I_x)$ does not necessary corresponding to a discrete integer coordinate of a pixel location, its location can be fractional according to the motion defined by $\mathbf{y} = \mathbf{H}_i\mathbf{x}$. We use bicubic interpolation to compute the value of $I(\mathbf{H}_i\mathbf{x})$. We finally derive $\sum_{i=1}^{N} P(\mathbf{y} = \mathbf{H}_i\mathbf{x})I(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} I(\mathbf{H}_i\mathbf{x})$ which is the equation of our projective motion blur model in Equation (3) for generating the prediction of a blurry image according to the clear image I and the given camera motion in terms of $\mathbf{H}_i$.

We substitute $B'(\mathbf{y}) = \frac{1}{N}\sum_{i=1}^{N} I^t(\mathbf{H}_i\mathbf{x})$, $E_y = B_y/B'_y$ and $\sum_y P(B_y|I_x) = \sum_{i=1}^{N} P(\mathbf{x} = \mathbf{H}_i^{-1}\mathbf{y})$[4] into Equation (6). After simplification, we obtain our iterative update rules for the projective motion blur model:

$$I^{t+1}(\mathbf{x}) = I^t(\mathbf{x}) \times \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}), \tag{11}$$

where $E^t(\mathbf{x}) = \frac{B(\mathbf{x})}{\frac{1}{N}\sum_{i=1}^{N} I^t(\mathbf{H}_i\mathbf{x})}$. Similarly, if the image noise model follows a Gaussian distribution, our iterative algorithm becomes

$$I^{t+1}(\mathbf{x}) = I^t(\mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}), \tag{12}$$

where $E^t(\mathbf{x}) = B(\mathbf{x}) - \frac{1}{N}\sum_{i=1}^{N} I^t(\mathbf{H}_i\mathbf{x})$. Our approach has essentially replaced the global correlation and convolution operators in the original RL algorithm with a sequence of forward projective motions and their inverses. Figure 5 shows the processes of our projective motion RL algorithm.

V. Adding Regularization

The result derived in the previous section is a maximum likelihood estimation with respect to Poisson or Gaussian noise models and imposes no regularization on the solution. Recent deblurring work [9], [16], [32] have shown that imposing certain image priors can be used to significantly reduce ringing. In this section, we discuss how to incorporate these regularization terms in the Projective Motion RL algorithm. In particular, we describe how regularization based on total variation [9], Laplacian [16], and the bilateral regularization in [32] can be incorporated into our algorithm. Note that the goal of this section is not to compare and evaluate the performance of different regularization in the deconvolution process. Instead, we want to show the capability of our Projective Motion RL to include regularization terms that have been used with the conventional RL. In addition, we discuss implementation details that allow the regularization terms to be more effective and allows us to avoid the tuning of the parameter, λ, that control the relative weight of regularization.

[4] $\sum_y P(B_y|I_x) = \sum_y P(\mathbf{y} = \mathbf{H}_i\mathbf{x}) = \sum_{i=1}^{N} P(\mathbf{x} = \mathbf{H}_i^{-1}\mathbf{y})$

Total variation regularization The total variation (TV) regularization was introduced by Dey *et al* [9]. The purpose of introducing this regularization is to suppress image noise amplified during the deconvolution process by minimizing the magnitude of gradients in deblurred image:

$$R_{TV}(I) = \int \sqrt{||\nabla I(\mathbf{x})||^2} d\mathbf{x} \tag{13}$$

where $\nabla I(\mathbf{x})$ is the first order derivative of $I(\mathbf{x})$ (in x and y direction). Substituting this regularization term into our Projective Motion RL algorithm, we get a regularized version of the update rules:

$$I^{t+1}(\mathbf{x}) = \frac{I^t(\mathbf{x})}{1 - \lambda\nabla R_{TV}(I)} \times \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}) \tag{14}$$

and

$$I^{t+1}(\mathbf{x}) = I^t(\mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}) + \lambda\nabla R_{TV}(I) \tag{15}$$

where $\nabla R_{TV}(I) = -\nabla\frac{\nabla I^t}{|\nabla I^t|}$ and λ is a parameter controlling the weight of regularization. As reported in [9], $\lambda = 0.002$[5] is used in their experiments.

Laplacian regularization The Laplacian regularization, is sometimes called the sparsity regularization, assert that for natural images its histogram of gradient magnitudes should follows a heavy-tailed distribution such as the Laplacian distribution. The Laplacian regularization, suggested by [16], it takes the following form:

$$R_L(I) = \exp(-\frac{1}{\eta}|\nabla I|^d), \tag{16}$$

where d is a parameter controlling the shape of distribution, and the term $\eta = 0.005$ (according to [16]) is the variance of the image noise. Note that Equation (16) is a Gaussian distribution when $d = 2$. In [16], d is set to 0.8. In our implementation and the source code submitted, we follow the same parameter setting with $d = 0.8$ and $\eta = 0.005$ for fairness of comparisons. The effects of Laplacian regularization is also to suppress noise and to reduce small ringing artifacts. However, this regularization tends to produce sharper edges as opposed to the total variation regularization which tends to produced over-smooth results.

Adding this regularization into our projective motion RL algorithm, we obtain another set of update rules:

$$I^{t+1}(\mathbf{x}) = \frac{I^t(\mathbf{x})}{1 - \lambda\nabla R_L(I)} \times \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}) \tag{17}$$

and

$$I^{t+1}(\mathbf{x}) = I^t(\mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}) + \lambda\nabla R_L(I) \tag{18}$$

where

$$\nabla R_L(I) = -\frac{1}{\eta}\exp(-\frac{1}{\eta}|\nabla I|^d)|\nabla I|^{d-1}\nabla^2 I. \tag{19}$$

A typical value for λ (according to the implementation of [16]) is between 0.001 to 0.004[6]. We note that in the implementation

[5] This is an un-normalized weight, a normalized weight equal to $0.002 \times 255 = 0.51$.

[6] These parameter values are also un-normalized, a normalized weight should be between 0.25 to 1.0.

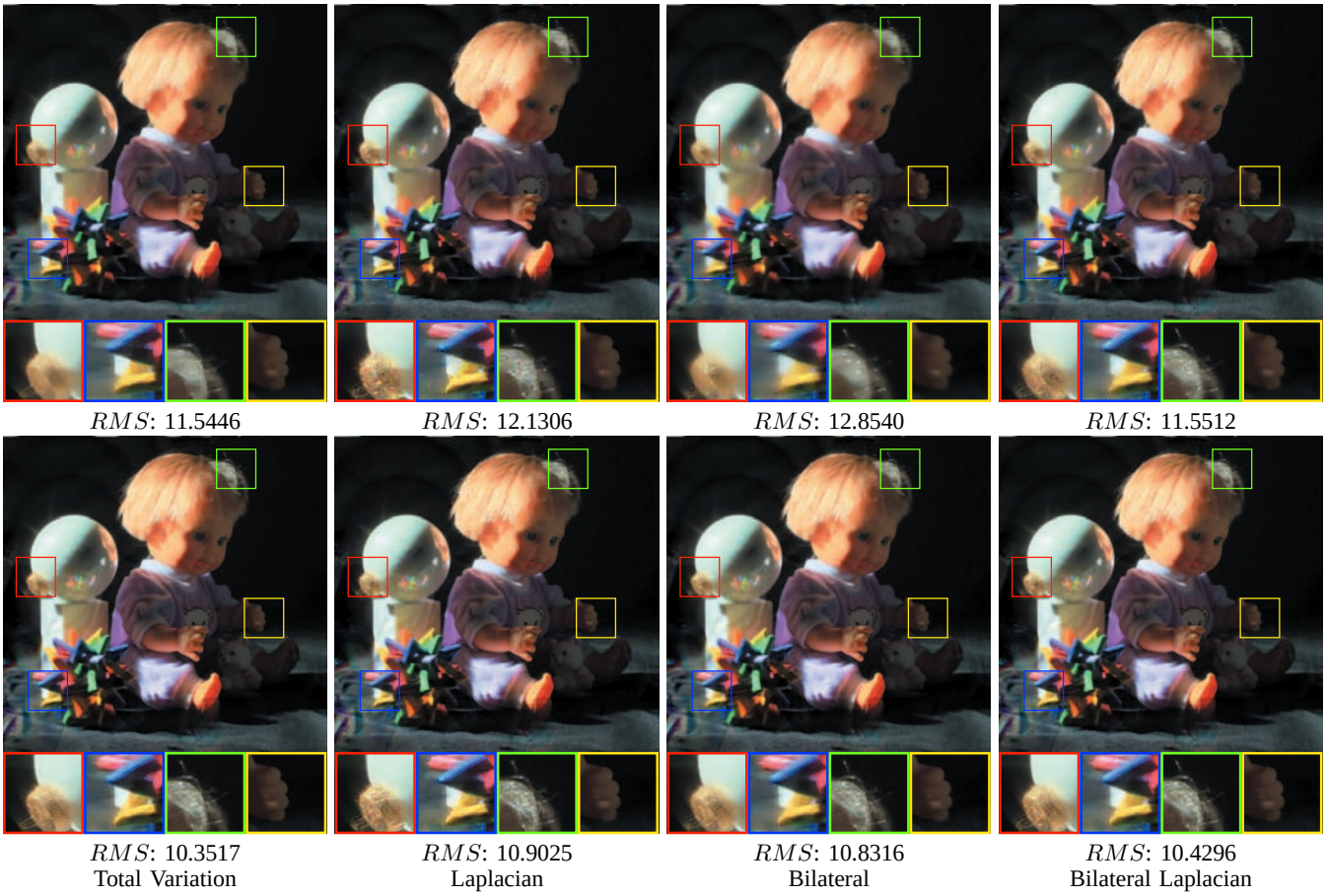


Fig. 6. Comparisons on different regularization: Total Variation (TV) regularization, Laplacian regularization, Bilateral regularization and Bilateral Laplacian regularization. The results on the top row are produced with fixed $\lambda = 0.5$; The results on the bottom row are produced with our suggested implementation scheme which λ decreases progressively as iterations increase. The number of iterations and parameters setting (except λ) are the same for all testing case. Our implementation scheme is effective, the deblurring quality in terms of both visual appearance and RMS errors have been improved. The input blur image and ground truth image can be found in Figure 1(a) and (d) respectively.

of [16] uses a slightly different weighting scheme:

$$\nabla R_L(I) = -\frac{1}{|\nabla I|^{d-2}}\nabla^2 I, \tag{20}$$

The effects of the two weighting schemes in Equation (19) and Equation (20), however, are similar with larger regularization weight given to smaller gradient and vice versa. In our implementation and source code submitted, we use the weighting scheme in Equation (19).

Bilateral regularization In order to suppress ringing artifacts while preserving sharp edges, Yuan *et al.* [32] proposed an edge-preserving bilateral regularization cost:

$$R_B(I) = \sum_{\mathbf{x}} \sum_{\mathbf{y}\in N(\mathbf{x})} g_1(||\mathbf{x}-\mathbf{y}||^2)(1-g_2(||I(\mathbf{x})-I(\mathbf{y})||^2)), \tag{21}$$

where $N(\mathbf{x})$ is local neighborhood of $\mathbf{x}$, $g_1(\cdot)$,$g_2(\cdot)$ are two Gaussian functions with zero mean and variance of σ_s^2,σ_r^2 respectively. In [32], the deblurring process is performed in multi-scale and $\sigma_s^2 = 0.5$ is the spatial variance and it is set to be variance of Gaussian blur kernel from one level to another level. The term σ_r^2 is the range variance and it is set to be $0.01 \times |\max(I) - \min(I)|$. The size of local neighborhood $N(\mathbf{x})$ is determined by σ_s.

In [32], ringing artifacts are progressively suppressed by both inter-scale and intra-scale regularization. The bilateral regularization corresponds to the inter-scale regularization. Our iterative update rule with the bilateral regularization term are derived as [32]:

$$I^{t+1}(\mathbf{x}) = \frac{I^t(\mathbf{x})}{1-\lambda\nabla R_B(I)} \times \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}), \tag{22}$$

and

$$I^{t+1}(\mathbf{x}) = I^t(\mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} E^t(\mathbf{H}_i^{-1}\mathbf{x}) + \lambda\nabla R_B(I), \tag{23}$$

where

$$\nabla R_B(I) = \sum_{y\in N(\mathbf{x})} (I_y^d - I_y^d D_y), \tag{24}$$

$$I_y^d(\mathbf{x}) = g_1(||\mathbf{x}-\mathbf{y}||^2) g_2(||I(\mathbf{x}) - I(\mathbf{y})||^2) \frac{(I(\mathbf{x}) - I(\mathbf{y}))}{\sigma_r}. \tag{25}$$

The term D_y is a displacement operator which shifts the entire image I_y^d by the displacement vector $(\mathbf{y}-\mathbf{x})$, and λ (as reported in [32]) is set to be a decay function which its weight decreases as iterations increases. The effects of this bilateral regularization is similar to the effects of Laplacian regularization[7]. However, we found that the bilateral regularization tends to produce smoother

[7]The two regularization are essentially the same if $N(\mathbf{x})$ is first order neighborhood and $g_2(\cdot)$ is a Laplacian function

result not only because they use a Gaussian distribution for $g_2(\cdot)$, but also because the larger local neighborhood size $N(\mathbf{x})$ produce smoothing effects. In our implementation, we also include a regularization where $g_2(\cdot)$ follows a Laplacian distribution. We call this regularization a "Bilateral Laplacian" regularization. The parameters of $g_2(\cdot)$ is set to be the same as in [16] for fair comparisons.

Additional Implementation Details One major challenge in imposing regularization in a deblurring algorithm is adjustment of parameters. The most important parameter is the term λ which adjust the relative weight between the data term and the regularization term. If λ is too large, details in the deblurred results can be overly smoothed. On the other hand, if λ is too small, ringing artifacts cannot be suppressed.

In our experiments, we found that having a large initial λ that progressively decrease with increasing number of iterations produces the best deblurring results. This implementation scheme has been used in [26], [32] and praised by [18] in regards to its effectiveness. In our implementation, we divided the iterations into five sets each containing 100 iterations. The λ are set to be $1.0, 0.5, 0.25, 0.125, 0.0$ in each set. We run our deblurring algorithm starting from the largest λ to the smallest λ. One interesting observations we found is that under this implementation scheme the total variation regularization produces the best results in terms of root-mean-square (RMS) error. One explanation is that the Laplacian and Bilateral regularization tend to protect edges which that arise from ringing artifacts.

The visual appearance of deblurring results with different regularization are similar, and all of the above regularization can successfully reduce ringing artifacts. One important thing to notice is that in the last set of iterations, we do not impose any regularization but we start with a very good initialization that are produced from the regularized deblurring algorithm. We found that even without regularization, having a very good initialization can effectively avoid ringing artifacts. Further, image details that have been smoothed out by regularization can be recovered in the last set of iterations.

VI. Motion Estimation

Although the focus of this paper is the derivation of our projective motion path blur model and its associated deblurring algorithm, for completeness we describe two methods to compute the projective motion within the exposure time. The first one makes use of auxiliary hardware during image capture while the second one assumes uniform motion. General algorithms for motion estimation is part of future research.

Auxiliary Hardware A direct method to estimate camera motion in terms of homographies during the exposure is to use a hybrid camera [4], [29] that captures an auxiliary high-frame-rate low-resolution video. From the auxiliary video, motion trajectory at each pixel (i.e. optical flow) can be computed as done in [29] and sequence of homographies can be fitted to each frame. Another promising hardware coupling is the use of accelerometers for estimating the device motion as demonstrated in [11].

Uniform Motion Assumption In this method, we assume the motion blur is caused by a constant projective motion over the exposure time. Hence, $\mathbf{h}_1 = \mathbf{h}_2 = \cdots = \mathbf{h}_N$. According to the definition of $\mathbf{H}_i = \prod_{j=1}^{i} \mathbf{h}_j$, we can get

$$\mathbf{h}_i = \sqrt[N]{\mathbf{H}_N} \qquad 1 \leq i \leq N. \tag{26}$$

Thus, we can obtain $\mathbf{h}_i$ by computing the N-th matrix root [5] of $\mathbf{H}_N$, and the estimation of the series of $\mathbf{h}_i$ is reduced to the estimation of $\mathbf{H}_N$. The easiest technique, described in Shan *et al.* [27], simply relies on the user to supply image correspondences to establish the transformation. Another highly promising technique is that proposed by Dai and Wu [8] that uses the blurred objects alpha matte to estimate $\mathbf{H}_N$. In our experiments, we use a former approach in [27].

VII. Experiment Results

In this section, we empirically examine our algorithms convergence by plotting the RMS error against iterations. Robustness is analyzed by comparing the results with different amount of additive noise. We will also evaluate the quality of our Projective Motion RL with and without regularization by creating a set synthetic test cases. Finally, we shows results on real images which the motion PSF were estimated using methods suggested in Section VI.

A. Convergence Analysis

While the conventional RL algorithm guarantees convergence, in this section, we empirically examine the convergence of our projective motion RL algorithm. At each iterations, we compute the RMS error of the current result against ground truth image. We run our algorithm for a total of 5,000 iterations for each case. The convergence rate of our projective motion RL for both the Poisson noise distribution and the Gaussian noise model are compared. Figure 7(a) shows the graphs plotting the RMS errors against the number of iterations.

Typically, our method "converges" within 300 to 400 iterations for both the Poisson noise model and Gaussian noise model with which the RMS errors difference between successive iterations is less than 0.01. The convergence rate of Poisson noise model is faster than the Gaussian noise model. The Poisson noise model typically produce better results in terms of RMS errors than that of Gaussian noise model. However, the visual appearance difference between the converged results of the two models are visually indistinguishable. As the number of iterations increases, the difference of RMS error between successive iterations decreases, however, after 500 iterations the quality in the deblurred result is unnoticeable. Figure 7(b)-(f) shows some intermediate results during the iterations.

B. Noises Analysis

To test for the robustness of our algorithm, we added different amounts of zero-mean Gaussian noise to the synthetic blur images. Note that the image noise added does not undergo the convolution process, hence it is independent of the motion blur effect. This to to approximate sensor noise effect in real world situation.

Figure 8 shows our deblurring results with different amounts of Gaussian noise added. We shows both results of our algorithm with and without regularization. For this experiment, we use the total variation regularization. As expected, our deblurring results without regularization amplifies image noise like other deblurring

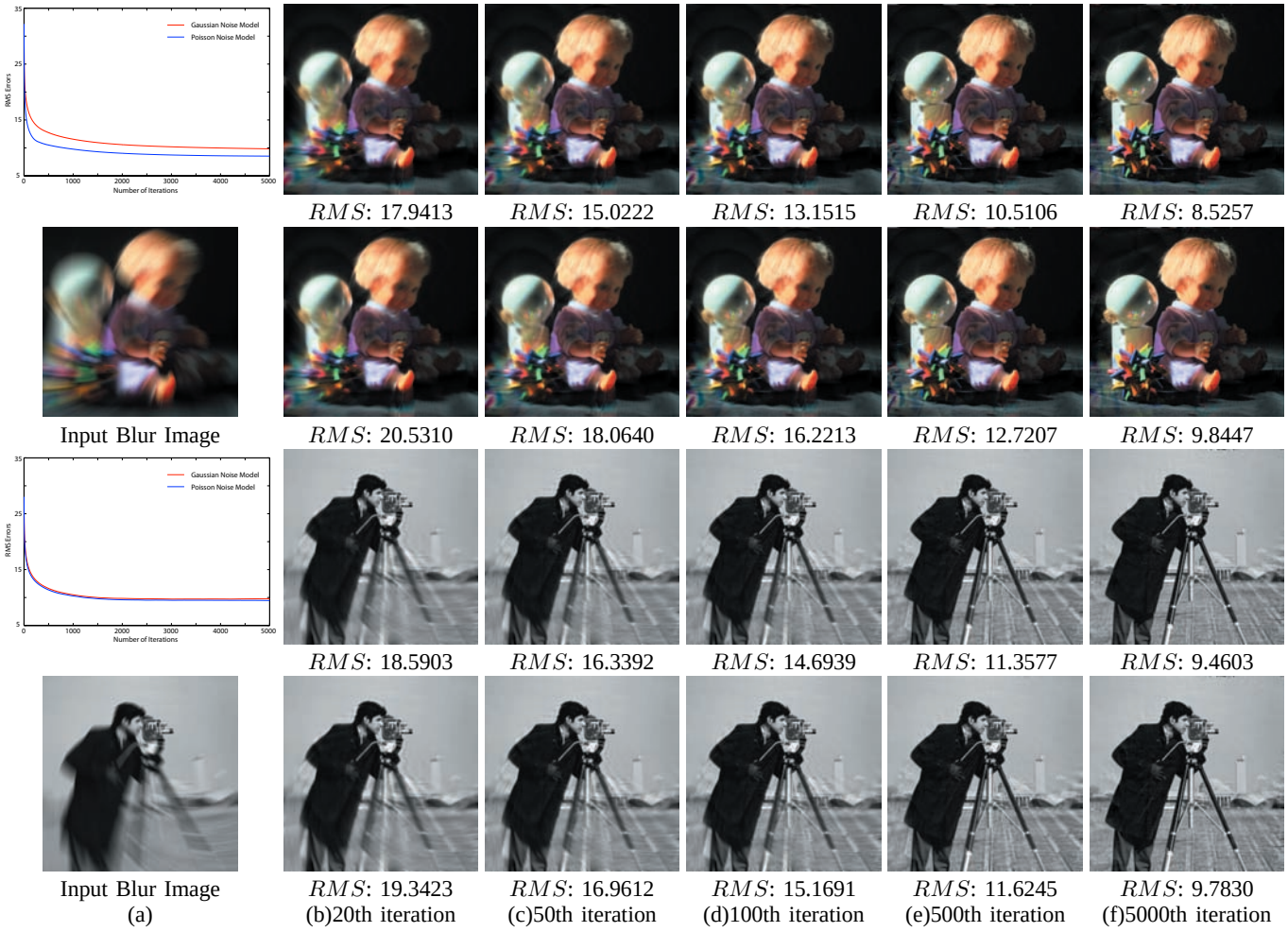


Fig. 7. Convergence rates of our projective motion RL algorithm. (a) The plot of RMS errors against number of iterations and the input motion blur image. Note that the motion blur are different for the two test cases. (b)-(f) Intermediate results at the 20th, 50th, 100th, 500th and 5000th iterations are shown. The first and third row are results using Poisson noise model and the second and fourth row are results using Gaussian noise model.

algorithms. The quality of our deblurring algorithm degrades as the amount of noise increased. We found that larger motions tend to produce more noisy results. In such cases, the noise in the image centers around the center of rotation and becomes less apparent in image regions with larger motion blur (i.e. the image boundaries). In the presence of image noise, the regularization term becomes important to suppressed amplified image noise. Better results can be achieved by increasing the regularization as the noise level increased. However, for fairness of comparisons, we use the same regularization weights as described in our implementation details discussed in Section V. The difference between the regularized and un-regularized results are significant both in terms of visual quality and RMS errors. However, when the amount of image noise added is very large, e.g. $\sigma^2 \geq 20$, the regularization term cannot suppressed image noise effectively. Despite the amplified image noise, our deblurring algorithm can still recover the "lost" image details that cannot be seen in the motion blurred images. The deblurred result without added noise is show in Figure 1 for comparison.

C. Qualitative and Quantitative Analysis

In this section, we demonstrate the effectiveness of our Projective Motion RL algorithm on both synthetic data and real data. Figure 1 has already shown a synthetic example of spatially varying motion blur with known $\mathbf{h}_i$.

To further evaluate our algorithm quantitatively, we have created a test set consisting of fifteen test cases and five test images: *Mandrill*, *Lena*, *Cameraman*, *Fruits* and *PAMI*. The *Mandrill* example contains significant high frequency details in the hair regions. The *Lena* example contains both high frequency details, smooth regions and also step edges. The *Cameraman* image is a monotone image, but the additional noise we added are not monotone. The *Fruits* example also contains many high frequency details and smooth regions. Lastly, the *PAMI* is a text-based images which its ground truth image contains black and white colors only. For each test case, we add additional Gaussian noise ($\sigma^2 = 2$) to simulate camera sensor noise. The parameters setting are the same for all test cases and the values we used are the same as in the implementation of [16]. The exact parameter values can be found in our submitted source code. We use the implementation scheme discussed in Section V for adjusting the regularization weight λ. While we found that better results can be produced with parameters tuning, we do not tune the parameters for the fairness of comparisons and evaluations.

We show the RMS of the input motion blurred image, the RMS of deblurred image using our basic Projective Motion RL

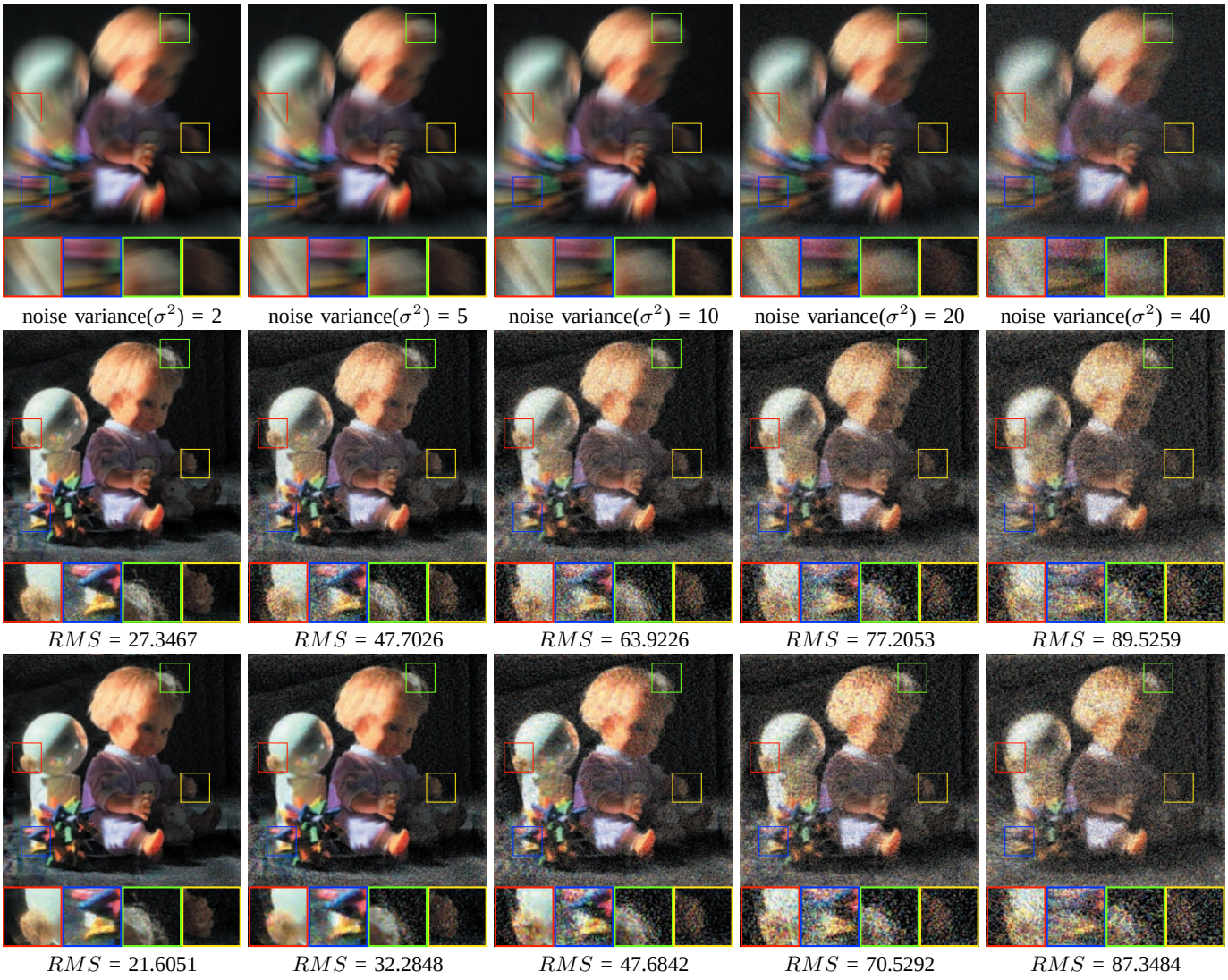


Fig. 8. Evaluation on the robustness of our algorithm by adding different amount of noise in blurred images. Top row: noisy blurred image, the amplitude of noise is determined by the noise variance (σ^2). Second row: Deblurring results with our basic projective motion RL algorithm. Third row: Deblurring results with total variation regularization. In the presence of image noise, our deblurring algorithm amplified the image noise in deblurred results. The effects of regularization hence become significant in suppressing amplified image noise.

algorithm and the RMS of the deblurred image with (Total Variation) regularization are provided. Figure 9 shows our results. Our Projective Motion RL is effective, especially when regularization is used with our suggested implementation scheme. These test cases also demonstrate that our Projective Motion RL is effective in recovering spatially varying motion blur that satisfied our Projective Motion Blur Model assumption. We note that in some test case, e.g. test case of *Fruits* example, the RMS errors of the deblurred results is larger than the RMS errors of input motion blurred images. This is due to the effects of amplified noise. After noise suppression with regularization, the RMS errors of the deblurred results are smaller that the RMS errors of input motion blurred images. High-resolution images of all these test cases have been submitted as supplemental materials. Some test cases are presented in Figure 13.

To test our algorithm on real input data, we obtain the homographies by the two different approaches suggested in Section VI. Figure 10 shows an example of global motion blur obtained from our previous work using a hybrid-camera system [29]. For this example, we show that our Projective Motion RL algorithm can handle the traditional global motion blur as long as we can estimate the motion vectors. We also show the effectiveness of our regularization compared with previous results. Our approach achieves comparable result with [29], however we do not use the low-resolution images as regularization as done [29]. Furthermore, unlike [29] who derived a PSF per pixel, our deblurring process is kernel free.

Figure 11 and Figure 12 show more real examples with zoom-in motion and rotational motion respectively. The motion blurred matrix $\mathbf{H}_N$ is obtained by fitting the transformation matrix with user markup as shown in Figure 11(a) and Figure 12(a) respectively. Each $\mathbf{h}_i$ is computed by assuming the motion is a uniform motion. We show the deblurred results from our algorithm without and with regularization respectively in (b) and (c). The ground truth image is shown in (d) for comparisons. We note that our real examples contains more ringing artifacts than synthetic examples. This is due to small estimation errors in $\mathbf{h}_i$. The effects of image noise in our real examples are not as significant as in our synthetic

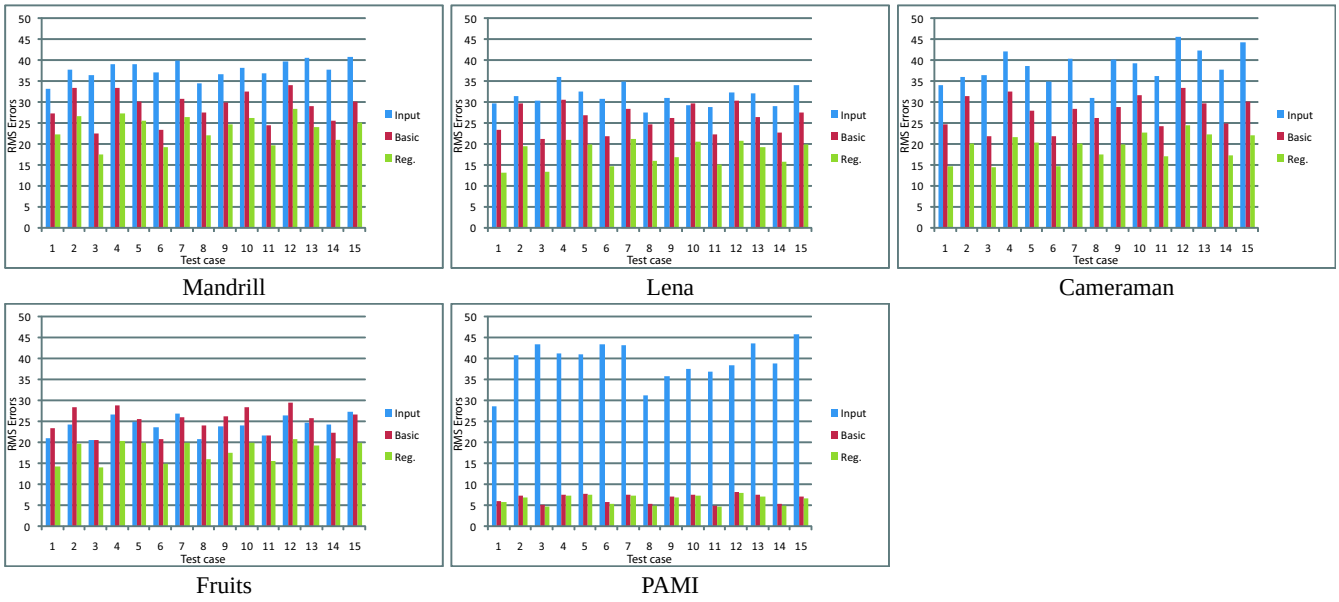


Fig. 9. RMS pixel errors for different example and test cases in our synthetic examples. We compare the RMS of the input blur image (blue), the deblurred image using basic Projective Motion RL (red) and the deblurred image using Projective Motion RL with (Total Variation) regularization (green). These test cases show that our Projective Motion RL is effective in dealing with rigid spatially varying motion blur. **All tested images and results are available in the supplemental materials.**

test case because long exposure time minimized the amount of image noise and also the motions in our real examples are not as large as in our synthetic example.

VIII. Discussion and Conclusion

This paper has introduced two contributions to motion deblurring. The first is our formulation of the motion blur as an integration of the scene that has undergone a projective motion path. While a straight-forward representation, we have found that this formulation is not used in image deblurring. The advantages of this motion blur model is that it is kernel-free and offers a compact representation for spatially varying blur due to projective motion. In addition, this kernel free formulation is intuitive with regards to the physical phenomena causing the blur. Our second contribution is an extension to the RL deblurring algorithm to incorporate our motion blur model in a correction algorithm. We have outlined the basic algorithm as well as details on incorporating state-of-the-art regularization. Our experimental result section has demonstrated the effectiveness of our approach on a variety of examples.

In this section, we discuss issues related to our Projective Motion Blur Model and our Projective Motion RL algorithm. In particular, we will discuss the pros and cons of our Projective Motion Blur Model in comparing with conventional PSF representation. We will also discuss limitations of our approaches and suggest some future research direction based on our model.

A. Conventional PSF representation versus Projective Model Blur Model

For global in-plane motion blur, the conventional representation of the PSF consists of a square matrix (i.e. kernel) that describes the contributions of pixels within a local neighborhood. This representation has several advantages. First, it is easy to understand and provides an intuitive represents of "point spread" about a point. Second, this model handles other global blurring effects (e.g. defocus) and motion blur in a unified representation, while our representation only targets motion blur. Third, by assuming the motion blur is globally the same, image deconvolution can be done in the frequency domain, while our Projective Motion RL algorithm can only be done in the spatial domain. This conventional representation, however, is not suitable for dealing with spatially varying motion blur. For such cases, our Projective Motion RL formulation becomes advantageous.

B. Other deblurring methods based on Projective Motion Model

While we have developed our deblurring algorithm based on Richardson-Lucy, we note that other algorithmic framework for motion deblurring could be used. For example, using our projective motion model, we could construct a system of linear equations in the form of $y = Ax + n$ where each row of A is the contribution of the pixels in the unblurred image x to the corresponding blurred pixel y. The term n could be used to represent noise. This would require a matrix of dimension $(N \times N)$, where N is the size of the image. While such a massive linear system is undesirable, the sparse nature of the matrix would allow sparse matrix approaches such as that demonstrated by Levin *et al.* [16]. Such methods do not require A to be explicitly represented, only the the evaluation of the matrix product to be constructed. How to incorporate regularization into such a method is not entirely straight-forward and one reason we have worked from the RL algorithm.

C. Limitations

Our projective motion RL algorithm has several limitation similar to other deblurring algorithms. A fundamental limitation to our algorithm is that the high frequency details that have been lost during the motion blur process cannot be recovered. Our algorithm can only recover the "hidden" details that remain inside the motion blur images. Another limitation is that our

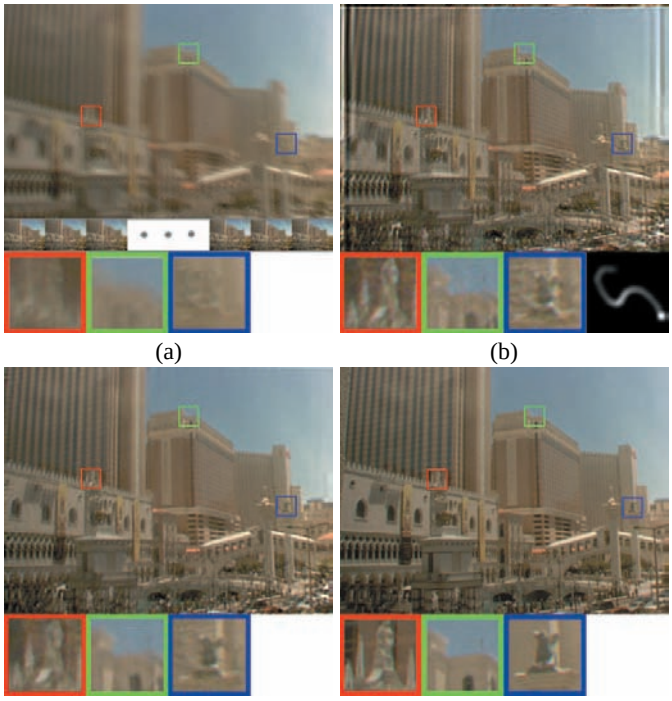


Fig. 10. Image deblurring using globally invariant kernels. (a) Input from a hybrid camera (courtesy of [29]) where the high-frame-rate low resolution images are also shown; (b) Result generated by [3](Standard RL algorithm from Matlab); (c) Result from our Projective Motion RL with regularization. (d) The ground truth sharp image. Close-up views and the estimated global blur kernels are also shown.

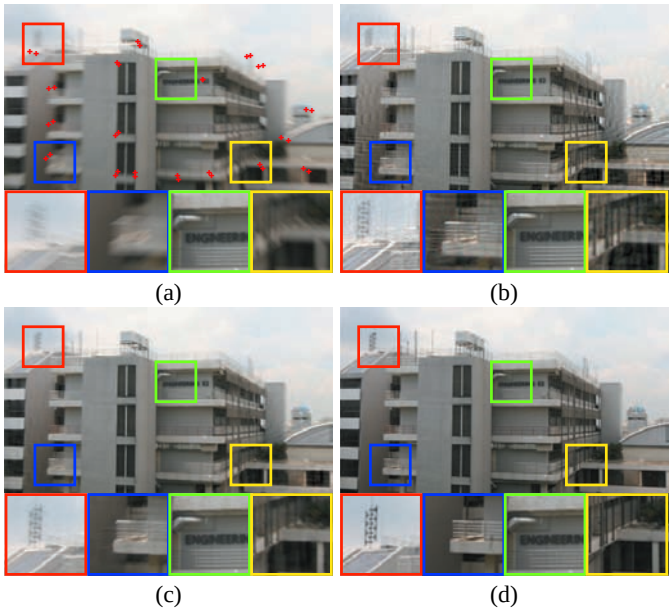


Fig. 11. (a)Input blur image, user's markups of correspondents for motion estimation are also shown, (b) Our result from the basic algorithm; (c) Our result with regularization; (d) Ground truth.

approach does not deal with moving or deformable objects or scene with significant depth variation. A preprocessing step to separate moving objects or depth layers out from background is necessary to deal with this limitation. Other limitations of our approach include the problem of pixel color saturations and severe image noise.

D. Running Time Analysis

Our current implementation takes about 15 minutes to run 500 iterations on an image size 512×512 with Intel(R)CPU L2400@1.66Ghz and 512MB of RAM. The running time of our algorithm depends on several factors, including image size $|I|$, number of discrete sampling N (number of homographies) and the number of iterations T. Hence, the running time of our algorithm is $O(|I|NT)$. We note that most of our running time in our implementation is spent in the bicubic interpolation process to generate the synthetic blur image and to integrate the residual errors. One possible solution to speed up our process is to use GPU for the projective transformation of each dt.

E. Future Directions

There are several major future directions of this work. One is to explore other existing algorithms and hardware (e.g. coded exposure and coded aperture) to use this blur formulation instead of assuming a global PSF. Another important direction is to consider how to use our framework to perform blind-deconvolution where the rigid motion is unknown. Estimating a piecewise projective camera motion is very different from estimating the global PSF in a conventional deblurring algorithm. However, if we assume the motion satisfied our Projective Motion Blur Model, we may be able to impose constrains that can estimate the projective motion path. Other possible directions is to include more than one motion blur images, e.g. noisy and blurry image pairs, for the estimation of the projective motion.

Another potential research direction is on about moving object/depth layer separation to extend our algorithm to deal with the limitation discussed above. In order to apply our algorithm to a moving object, we not only need to segment the moving object out from background, but we have to segment the moving object in a piecewise manner such that each piece of segment itself satisfied our Projective Motion Blur Model. For this kind of segmentation, we cannot use traditional "hard" segmentation algorithm, since for the boundaries of motion blurred object is a fractional boundary much like that used in image matting. However, unlike conventional image matting, we can impose additional constraints regarding object's motion.

REFERENCES

[1] A. Agrawal and R. Raskar. Resolving objects at higher resolution from a single motion-blurred image. In *CVPR*, 2007.

[2] J. Bardsley, S. Jefferies, J. Nagy, and R. Plemmons. Blind iterative restoration of images with spatially-varying blur. In *Optics Express*, pages 1767–1782, 2006.

[3] M. Ben-Ezra and S. Nayar. Motion Deblurring using Hybrid Imaging. In *CVPR*, volume I, pages 657–664, Jun 2003.

[4] M. Ben-Ezra and S. Nayar. Motion-based Motion Deblurring. *IEEE Trans. PAMI*, 26(6):689–698, Jun 2004.

[5] D. A. Bini, N. J. Higham, and B. Meini. Algorithms for the matrix p'th root. *Numerical Algorithms*, 39(4):349–378, 2005.

[6] J. Chen and C. K. Tang. Robust dual motion deblurring. In *CVPR*, 2008.

[7] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. In *ICCV*, 2007.

[8] S. Dai and Y. Wu. Motion from blur. In *CVPR*, 2008.

[9] N. Dey, L. Blanc-Fraud, C. Zimmer, Z. Kam, P. Roux, J. Olivo-Marin, and J. Zerubia. A deconvolution method for confocal microscopy with total variation regularization. In *IEEE International Symposium on Biomedical Imaging: Nano to Macro*, 2004.

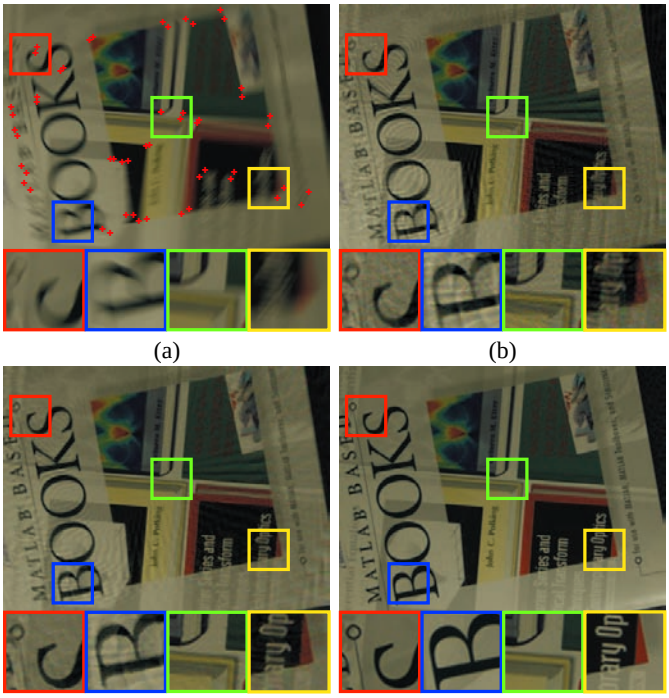


Fig. 12. (a)Input blur image, user's markups of correspondents for motion estimation are also shown, (b) Our result from the basic algorithm; (c) Our result with regularization; (d) Ground truth.

[10] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Trans. Graph.*, 25(3), 2006.
[11] G. Hong, A. Rahmati, Y. Wang, and L. Zhong. Sensecoding: Accelerometer-assisted motion estimation for efficient video encoding. In *ACM Mutlimedia'08*, 2008.
[12] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP*, 53(3):231–239, 1991.
[13] J. Jia. Single image motion deblurring using transparency. In *CVPR*, 2007.
[14] N. Joshi, R. Szeliski, and D. Kriegman. Psf estimation using sharp edge prediction. In *CVPR*, 2008.
[15] A. Levin. Blind motion deblurring using image statistics. In *NIPS*, pages 841–848, 2006.
[16] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Trans. Gr.*, 2007.
[17] A. Levin, P. Sand, T. S. Cho, F. Durand, and W. T. Freeman. Motion-invariant photography. *ACM Trans. Gr.*, 2008.
[18] A. Levin, Y. Weiss, F. Durand, and W. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, 2009.
[19] F. Li, J. Yu, and J. Chai. A hybrid camera for motion deblurring and depth map super-resolution. In *CVPR*, 2008.
[20] L. Lucy. An iterative technique for the rectification of observed distributions. *Astron. J.*, 79, 1974.
[21] V. Nostrand. *The International Dictionary of Mathematics*. Princeton, 1960. p70,"Baye's Theorem".
[22] R. Raskar, A. Agrawal, and J. Tumblin. Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans. Graph.*, 25(3), 2006.
[23] A. Rav-Acha and S. Peleg. Two motion blurred images are better than one. *PRL*, 26:311–317, 2005.
[24] W. Richardson. Bayesian-based iterative method of image restoration. *J. Opt. Soc. Am.*, 62(1), 1972.
[25] A. A. Sawchuk. Space-variant image restoration by coordinate transformations. *Journal of the Optical Society of America*, 64(2):138–144, 1974.
[26] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Trans. Gr.*, 2008.
[27] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *ICCV*, 2007.
[28] Y. Tai, H. Du, M. Brown, and S. Lin. Image/video deblurring using a hybrid camera. In *CVPR*, 2008.
[29] Y. Tai, H. Du, M. Brown, and S. Lin. Correction of spatially varying image and video blur using a hybrid camera. *IEEE Trans. PAMI*, 2009.
[30] Wiener and Norbert. Extrapolation, interpolation, and smoothing of stationary time series. *New York: Wiley*, 1949.
[31] L. Yuan, J. Sun, L. Quan, and H. Shum. Image deblurring with blurred/noisy image pairs. In *ACM Trans. Graph.*, page 1, 2007.
[32] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Progressive inter-scale and intra-scale non-blind image deconvolution. *ACM Trans. Gr.*, 2008.

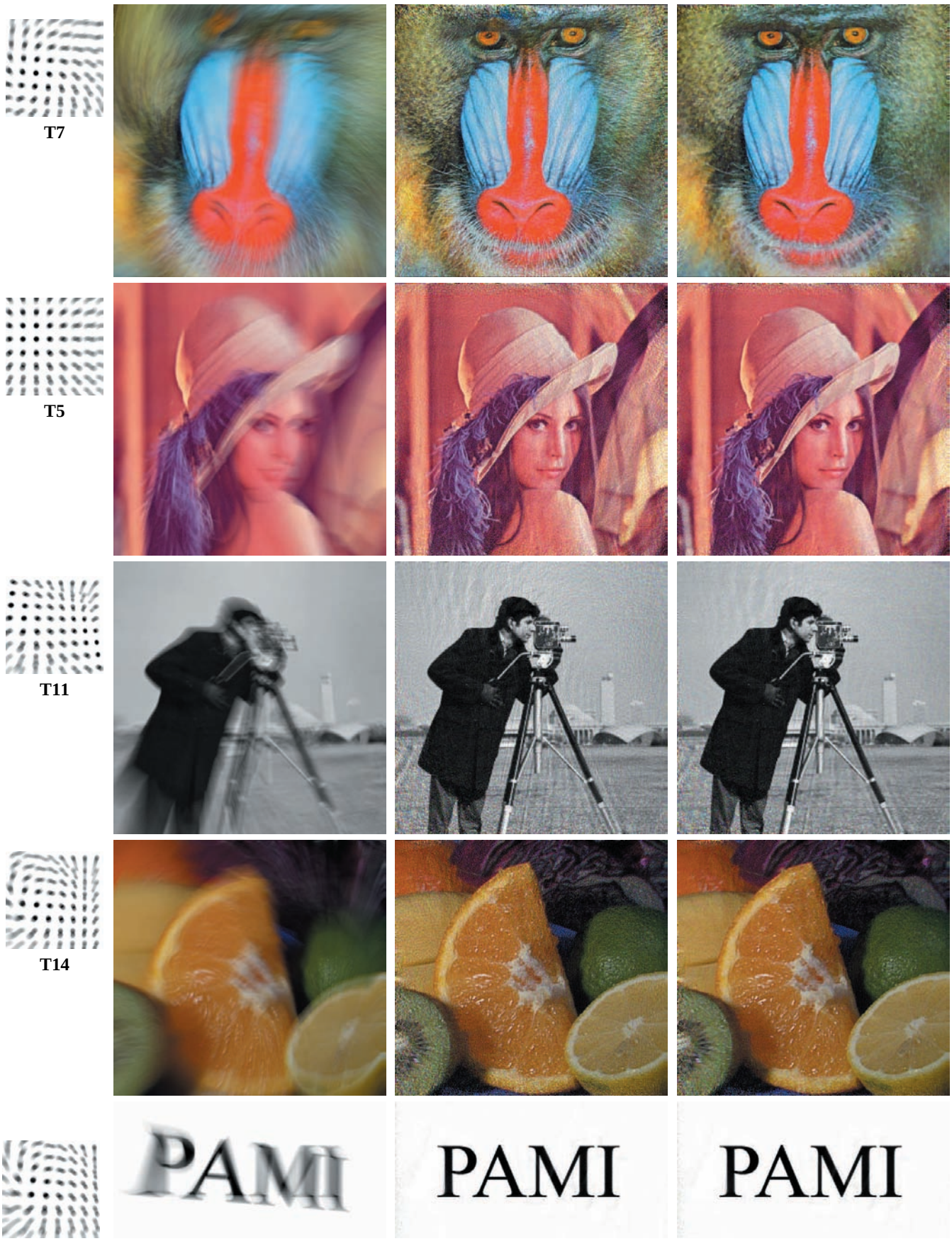


Fig. 13. In this figure, we show several examples of our test cases in Figure 9. The whole set of test cases and results are submitted in supplemental materials.