Image Registration and Change Detection under Rolling Shutter Motion Blur

Vijay Rengarajan, Ambasamudram Narayanan Rajagopalan, Rangarajan Aravind, and Guna Seetharaman

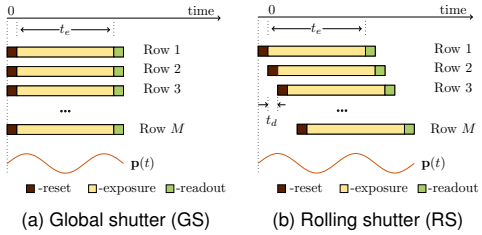
Abstract—In this paper, we address the problem of registering a distorted image and a reference image of the same scene by estimating the camera motion that had caused the distortion. We simultaneously detect the regions of changes between the two images. We attend to the coalesced effect of rolling shutter and motion blur that occurs frequently in moving CMOS cameras. We first model a general image formation framework for a 3D scene following a layered approach in the presence of rolling shutter and motion blur. We then develop an algorithm which performs layered registration to detect changes. This algorithm includes an optimisation problem that leverages the sparsity of the camera trajectory in the pose space and the sparsity of changes in the spatial domain. We create a synthetic dataset for change detection in the presence of motion blur and rolling shutter effect covering different types of camera motion for both planar and 3D scenes. We compare our method with existing registration methods and also show several real examples captured with CMOS cameras.

Index Terms—Rolling shutter, motion blur, change detection, image registration, aerial imaging

1 Introduction

Image registration [1] is the process of spatially aligning two images of the same scene and is critical for detecting changes in many application areas including aerial and satellite imagery. Earlier works have handled simple global translational and rotational differences [2], [3]. The process could be feature-based or intensity-based. The former class of techniques (such as [4]) first detects interesting points called feature points in both the images and use them to geometrically align the images by an affine or a projective transformation. Intensity methods such as [5] commonly employ correlation-based matching. Some of the problems encountered in image registration include the presence of motion blur due to camera motion, illumination changes due to the temporal change between captures, and the presence of 3D objects in the scene which poses issues with using a global homography.

In this paper, we address the problem of two-image registration and change detection in 3D scenes in the presence of camera motion for both global and rolling shutter cameras. We consider two important challenges specific to relative motion between the scene and the camera, namely motion blur and rolling shutter effect. These effects introduce additional level of complexity to the problem on hand. In a typical camera using CCD sensors, all pixels are exposed at the same time; these cameras are called global shutter (GS) cameras. They produce motion blur (MB) in the captured image when there is camera motion during exposure. Contemporary CMOS sensors employ an electronic rolling shutter (RS) in which the horizontal rows of the sensor array are scanned at different times. This behaviour results in additional distortions when capturing dynamic scenes and when imaging from moving cameras. Fig. 1 shows the mechanism by which sensors are exposed in RS and GS cameras. A GS camera exposes all the pixels at the same time for a time period t_e. Fig. 1(a) illustrates this operation by showing same start and end exposure times for each row of the sensor array. The M rows of an RS camera sensor array, on the other hand, are not exposed simultaneously. Instead, the exposure of consecutive rows starts sequentially with a delay t_d as shown in Fig. 1(b).



(a) Global shutter (GS) (b) Rolling shutter (RS)

Fig. 1. Exposure mechanism of GS and RS cameras.

The interplay between the rolling shutter effect and motion blur in an RS camera depends on the row exposure time t_e and the delay between the start of exposure of first and last rows $(M-1)t_d$. Fig. 2 shows three different scenarios corresponding to $t_e \ll (M-1)t_d$, $t_e \approx (M-1)t_d$ and $t_e \gg (M-1)t_d$ for a sensor with $t_d = 39.96\mu s$ in a 30fps camera having 834 scanlines for $M = 800$ rows (corresponding to 1125 scanlines for 1080 rows [6]). Though the time delay between rows in all the three scenarios is same, the amount of blur that each row experiences and the amount of intersection among the blur of different rows varies with respect to the row exposure time. For ease of illustration and understanding, we consider only x and y translational motion, since the motion about optical z-axis affects different image areas differently. In case (a), the number of camera poses that each row experiences is very low due to low exposure time, and hence the image is

- *V. Rengarajan, A.N. Rajagopalan, and R. Aravind are with the Department of Electrical Engineering, Indian Institute of Technology Madras, Chennai - 600036, India. E-mail: {vijay.ap,raju,aravind}@ee.iitm.ac.in*
- *G. Seetharaman is with the Information Directorate, Air Force Research Laboratory, Rome, NY 13441 USA. E-mail: guna@ieee.org*

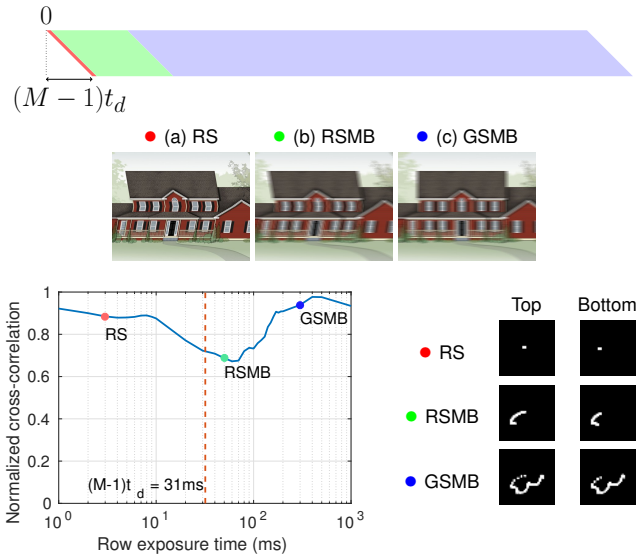


Fig. 2. Distortions in RS cameras. Top: Three types of distortions based on the amount of row exposure. Bottom left: Plot of normalized cross-correlation between kernels from top and bottom image regions for different exposure times. Bottom right: Sample top and bottom blur kernels for different distortions.

devoid of motion blur even though the RS effect is present due to the inter-row delay. In case (b), each row experiences different type of motion with sufficient exposure period to exhibit blur producing what we call as an RSMB image. In case (c), the row-exposure time dominates the total row delay, and every row experiences almost the same camera motion barring a few camera poses, and thus the resulting image tends to be a GSMB image as if the image is captured using a GS camera. Fig. 2 shows the variation in normalized cross-correlation between the local blur kernels located in top and bottom regions of the image. The plot can be divided into three regions corresponding to RS, RSMB, and GSMB. In the RS and GSMB regions, the two blur kernels are correlated very well, since in the RS case they are mere impulses (though shifted), and in the GSMB case, the camera motion experienced by the top and bottom regions over the long exposure is almost the same. The middle region of the plot wherein the local blur kernels are different (even for 2D translational motion) corresponds to the RSMB case.

In an earlier work [7], we tackled the problem of change detection in planar scenes in the presence of RS and MB. In this work, we develop a general model to address both RS and MB artifacts in 3D scenes that is equally applicable to both GS and RS cameras. Instead of the customary rectify-difference pipeline, we follow a distort-difference pipeline, in which we first distort the reference image to register it with the observed image followed by change detection. We follow a layered scene model [8], [9] wherein the image of a 3D scene is deemed to have been formed by stacking a number of fronto-parallel layers at different depths. We assume that the reference image is captured under conducive conditions and is free from RS and MB artifacts. For example, aerial images captured from fast-moving aircrafts will have distortions, while the reference is captured beforehand without any distortions from a satellite or a steady flying vehicle. Even though there are other factors such as changes in camera colour characteristics and sunlight conditions that could affect change detection methods, considering all possible artifacts is beyond the scope of our work. We have primarily concentrated on the distortions specific to RS camera motion as these are equally important and prevalent. Our method could be plugged into a complete end-to-end framework that attempts to tackle all issues. We would like to mention here that though we had pointed out one important application (aerial imagery), the theory developed in our work is general, and can be extended to other applications as well including mosaicing, super-resoution, and tracking, wherein the assumption of a global warp would fail in the case of RS motion and row-wise warping becomes imperative.

1.1 Related Works

In this section, we discuss works related to motion blur, rolling shutter, and 3D scene modelling.

Motion Blur: The study of motion blur is a classical problem in image processing and a large corpus of works exists that deals with the removal of motion blur. The motion blur is modelled as a set of projective homographies in a number of works [10], [11], [12], [13] and different optimization techniques are employed for deblurring such as gradient-based priors [10], [13] and modified Richardson-Lucy algorithm [11]. Information from inertial sensors are also used to estimate the camera poses such as in [14]. The work of [15] estimates homographies in the MB model posed as a set of image registration problems. Multiple images are employed in [16] to perform joint alignment, deblurring, and resolution enhancement, while a high sparsity-pursuit regularisation is employed in [17] to preserve salient edges that aid in kernel estimation during single image deblurring. A joint method to estimate scene depth and to remove MB through a unified layer-based model is proposed in [18] using a single blurred image. In our earlier work [19], we proposed a method to estimate camera motion in the presence of motion blur. In particular, we dealt with very large images where we leveraged the fact that the camera motion is same in all subimages inside the very large image. In this work, we do not deal with very large images and restrict ourselves to images captured using mobile phone CMOS cameras.

Rolling Shutter Effect: In [20], which is one of the first works that deals with RS rectification, a global translatory motion is estimated between two successive frames of a video, and the motion of every row is interpolated using a Bézier curve, which is then used to rectify the RS effect. A flow-based motion vector approach is taken to estimate the translatory motion. The RS wobble is removed from a video in [21] by posing it as a temporal super-resolution problem. Intra-frame motion is modeled as a super-resolved version of inter-frame global motion. A rotation-only camera motion for 3D scenes is modeled in [22] for RS video rectification. Key-rows are defined in every frame for which the camera motion is estimated using nonlinear least squares based on feature correspondences. The motion for remaining rows are interpolated using spherical linear interpolation. The method is also applicable for translatory camera motion for planar scenes. An algorithm based on homography mixtures

is proposed in [23] in which direct linear transform (DLT) is used to remove RS effect from videos. After locating feature matches between two frames, a modified DLT is used to estimate the camera motion in which the homography of a row is defined as a weighted average of that of pre-selected key-rows. A unified approach to estimate both RS and MB is formulated in [24], but a uniform velocity of the camera is assumed during the exposure of a video frame. In [25], information from multiple sensors of an RS camera is fused to estimate camera motion by fitting a continuous B-spline model. This framework is applied to simultaneous localisation and mapping. The work of [26] performs joint removal of MB and RS effect. The blur kernels are estimated across rows using which a polynomial-based camera trajectory is fitted, and a latent image is estimated using the trajectory. The downside of the work are the assumption of known inter-row delay time t_d of the RS camera and the consideration of only 2D out-of-plane rotational or 2D translational camera motion. Accurate estimation of t_d requires the use of imaging a flashing LED as employed in [22].

3D Scene Modelling: Modelling the 3D scene as an amalgamation of many flat layers situated at different depths is common in literature [8], and is used in areas including computational photography [9] and motion blur modelling [27]. There also exist works that locally adapt homographies for 3D scene registration. A piecewise image registration method is followed in [28] to handle 3D scenes, where the geometry of the scene is approximated by a piecewise continuous surface. In [29], region correspondences are determined to register 3D objects in the scene. To tackle the 3D nature of the scene, the image is divided into a number of subimages in [30] using Voronoi subdivision, and the subimages are registered using feature correspondences. A hybrid model of homography and content preserving warps to handle parallax and local distortions is followed in [31] for the application of mosaicing. As-projective-as-possible warping is employed in [32] to register and mosaic two images of 3D scenes. In [33], a global homography is used to coarsely align the two images, and local homographies are used to register image blocks.

1.2 Contributions

The main contribution of our work is the development of a combined image formation model to take care of the coupled nature of RS and MB effects. A unified framework to jointly perform image registration and change detection using a sparsity-based optimisation problem is presented. A layered registration approach is followed to consider the 3D nature of the scene.

Previous works on RS rectification [22], [23] have considered only the RS effect ignoring the presence of MB, and also estimated only a coarse row-wise camera motion through interpolation. We estimate dense camera motion through all rows of an image considering both RS and MB. Works that have dealt with both RS and MB, have their own restrictions, such as the assumption of uniform camera velocity [24] and the assumption of 2D parametric camera motion [26]. In this work, we deal with both RS and MB under a single roof without making any assumptions on camera velocity and on camera motion path parametrisation.

We extend our own work [7] in the following ways. In [7], we framed our formulation of change detection for only planar scenes dealing with RS and MB. In this work, we extend and generalise the image formation model and the change detection algorithm for 3D scenes. The experimental section has been expanded to include quantitative comparisons with the existing methods using a synthetic dataset created specifically for change detection in the presence of RS and MB, and change detection in both planar and 3D scenes with comparisons. In the supplementary material, we have included synthetic experiments on camera motion estimation for uniform and nonuniform trajectories.

2 Image Formation Model

In the following discussion, we first briefly describe the combined rolling shutter and motion blur (RSMB) model for planar scenes from our work [7], and then we generalise it for 3D scenes by following a layered approach. We then proceed to develop an algorithm for change detection, where we first register the background layer by estimating its camera motion and then register other layers while simultaneously detecting changes, if any.

2.1 RSMB Imaging in Planar Scenes

In a rolling shutter camera that is imaging M rows, the ith row of the sensor plane is exposed during the time interval $[(i-1)t_d, (i-1)t_d + t_e]$ for $1 \leq i \leq M$. Here t_e is the exposure time of a row and t_d is the inter-row delay time with $t_d < t_e$ (Refer Fig. 1). The total exposure time of the image is given by $T_e = (M-1)t_d + t_e$. For the camera path $\mathbf{p}(t)$ representing 6D camera motion, the ith row is blinded to the whole time except for $(i-1)t_d \leq t \leq (i-1)t_d + t_e$.

Let $\mathbf{f}$ be the image captured by the RS camera with no camera motion, and let $\mathbf{g}$ be the distorted RSMB image captured when the camera trajectory is $\mathbf{p}(t)$. Then, each row of $\mathbf{g}$ is an averaged version of the corresponding rows in warped versions of $\mathbf{f}$ due to the camera motion in its exposure period. We have

$$\mathbf{g}^{(i)} = \frac{1}{t_e} \int_{(i-1)t_d}^{(i-1)t_d + t_e} \mathbf{f}^{(i)}_{\mathbf{p}(t)} \, dt, \text{ for } i = 1 \text{ to } M, \qquad (1)$$

where $\mathbf{f}^{(i)}_{\mathbf{p}(t)}$ is the ith row of the warped version of $\mathbf{f}$ due to the camera pose $\mathbf{p}(t)$ at a particular time t.

The discretised model with respect to a finite camera pose space $\mathcal{S}$ is given by

$$\mathbf{g}^{(i)} = \sum_{\boldsymbol{\tau}_k \in \mathcal{S}} \omega^{(i)}_{\boldsymbol{\tau}_k} \, \mathbf{f}^{(i)}_{\boldsymbol{\tau}_k}, \qquad (2)$$

where $\mathcal{S} = \{\boldsymbol{\tau}_k\}_{k=1}^{|\mathcal{S}|}$, $|\mathcal{S}|$ denotes the cardinality of $\mathcal{S}$, $\mathbf{f}^{(i)}_{\boldsymbol{\tau}_k}$ is the ith row of the warped reference image $\mathbf{f}_{\boldsymbol{\tau}_k}$ due to the camera pose $\boldsymbol{\tau}_k$. Each element of the pose weight vector $\omega^{(i)}_{\boldsymbol{\tau}_k}$ represents the fraction of time that the camera stayed in the pose $\boldsymbol{\tau}_k$ during the ith row exposure. When the exposure times of $\mathbf{f}^{(i)}$ and $\mathbf{g}^{(i)}$ are same, then by conservation of energy, we have $\sum_{\boldsymbol{\tau}_k \in \mathcal{S}} \omega^{(i)}_{\boldsymbol{\tau}_k} = 1$ for each i.

2.2 RSMB Imaging in 3D Scenes

In this subsection, we explain the generalization of the image formation model in (2) to 3D scenes by following the layered approach.

Let us consider L depth layers in the scene, and let the set $\mathcal{L} = \{\ell\}_{\ell=1}^{L}$ index the layers. The relative depth of each layer is given by $\{d_\ell\}_{\ell=1}^{L}$, where d_1 is the layer closest to the camera and d_L is the layer farthest from the camera (i.e. the background layer). These values are normalised with respect to the background layer so that $d_L = 1$, and $d_i > d_j$ for $i > j$. During the exposure of the scene, the layer ℓ could possibly be masked by the layers $\{\ell'\}_{\ell'=1}^{\ell-1}$ at the image plane of the camera. The mask at each layer depends on the homography due to the camera pose at that layer since the motion depends on depth.

Let $\boldsymbol{\alpha}_{(\boldsymbol{\tau}_k,\ell)}$ be the object mask of a layer ℓ at camera pose $\boldsymbol{\tau}_k$. This variable indicates where the objects are present at a particular layer. $\boldsymbol{\alpha}_{(\boldsymbol{\tau}_k,\ell)}$ will be 1 for the pixels where objects are present (i.e. where the layer could possibly contribute to the final image) and 0 otherwise. Let $\boldsymbol{\beta}_{(\boldsymbol{\tau}_k,\ell)}$ denote the final layer mask that indicates the actual contribution of a layer to the final image. We have

$$\boldsymbol{\beta}_{(\boldsymbol{\tau}_k,\ell)} = \boldsymbol{\alpha}_{(\boldsymbol{\tau}_k,\ell)} \prod_{j=1}^{\ell-1} \overline{\boldsymbol{\alpha}}_{(\boldsymbol{\tau}_k,j)}, \tag{3}$$

where $\overline{\boldsymbol{\alpha}}_{(\boldsymbol{\tau}_k,j)}$ is the complement of the object mask indicating the occlusion of a layer from being seen at the image plane due to the layers in front of it. The above discussion is valid for each row of the distorted image. Hence the distorted image resulting from all the camera poses is given by

$$\mathbf{g}^{(i)} = \sum_{\boldsymbol{\tau}_k \in \mathcal{S}} \omega_{\boldsymbol{\tau}_k}^{(i)} \sum_{\ell=1}^{L} \boldsymbol{\beta}_{(\boldsymbol{\tau}_k,\ell)} \widehat{\mathbf{f}}_{(\boldsymbol{\tau}_k,\ell)}^{(i)}, \text{ for } i = 1, \ldots, M. \tag{4}$$

Here $\widehat{\mathbf{f}}$ represents the complete scene information at each layer, i.e. $\widehat{\mathbf{f}}_{(\boldsymbol{\tau}_k,\ell)}$ is the image seen by the camera if it was the only layer present in the scene. Note here that although the camera poses are same for all layers in a particular row, the actual warp experienced by each layer is different based on its depth.

In (4), let $\mathbf{f}_{(\boldsymbol{\tau}_k,\ell)}^{(i)} = \boldsymbol{\beta}_{(\boldsymbol{\tau}_k,\ell)} \widehat{\mathbf{f}}_{(\boldsymbol{\tau}_k,\ell)}^{(i)}$ represent the disjoint layer image for a particular pose $\boldsymbol{\tau}_k$ at layer ℓ. Thus, the distorted image can be written as

$$\mathbf{g}^{(i)} = \sum_{\boldsymbol{\tau}_k \in \mathcal{S}} \left(\omega_{\boldsymbol{\tau}_k}^{(i)} \sum_{\ell=1}^{L} \mathbf{f}_{(\boldsymbol{\tau}_k,\ell)}^{(i)} \right). \tag{5}$$

The distorted image is thus represented as a weighted sum of warped images from all the layers. It can be equivalently expressed as an image formed by collecting images layer by layer and fusing them as given in (6) below.

$$\mathbf{g}^{(i)} = \sum_{\ell=1}^{L} \left(\sum_{\boldsymbol{\tau}_k \in \mathcal{S}} \omega_{\boldsymbol{\tau}_k}^{(i)} \mathbf{f}_{(\boldsymbol{\tau}_k,\ell)}^{(i)} \right) \tag{6}$$

This is a neat extension of our planar RSMB framework in [7] to 3D scenes. During depth inference (layer assignment), some pixels (at edges) could embed information from more than one depth layer. However, this issue is not significant so long as the blur is not severe in RSMB. The planar scene model in (2) is a special case of (6) when $L = 1$, the only layer being the background layer. Our model neatly encompasses both GS and RS camera acquisition mechanisms with and without MB. If $\|\boldsymbol{\omega}^{(i)}\|_0 = 1$, then there is only one active pose for each row, and thus the model in (6) corresponds to the RS-only framework. If $\boldsymbol{\omega}^{(i)}$ is same for all i, then the model represents global shutter cameras. With $\boldsymbol{\omega}^{(i)} = \boldsymbol{\omega}^{(j)}$ for all i, j, $\|\boldsymbol{\omega}^{(i)}\|_0 = 1$ represents the GS-only case, while $\|\boldsymbol{\omega}^{(i)}\|_0 > 1$ represents the GSMB model.

2.3 Change Modelling

The relationship (5) between the reference image and the distorted image due to camera motion accounts only for the camera motion and does not account for actual changes in the scene. We model any new objects or changes in the scene as an additive component. Thus, the distorted image is given by

$$\mathbf{g}^{(i)} = \sum_{\boldsymbol{\tau}_k \in \mathcal{S}} \omega_{\boldsymbol{\tau}_k}^{(i)} \sum_{\ell=1}^{L} \mathbf{f}_{(\boldsymbol{\tau}_k,\ell)}^{(i)} + \boldsymbol{\chi}^{(i)}, \text{ for } i = 1, \ldots, M, \tag{7}$$

where $\boldsymbol{\chi}^{(i)}$ is the change vector which contains nonzero values in pixels where there are changes in the distorted image with respect to the reference image. Collecting all rows as an image, $\{\boldsymbol{\chi}^{(i)}\}_{i=1}^{M}$ represents all the changes. Segmenting $\{\boldsymbol{\chi}^{(i)}\}_{i=1}^{M}$ itself into different depth layers is not handled here, since it is more aligned with the application of single image depth estimation. In our case, $\boldsymbol{\chi}^{(i)}$ incorporates changes in all the layers.

The linear combination of different warps of the image in (7) can be expressed in matrix-vector multiplication form as

$$\mathbf{g}^{(i)} = \sum_{\ell=1}^{L} \mathbf{F}_{\ell}^{(i)} \, \boldsymbol{\omega}^{(i)} + \boldsymbol{\chi}^{(i)} \tag{8}$$

where the columns of $\mathbf{F}_{\ell}^{(i)}$ contain the rows of warped versions of the reference image at layer ℓ and each column gets a weight value from $\boldsymbol{\omega}^{(i)}$. Upon rearranging,

$$\mathbf{g}^{(i)} = \begin{bmatrix} \sum_{\ell=1}^{L} \mathbf{F}_{\ell}^{(i)} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega}^{(i)} \\ \boldsymbol{\chi}^{(i)} \end{bmatrix} := \mathbf{B}^{(i)} \boldsymbol{\xi}^{(i)} \tag{9}$$

The matrix $\mathbf{B}^{(i)}$ has two parts, the first one being the collection of all warps of the reference image within the pose space $\mathcal{S}$, and the second part being the identity matrix to represent the location of the changes. The first part of $\boldsymbol{\xi}^{(i)}$ weights these warps and the second part of $\boldsymbol{\xi}^{(i)}$ weights the identity matrix (i.e. the values for the changed pixels). To stack up the columns with warps, the depth map of the scene must be known so that a homography can found out for each depth layer in the scene. If the depth map of the scene is known, we can detect changes in the distorted image by solving for the camera motion and the change vector weights in (9). As this problem is under-determined, we minimise the following energy to solve for the camera motion and the changes jointly:

$$\begin{aligned} \min\{E(\boldsymbol{\xi}^{(i)}) = \|\mathbf{g}^{(i)} - \mathbf{B}^{(i)}\boldsymbol{\xi}^{(i)}\|_2^2 + \lambda_1 \|\boldsymbol{\omega}^{(i)}\|_1 \\ + \lambda_2 \|\boldsymbol{\chi}^{(i)}\|_1\} \text{ with } \boldsymbol{\omega}^{(i)} \succeq \mathbf{0}, \end{aligned} \tag{10}$$

where λ_1 and λ_2 are non-negative regularisation parameters and $\succeq$ denotes non-negativity of each element of the vector. The first term in (10) imposes the photometric constancy constraint accounting for camera motion and the change, while the ℓ_1 norm is used as the prior on the camera motion and the region of changes to encourage sparsity. We observe that (i) the camera can move only by a small amount in the whole space of 6D camera poses, and (ii) the number of changed pixels is sparse in every row in the spatial domain. We note that (ii) indirectly relates to the amount of environmental changes between the two time of captures. To enforce different sparsity levels on the camera motion and the changes, we use two ℓ_1 regularisation parameters with different values. We also note that the camera pose weights represent the fraction of time for camera poses, and hence we also impose a non-negativity constraint on $\boldsymbol{\omega}^{(i)}$.

If the depth map of the scene is not known, which is the case in most scenarios, it is not possible to warp different layers according to their depths and stack up the columns of $\mathbf{B}^{(i)}$. Hence, we follow a layered registration approach in which we start by registering the background layer of the distorted image with the reference image by estimating the camera motion and changes, and then registering the changed regions one-by-one to other layers, thereby detecting the actual changes as regions which are not registered to any of the layers. The procedure is enumerated as follows:

1) Registration of background layer : Section 2.3.1
2) Segmentation of changed regions : Section 2.3.2
3) Detection of final changes : Section 2.3.3

2.3.1 Registration of Background Layer

To register the background layer, we rewrite the formulation in (9) with respect to only the background layer (i.e. $\ell = L$). The information in all the remaining layers are accounted as change.

$$\mathbf{g}^{(i)} = \begin{bmatrix} \mathbf{F}_L^{(i)} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega}_L^{(i)} \\ \boldsymbol{\chi}_L^{(i)} \end{bmatrix} := \mathbf{B}_L^{(i)} \boldsymbol{\xi}_L^{(i)} \tag{11}$$

The vector $\boldsymbol{\omega}_L^{(i)}$ provides weights to the warps of the background layer due to the camera motion. The change vector $\boldsymbol{\chi}_L^{(i)}$ contains both the actual changes between the two images and the changes due to misregistration of the image in other depth layers $\{\ell\}_{\ell \neq L}$, since the applied warp and the actual warp in that layer are different. Thus, the changes with respect to the background layer $\boldsymbol{\chi}_L^{(i)}$ subsumes the actual changes at all layers, $\boldsymbol{\chi}^{(i)}$, and the nonzero pixels located in other layers, $\sum_{\ell=1}^{L-1} \mathbf{F}_\ell^{(i)} \boldsymbol{\omega}^{(i)}$.

$$\boldsymbol{\chi}_L^{(i)} = \boldsymbol{\chi}^{(i)} + \sum_{\ell=1}^{L-1} \mathbf{F}_\ell^{(i)} \boldsymbol{\omega}^{(i)} \tag{12}$$

We formulate and solve the following optimisation problem to obtain the camera motion and the change vector for the background layer.

$$(\boldsymbol{\omega}_L^{(i)*}, \boldsymbol{\chi}_L^{(i)*}) = \arg \min_{\boldsymbol{\omega}_L, \boldsymbol{\chi}_L} \|\mathbf{g}^{(i)} - \mathbf{B}_L^{(i)} \boldsymbol{\xi}_L^{(i)}\|_2^2 + \lambda_1 \|\boldsymbol{\omega}_L^{(i)}\|_1 + \lambda_2 \|\boldsymbol{\chi}_L^{(i)}\|_1 \text{ subject to } \boldsymbol{\omega}_L^{(i)} \geq 0 \tag{13}$$

The estimated vector $\boldsymbol{\omega}_L^{(i)*}$ provides weights for the camera poses in $\mathcal{S}$. The nonzero pixels in $\boldsymbol{\chi}_L^{(i)*}$ correspond to the changes in the image with respect to the background layer. We modify the *nnLeastR* function provided in the SLEP package (Liu et al. [34]) to account for the partial non-negativity of $\boldsymbol{\xi}_L^{(i)}$ and solve (13). Instead of using the same search space $\mathcal{S}$ for all rows, we adapt the space for every row as will be discussed in Section 2.4.

2.3.2 Segmentation of Changed Regions

Solving (13) gives an *image* $\{\boldsymbol{\chi}_L^{(i)*}\}_{i=1}^M$ that contains both the non-background layers and the actual changes in the distorted image, but only the regions which are not present in the reference image should be detected as final changes. To mark the regions of changes from $\{\boldsymbol{\chi}_L^{(i)*}\}_{i=1}^M$, we follow the thresholding algorithm in [35], which calculates a threshold based on the entropy of the histogram. This algorithm has been shown to perform better than other thresholding methods [36]. After applying the threshold, to deal with noise-like objects, we perform connected component analysis and eliminate components which are smaller than a fixed size.

The segmented regions are not smooth and continuous, in general. Due to the homogeneous regions present inside the objects, the changes detected may contain holes in them. Before trying to register each object, we need to extract each object separately. We determine the distance transform image of the resulting object layer image. Distance transform assigns a value for each pixel based on its distance from the nearest black pixel. We then threshold the distance transform image, and create a binary image. The pixels which have distance transform values less than a preset threshold (six for 384x256 image) are assigned a value of one, and the rest are assigned zero. The resultant image has closed holes corresponding to unregistered objects. We then perform hole filling and eroding operations to arrive at a binary image containing all the objects without any holes in them. We extract these individual object regions using a simple connected component approach. We name these binary object regions $\{\widehat{\boldsymbol{\chi}}_p\}_{p=1}^P$ and extract regions containing the objects from the distorted image $\mathbf{g}$ as $\{\mathbf{g}_p\}_{p=1}^P = \{\mathbf{g} \cdot \widehat{\boldsymbol{\chi}}_p\}_{p=1}^P$, where P is the number of extracted object regions.

2.3.3 Detection of Final Changes

We now aim to register each of these objects $\{\mathbf{g}_p\}_{p=1}^P$ with the reference image. If the registration of an object is not successful, then the corresponding region is considered a change. While it is possible to register each extracted object region from the distorted image with the reference image by estimating the camera motion for its rows anew, it is wiser to adopt a simpler procedure which uses the following facts for a particular row i: (i) both the background layer and all other layers in that row are affected by the same camera motion $\mathbf{p}(t), (i-1)t_d \leq t \leq (i-1)t_d + t_e$, and (ii) the pose weight vector $\boldsymbol{\omega}_L^{(i)}$ for all layers remains unaltered, since the exposure period for all layers is same. Even if the camera motion is same, a single camera pose affects points at different layers differently. As a homography, it pertains only to a plane in the scene. But if we know the rotations and translations observed on the image plane for a particular layer due to camera motion, then we can arrive at the set of rotations and translations observed for another

layer. Rotations remain the same irrespective of the depth of the layer, while translations scale with respect to depth.

Consider a changed region $\mathbf{g}_p$ which spans a sequence of rows $m \leq i \leq n$. The corresponding search spaces are $\{\mathcal{S}^{(i)}\}_{i=m}^{n}$ and the estimated weight vectors for the background layer are $\{\boldsymbol{\omega}_L^{(i)*}\}_{i=m}^{n}$. Though the region of change is not registered at the background layer using the poses in the tuples $\{(\mathcal{S}^{(i)}, \boldsymbol{\omega}_L^{(i)*})\}$, a scaled space containing scaled translational poses $\mathcal{S}_\ell^{(i)}$ would register it correctly at layer ℓ, if it is present at depth d_ℓ in the reference image. For a pose $\boldsymbol{\tau}_k^{(i)}$ equivalently represented by image plane motion $(t_{x_L}, t_{y_L}, s_L, r_{x_L}, r_{y_L}, r_{z_L})^{(i)}$ at the background layer, we determine the pose vector $(t_{x_\ell}, t_{y_\ell}, s_\ell, r_{x_L}, r_{y_L}, r_{z_L})^{(i)}$ for the relative depth d_ℓ using the following relation [37]: $s_\ell = \frac{d_\ell}{\rho}, t_{x_\ell} = (\rho - d_\ell + 1)t_{x_L}/\rho, t_{y_\ell} = (\rho - d_\ell + 1)t_{y_L}/\rho$, where $\rho = d_\ell + (1/s_L - 1)$, and the translation along optical axis is represented as a scale factor s.

A scaled space $\mathcal{S}_\ell^{(i)}$ corresponding to $\mathcal{S}^{(i)}$ could thus be generated to register the object $\mathbf{g}_p$. Since we do not know the depth of the object, we consider a series of finely separated discrete scale values for d_ℓ for each row, and generate a new scaled pose space $\mathcal{S}_\ell^{(i)}$ using the estimated pose space of the background layer. We forward-blur the reference image using the tuples $\{(\mathcal{S}_\ell^{(i)}, \boldsymbol{\omega}_L^{(i)*})\}$, and determine the energy difference between $\mathbf{g}_p$ and forward-blurred image only in the pixels covering the object p. We decide that the object is present at this layer ℓ if these two match, i.e. if the RMSE is below a threshold. The object is marked as a change if no relative depth value d_ℓ is able to explain the distortion. Algorithm 1 summarises the steps involved in image registration and change detection in 3D scenes.

Algorithm 1 Steps for 3D change detection

1: Register the background layer of every row using the pose spaces $\{\mathcal{S}^{(i)}\}$ using (13) and estimate the pose weight vectors $\{\boldsymbol{\omega}_L^{(i)*}\}$ and the change vectors $\{\boldsymbol{\chi}_L^{(i)*}\}$.
2: Segment and extract objects $\{\mathbf{g}_p\}$ from the changes $\{\boldsymbol{\chi}_L^{(i)*}\}$ using the procedure given in Section 2.3.2.
3: Register each object $\mathbf{g}_p$ at a layer ℓ with relative depth d_ℓ using the scaled tuples $\{(\mathcal{S}_\ell^{(i)}, \boldsymbol{\omega}_L^{(i)*})\}$ following Section 2.3.3.
4: If the object $\mathbf{g}_p$ does not get registered at any layer ℓ, mark it as change.

Importance of Relative Depth: With the increasing use of imaging from low-flying drones, it has become necessary to consider the relative depth of the scene as against a planar assumption in traditional aerial imaging using high-altitude aircrafts. For instance, a conservative estimate of the average height of top ten tallest buildings in major Indian cities as on March 2016 [38] are 120m (Bengaluru), 115m (Chennai), 135m (Delhi), 122m (Hyderabad), and 227m (Mumbai). Hence, any image captured from an altitude of 1000m or less will experience depth-dependent motion artifacts in the image plane. Fig. 3 (left) shows the relative depth of a building at different heights for camera-to-ground distances of 600m and 1000m. For a camera with focal length of 2000 pixels, the difference in the length of the blur caused for the points on the ground plane and the building are not negligible as can be observed from Fig. 3 (right).

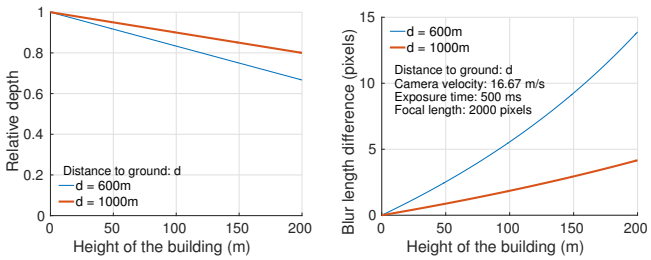


Fig. 3. Variation of relative depth and blur length versus building height for camera-to-ground distances of 600m and 1000m.

2.4 Pose Space Adaptation

We now discuss the following: (i) how the row-wise pose space adaptation helps in speeding up the motion estimation and imposing smoothness, (ii) what motion parameters can be estimated during registration of a single row, and (iii) how to tackle registration of homogeneous rows.

Dynamically Varying Pose Space : Since the camera motion changes continuously for every row, we use different pose search spaces $\mathcal{S}^{(i)}$ for each row i following [7], instead of a single common pose space $\mathcal{S}$ for all the rows. Through this adaptation of the pose space, we impose smoothness on the camera trajectory during background registration in (13). This also speeds up the optimisation problem, since the size of the search space for each row is small. Let $N(\boldsymbol{\tau}, \mathbf{b}, \mathbf{s}) = \{\boldsymbol{\tau} + q\mathbf{s} : \boldsymbol{\tau} - \mathbf{b} \preceq \boldsymbol{\tau} + q\mathbf{s} \preceq \boldsymbol{\tau} + \mathbf{b}, q \in \mathbb{Z}\}$ denote the neighbourhood of poses around a particular 6D pose vector $\boldsymbol{\tau}$, where $\mathbf{b}$ is the bound around the pose vector and $\mathbf{s}$ is the step-size vector. We estimate the pose weight vector $\boldsymbol{\omega}_L^{(i)}$ row-wise with $\mathcal{S}^{(i)} = N(\boldsymbol{\tau}_c^{(j)}, \mathbf{b}, \mathbf{s})$, where $j = i + 1$ for $i < M/2$ and $j = i - 1$ for $i > M/2$, M is the number of image rows, and $\boldsymbol{\tau}_c^{(j)}$ is the centroid pose of the jth row, which is given by

$$\boldsymbol{\tau}_c^{(j)} = \frac{\sum_{\boldsymbol{\tau}_k} \omega_{\boldsymbol{\tau}_k}^{(j)} \boldsymbol{\tau}_k}{\sum_{\boldsymbol{\tau}_k} \omega_{\boldsymbol{\tau}_k}^{(j)}}. \tag{14}$$

Thus, each row of the distorted image after background-registration is associated with a tuple $(\mathcal{S}^{(i)}, \boldsymbol{\omega}_L^{(i)*})$.

Degrees of Freedom: In this work, we follow a projective motion blur model for each row. The relation between the projections of a 3D point in a planar scene on the image plane when the camera is at the origin and at position $(\mathbf{R}, \mathbf{T})$, is given by a homography $\mathbf{H}$ as [12], $\mathbf{x}_{(\mathbf{R},\mathbf{T})} = \mathbf{H}\mathbf{x}$, where

$$\mathbf{H} = \mathbf{K}_\nu \left([\mathbf{R}] + \frac{1}{d} \mathbf{T}\mathbf{n}^T \right) \mathbf{K}_\nu^{-1}. \tag{15}$$

Here $\mathbf{R} = [R_x, R_y, R_z]^T$ and $\mathbf{T} = [T_x, T_y, T_z]^T$ are camera rotation and translation vectors, $[\mathbf{R}]$ is the matrix exponential equivalent of $\mathbf{R}$ [12], $\mathbf{K}_\nu$ is the known camera intrinsic matrix, ν is the focal length, $\mathbf{n} = [0, 0, 1]^T$ is the normal of the scene plane which is parallel to the image plane, and d is the distance between the camera centre and the scene. Thus, each $\mathcal{S}^{(i)}$ represents a 6D camera pose space.

Estimation of 6D camera motion by registering a single row of the distorted image to the reference image is not unambiguous. A vertical displacement of a horizontal line could be due to a rotation about horizontal axis (x) or a translation along vertical axis (y). Hence, the estimation of camera motion from a single row could lead to the above ambiguity. The homography due to the rotation "$R_x = \theta$" is given by H_1 in (16). This is equivalent to the homography H_2 corresponding to a vertical translation "$T_y = y_0(\cos\theta - 1) + \nu \sin\theta$" and "$T_z = -y_0 \frac{\sin\theta}{\nu} + \cos\theta$", for the points on the row y_0.

$$H_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \nu\sin\theta \\ 0 & -\frac{\sin\theta}{\nu} & \cos\theta \end{bmatrix} \text{ and } H_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & y_0(\cos\theta - 1) + \nu\sin\theta \\ 0 & 0 & -y_0\frac{\sin\theta}{\nu} + \cos\theta \end{bmatrix} \quad (16)$$

If the pose search space contains both these transformations, then our optimisation problem could pick either one, since both are correct. Hence, as an initialisation, we solve the 6D camera motion for a block of rows around the middle row, instead of a single row. From this estimate $\boldsymbol{\omega}_L^{(b)*}$, we calculate the centroid pose $\boldsymbol{\tau}_c^{(b)}$ using (14). Then, we proceed with the camera motion estimation of the middle row using $\mathcal{S}^{(M/2)} = N(\boldsymbol{\tau}_c^{(b)}, \mathbf{b}, \mathbf{s})$.

Homogeneous Rows: While estimating motion for a particular row using (13), the optimisation problem may not converge or may lead to a trivial solution if the row of the distorted image is uninformative. We therefore detect homogeneous rows and ignore them during row-wise estimation. For these rows, the centroid of the camera motion is interpolated from the estimated camera motion of neighbouring informative rows. For every row, we determine the horizontal gradient using the filter $[-1, 1]$. We then count the number of values (N_ϵ) greater than ϵ. If the ratio of this value and the number of columns, $N_\epsilon/N < n_0$ for a particular row, then we consider that row as homogeneous. In addition, the homogeneity check helps in automatically choosing the initial block of rows to kickstart the row-wise estimation. We choose the block of rows nearest to the centre which is devoid of homogeneity for this purpose.

3 Experiments

The experiments section is divided into three parts. In the first part, we demonstrate the working of our change detection method for 3D scenes using synthetic examples. In the next part, we provide a quantitative analysis of our algorithm and compare with existing methods using our synthetic dataset. Since there are no existing datasets of RSMB images for change detection, and also since existing change detection datasets do not contain RS artifacts, we created and employed our own dataset for validation with different camera trajectories exhibiting RS and MB distortions. Finally, we show results on real examples for both planar and 3D scenes.

Comparisons: We compare our joint RSMB framework with sequential frameworks of MB-RS deblur-register and RS-MB rectify-register, employing state-of-the-art methods to address motion blur and rolling shutter independently. We use the method of Xu et al. [17] for nonuniform deblurring, and the methods of Liang et al. [20], Ringaby and Forssén [22] and Grundmann et al. [23] for RS rectification. We also compare with the recent joint RSMB deblurring algorithm of [26], which performs deblurring on a single RSMB image assuming that the camera motion is only 2D out-of-plane rotations (or in-plane 2D translations). For real 3D scenes, we compare with locally adaptive image registration methods of Linger and Goshtasby [33], and Zaragoza et al. [32] that take care of local geometric variations.

For [20] and [22], we implemented a two-frame version of their multi-frame algorithm; both our method and their methods are given the knowledge that the first image is the reference image. For [23], we used the stabilization option from YouTube. We used our own implementation for [33]. We used the author provided codes for [32], [17], and [26].

Synthetic 2D Experiments Due to space constraints, we discuss the following synthetic experiments with respect to planar scenes in the supplementary material: (i) the accuracy of the camera motion estimation of our method during the registration of two images without any changes, where we also demonstrate the inaccurate motion estimation of [20] and [22], (ii) comparisons for change detection in planar scenes with the sequential frameworks [17]+[20] and [17]+[22], and the joint RSMB framework [26].

3.1 Synthetic Experiments

The effect of RS and MB is simulated in the following manner. We generate a series of camera pose values forming a trajectory. From this series, we assign a set of poses consecutively to each row with successive rows sharing a subset of poses. We determine the homography corresponding to these poses using (15) and generate the RSMB image using (5). To generate RS-only image, we use one pose per row.

(a) (b) (c)

Fig. 4. **Synthetic experiment**: Change detection in a 3D scene. (a) Reference image, (b) Depth map, and (c) RSMB image.

3D Scenes: We show the performance of our method for change detection in 3D scenes. Fig. 4(a) shows a clean image of a 3D scene and Fig. 4(b) shows its corresponding depth map. The darker the region, the farther it is from the camera. The background layer, having a relative depth value of 1, is shown in dark gray, since it is farthest from the camera. There are two objects at different depths; one (*bunny*) at a relative depth of 0.5 and another (*block*) at 0.4 with respect to the background. The 3D camera motion is applied synthetically on this 3D scene with objects according to the homography equation in (15) and the image formation model in (5). The object with lower relative depth experiences larger motion, since it is closer to the camera. We consider three cases of change detection by varying the reference image: (i) one-object change, (ii) two-objects change, and (iii) no change.

Case-(i): We first consider the reference image with only the *bunny* as in Fig. 5(i)-(a). We follow the steps in Algorithm 1 to detect the changes. To perform the registration of

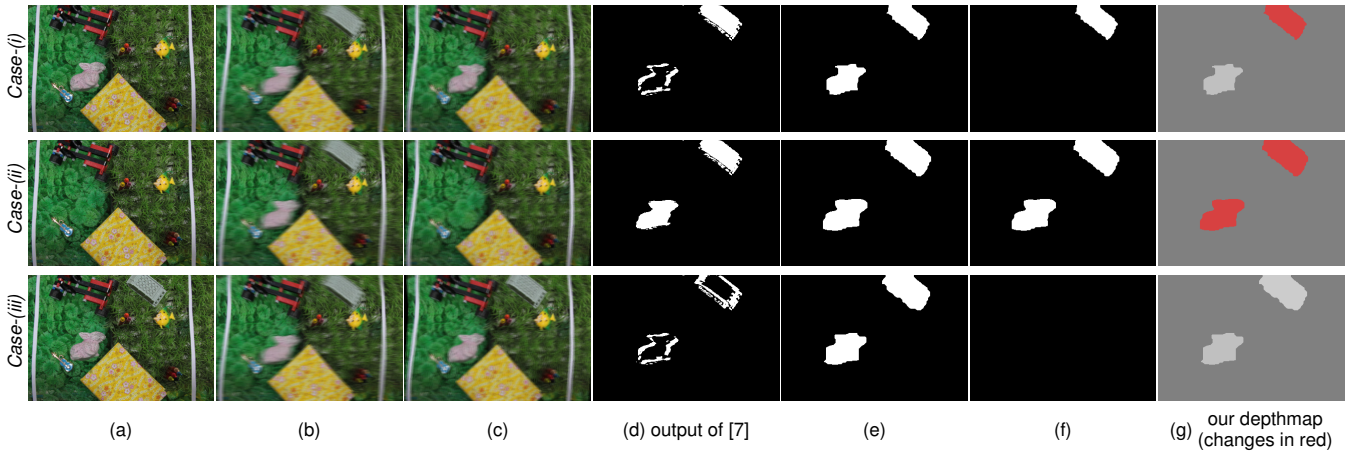


Fig. 5. **Synthetic experiment**: Change detection in a 3D scene. Row-1: *Case-(i)* One-object change, Row-2: *Case-(ii)* Two-object change, and Row-3: *Case-(iii)* No-object change. (a) Reference image, (b) RSMB image, (c) Background registered image, (d) Detected changes after background registration, (e) Extracted objects, (f) Detected changes, and (g) Estimated depth map (with detected changes in red).

background, we solve (13) to get pose weight vectors for each row. The background-registered image is shown in Fig. 5(i)-(c). The change vector weights are nonzero for the unregistered regions, which correspond to both the objects, but only one of which is the actual change. This is shown as a binary image in Fig. 5(i)-(d), which is the result of the 2D approach [7]. Note that the *bunny* object which is actually present at a different depth is registered except for the borders, since the intensities within the object are mostly homogeneous. The size of the blur along borders though, is different, and hence, the border is clearly marked as a change. We fill and extract these binary regions using the method in Section 2.3.2. The filled-in binary object image showing the extracted objects is shown in Figs. 5(i)-(e).

Following Section 2.3.3, we find the RMSE between the object regions in the reblurred image at different depths and the RSMB image, and this variation of RMSE is plotted in Fig. 6(i). We set the threshold value to be 20. We can observe from the plot that the *bunny* gets registered at a relative depth of 0.5 with an RMSE value 8.84, but the *block* does not register at all at any depth, since the RMSE values are consistently high. Hence, the *block* is marked as a change as shown in Fig. 5(i)-(f). The estimated depthmap is shown in Fig. 5(i)-(g) with detected changes marked in red.

Case-(ii): We now use the reference image with neither *bunny* nor *block* as shown in Figs. 5(ii)-(a). The RSMB image is shown in Fig. 5(ii)-(b) which is same as in the previous case. The background-registered image is shown in Fig. 5(ii)-(c). The detected regions of changes and extracted objects are shown in Figs. 5(ii)-(d) and (e), respectively. In background layer registration, both objects are detected as changes, since they do not belong to the background. Further now, for both objects, there is no relative depth that could register them with the reference as can be observed from the plot of Fig. 6(ii). Hence, our method detects both the objects as changes correctly, as shown in Fig. 5(ii)-(f). The estimated depthmap is shown in Fig. 5(ii)-(g) with changes marked in red.

Case-(iii): With the same RSMB image as before as shown in Fig. 5(iii)-(b), we now use the reference image in Fig. 5(iii)-(a) with both objects, *bunny* and *block*, present. Our method correctly registers the two objects at relative depths

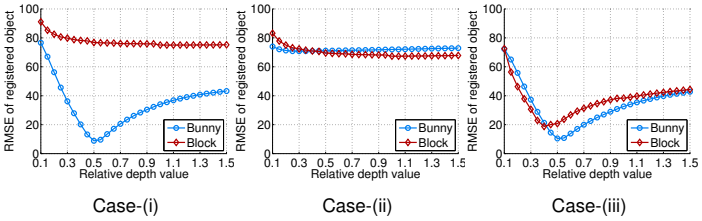


Fig. 6. RMSE vs. d_ℓ for registration of *bunny* and *block* in Fig. 5.

0.5 and 0.4 after background registration corresponding to the lowest RMSE values in Fig. 6(iii). Thus, our method reports no changes between the reference and RSMB images as shown in Fig. 5(iii)-(f). The estimated depthmap is shown in Fig. 5(iii)-(g), and there are no red regions corresponding to detected changes.

3.2 Quantitative Comparisons

To perform quantitative analysis, we created a synthetic dataset for two-image change detection. We cover the following scenarios: (a) motion estimation with no change, (b) change detection in a planar scene, and (c) change detection in a 3D scene. The types of camera motion considered are (i) 2D (t_x, t_y), (ii) 3D (t_x, t_y, r_z), and (iii) 3D (r_x, r_y, r_z). Here, x and y denote horizontal and vertical axes, respectively, and z denotes the optical axis. Both uniform and nonuniform camera motions are considered. For (a), we estimate camera motion using our method without appending the identity matrix for the change vector. For (b), we use our joint estimation method of (13), and for (c), we follow our Algorithm 1. For comparisons, we use [17]+[20] and [17]+[22] for RSMB images. For RS-only images, we directly perform RS registration using [20] and [22] without deblurring. The joint RSMB framework of [26] does not perform well even for planar scenes as shown in Fig. 20 in the supplementary material. Hence, we do not provide its quantitative comparisons on the synthetic dataset, but only show results on real images in Sec. 3.3. We use the following metrics suggested in [39] for change detection: (i) *precision* (Pr), (ii) *percentage of wrong classification* (PWC),

TABLE 1
Comparison of RMSE values between RS/RSMB and registered images for different methods in the case of no change.

RSMB	Uniform			Nonuniform		
Method	t_x, t_y	t_x, t_y, r_z	r_x, r_y, r_z	t_x, t_y	t_x, t_y, r_z	r_x, r_y, r_z
[17]+[20]	6.78	7.75	7.48	6.39	5.97	8.44
[17]+[22]	6.79	4.42	5.19	6.05	5.23	6.92
Ours	**3.43**	**1.46**	**1.13**	**1.85**	**1.17**	**2.26**

RS	Uniform			Nonuniform		
Method	t_x, t_y	t_x, t_y, r_z	r_x, r_y, r_z	t_x, t_y	t_x, t_y, r_z	r_x, r_y, r_z
[20]	2.67	7.53	7.61	6.32	6.09	7.88
[22]	3.19	4.33	4.16	4.92	4.52	5.25
Ours	**0.52**	**1.47**	**3.56**	**0.49**	**1.75**	**3.27**

TABLE 2
Comparison of quantitative metrics for different methods in change detection of planar scenes.

RSMB	t_x, t_y			t_x, t_y, r_z			r_x, r_y, r_z		
Method	Pr	PWC	Fm	Pr	PWC	Fm	Pr	PWC	Fm
[17]+[20]	0.80	3.29	0.76	0.81	3.69	0.75	0.81	7.45	0.59
[17]+[22]	0.80	2.87	0.78	0.80	2.40	0.82	0.81	2.91	0.79
Ours	**0.91**	**0.99**	**0.92**	**0.91**	**0.67**	**0.95**	**0.90**	**0.82**	**0.93**

RS	t_x, t_y			t_x, t_y, r_z			r_x, r_y, r_z		
Method	Pr	PWC	Fm	Pr	PWC	Fm	Pr	PWC	Fm
[20]	0.80	3.18	0.78	0.81	4.33	0.72	0.81	6.45	0.62
[22]	0.80	2.37	0.82	0.81	2.70	0.80	0.81	1.67	0.86
Ours	**0.90**	**0.82**	**0.93**	**0.90**	**0.79**	**0.94**	**0.90**	**1.26**	**0.90**

and (iii) *F-measure* (Fm). The formulae are provided in the supplementary material. Values closer to one are preferred for Pr and Fm, while a low value is preferred for PWC.

No-change performance: In Table 1, we show the performance of camera motion estimation of different methods. Our method consistently outperforms other methods due to our fine row-wise motion estimation as against the use of interpolation in other methods. The performances of other methods are better in the absence of motion blur (RS case) in comparison to their own results for the RSMB case, since deblurring inadvertently introduces artifacts. For uniform camera velocity in the RS case, [22] does well with RMSE less than 5. [20] performs well for the translation-only case as it deals with just that. The performances of both [20] and [22] reduce for the RSMB case, while the RMSEs of our method are still low.

Change Detection in Planar Scenes: For planar scenes, we add an object while creating RS/RSMB images using different motions. In Table 2, we show the metric values for all the three methods. Our Pr and Fm values are consistently at or above 0.90 in all the cases, while the other two methods perform worse. Similarly, PWC values are within 1 for our method, but they are higher for the other two methods. Our method comes first in all cases, while [22] comes next, since it is able to handle rotations as well, unlike [20].

Change Detection in 3D Scenes: For 3D scenes, we employ the one-object change case, since it is more challenging as one object has to be registered at a different depth from that of background, while the other object has to be detected as a change. Table 3 shows the results. We note here that [20] and [22] do not handle 3D scenes in the presence of translations, and their performance is worse, as expected. Our performance in the 3D scenario is as good as that for the planar case, which shows the effectiveness of our layered registration approach. We note here that filling-in of objects is not performed for the planar case unlike the 3D case, and hence the metric values for the planar case are slightly lower than that of 3D case for our method.

TABLE 3
Comparison of quantitative metrics for different methods in change detection of 3D scenes.

RSMB	t_x, t_y			t_x, t_y, r_z			r_x, r_y, r_z		
Method	Pr	PWC	Fm	Pr	PWC	Fm	Pr	PWC	Fm
[17]+[20]	0.77	3.62	0.57	0.77	4.69	0.56	0.79	7.40	0.42
[17]+[22]	0.77	3.55	0.59	0.77	4.33	0.56	0.79	2.39	0.70
Ours	**0.89**	**0.59**	**0.90**	**0.86**	**0.71**	**0.89**	**0.95**	**0.44**	**0.93**

RS	t_x, t_y			t_x, t_y, r_z			r_x, r_y, r_z		
Method	Pr	PWC	Fm	Pr	PWC	Fm	Pr	PWC	Fm
[20]	0.79	3.66	0.59	0.78	6.89	0.50	0.79	6.19	0.44
[22]	0.79	4.72	0.54	0.78	6.48	0.50	0.79	1.37	0.79
Ours	**0.99**	**0.35**	**0.95**	**0.98**	**0.48**	**0.94**	**0.99**	**0.33**	**0.95**

3.3 Real Experiments

For real experiments, we capture images using mobile phone cameras with motion to generate the desired RSMB effect. The reference image is captured with no camera motion. We use different mobile phones throughout our experiments: Google Nexus 4 for Figs. 7, 9(top) and Figs. 21, 24, and 25 in the supplementary, Motorola MotoG2 for Fig. 9(bottom), and Xiaomi Redmi for Fig. 22 in the supplementary. The images in Fig. 12 are drone images. The camera intrinsic matrix of each of these mobile phones is obtained by the precalibration procedure of Zhang [40].

2D Scene: The reference image is a scene with horizontal and vertical lines, and static objects as shown in Fig. 7(a). This is captured with a static mobile phone camera. We then add a new object to the scene. With the camera at approximately the same position, we recorded a video of the scene with free-hand camera motion. The purpose of capturing a video (instead of an image) is to enable comparison with Grundmann et al. [23]. From the video, we extracted a frame with high RS and MB artifacts and this is shown in Fig. 7(b). Our algorithm takes only these *two images* as input. We start Algorithm 1 by performing background layer registration and change detection simultaneously using (13). The registered reference image is shown in Fig. 7(c). The lines in the reference image are correctly registered as curves, since we forward-warp the reference image by incorporating RS effect. The presence of motion blur can also be noted. This elegantly accounts for both geometric and photometric distortions during registration. The change detection after background registration is shown in Fig. 7(d), which is also the result of the 2D approach [7]. We then extract other depth layers and object layers using the procedure described in Sec. 2.3.2, and the extracted objects are shown in Fig. 7(e). In this example, the scene is planar and there is only one object. Our algorithm correctly captures this during the object registration step and marks the object as the final change. Fig. 7(f) shows the detected changes in red with the background as the only layer marked in gray.

Comparisons: We first deblur the RSMB image using the nonuniform deblurring method of [17]. The reference and deblurred images are then given as inputs to the RS rectification algorithms of [20] and [22]. [20] tries to detect a translatory camera motion while the camera motion

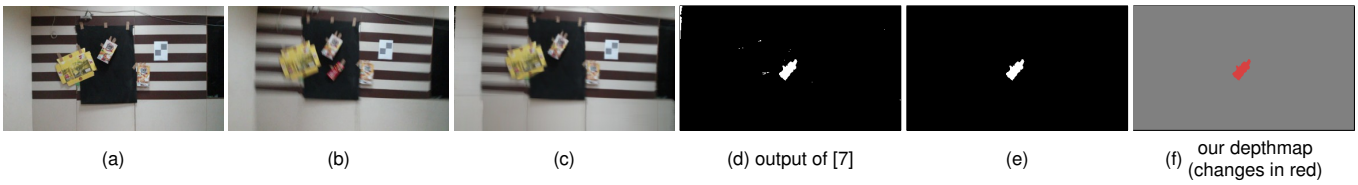
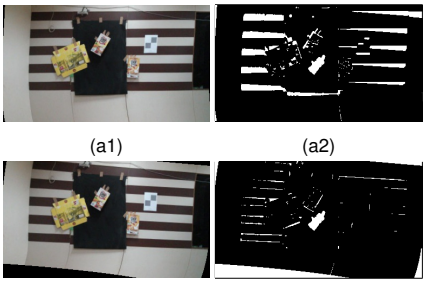
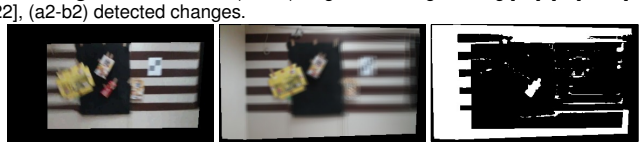


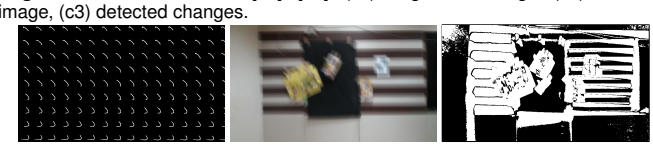
Fig. 7. **Real experiment**: Change detection in a 2D scene. (a) Reference image, (b) RSMB image, (c) background registered image, (d) detected changes [7], (e) object detection, and (f) estimated depth map and detected changes (red).



Deblur-Register framework: (a1-b1) Registered images using [17]+[20] and [17]+[22], (a2-b2) detected changes.



Register-Reblur framework [23]+ [12]: (c1) Registered image, (c2) reblurred image, (c3) detected changes.



Joint RSMB framework [26]: (d1) Local blur kernels, (d2) reblurred image, (d3) detected changes.

Fig. 8. Comparisons for the example in Fig. 7.

involves rotations too, and hence there are false positives in the detected changes as shown in Fig. 8(a2). [22] performs quite well though there are misregistrations along the edges which can be observed in Fig. 8(b2). This is due to its approximation of continuous camera trajectory by a piecewise linear motion interpolated using the camera poses at certain key rows. (This is demonstrated further in the supplementary material.) Thus, our method clearly performs better than these methods.

We next compare our algorithm with a serial framework that will rectify the RS effect first and then account for MB. We use the RS video rectification method of Grundmann et al. [23], and the state-of-the-art deblurring method of Whyte et al. [12] for non-uniform motion blur estimation. The captured video is given as input to [23] which gives an RS rectified video as output. The rectified frame corresponding to the RSMB image we had used in our algorithm is shown in Fig. 8(c1). We now estimate the global camera motion of the rectified image using [12]. We then apply the estimated camera motion from the rectified frame on the reference image, and detect the changes with respect to the rectified frame. This reblurred image are shown in Fig. 8(c2). Note that from Fig. 8(c3), the performance of change detection is much worse than our algorithm. The number of false positives is high as can be observed near the horizontal edges in Fig. 8(c3). Though the RS rectification of [23] works reasonably well to stabilize the video, the rectified video is not equivalent to a global shutter video especially in the presence of motion blur. The camera motion with non-uniform velocity renders invalid the notion of having a global non-uniform blur kernel.

We finally compare with the joint RSMB motion estimation framework of [26]. For the comparison to be fair, we first estimate the camera trajectory using [26], and then reblur the reference image using the motion to produce the RSMB effect. We then detect changes between the reblurred image and the observed RSMB image. The method of [26] does not model inplane rotations; the estimated local blur kernels are shown in Fig. 8(d1) and they are invariant within a row, whereas the actual motion involves variation of blur even within a row due to inplane rotation. Hence, the reblurred image in Fig. 8(d2) does not match well with the RSMB image in Fig. 7(b), and the change detection performance is quite poor as shown in Fig. 8(d3).

3D Scenes: We now show examples for change detection in real 3D scenes. In the first scenario, we capture a scene from the top of a building looking down using a mobile phone camera. The reference image is captured without moving the camera and is shown in Fig. 9[top](a). The distorted image is captured with predominant horizontal translatory motion of the camera. This can be observed from the shearing effect in Fig. 9[top](b). This image also has heavy motion blur, and it has two new objects as changes. The majority of the scene is the ground plane and can be considered planar, but the small parapet in the bottom right is at a distance different from that of the ground, and hence it incurs a different amount of blur and rolling shutter effect.

We first register the background plane using (13) as shown in Fig. 9[top](c). The ground gets registered, but the parapet does not, as expected. This can be seen from the border of the parapet in the detected changes after thresholding in Fig. 9[top](d). This is the result of the 2D approach [7], where the 3D nature of the scene is not taken care of. Though at this stage, the actual changes have also been correctly detected. We then extract each object by filling in the holes following the procedure in Sec. 2.3.2. For each object in Fig. 9[top](e), we find a scale so that it gets registered at a particular relative depth following Algorithm 1. The parapet gets registered at a relative depth of 0.8. The other two objects are not registered, and hence they are considered as the final changes, which are shown in Fig. 9[top](f).

Comparisons: In Fig. 10[top], we show the results of the joint RSMB method [26]. Since the RSMB image in

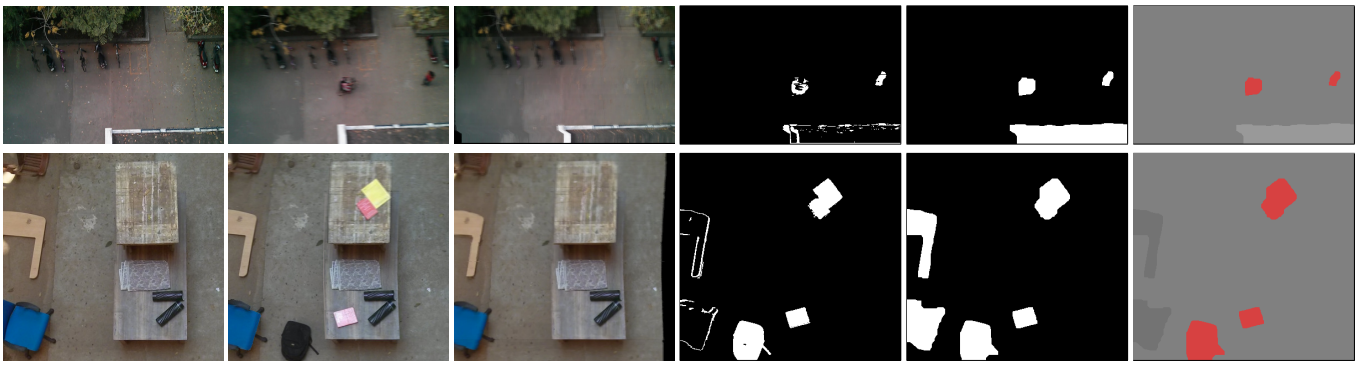
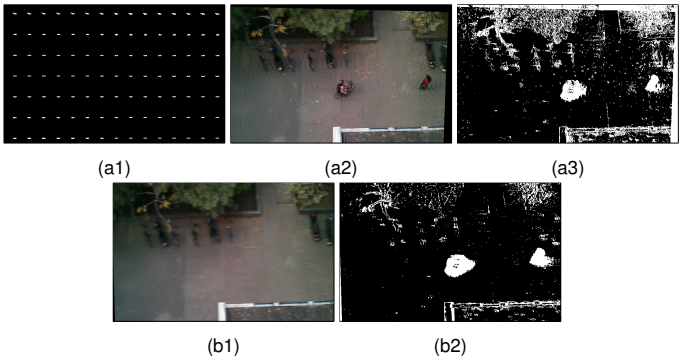
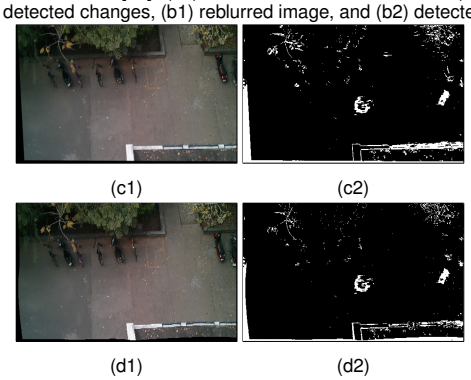


Fig. 9. **Real experiment**: Change detection in a 3D scene. (a) Reference image, (b) RSMB image, (c) background-registered image (d) detected changes after background registration (output of [7]), (e) extracted objects, and (f) estimated depth map and detected changes (in red).



Joint RSMB framework [26]: (a1) Estimated local blur kernels, (a2) deblurred image, (a3) detected changes, (b1) reblurred image, and (b2) detected changes.



Locally adaptive registration [17]+[33] and [17]+[32]: (c1-d1) Registered images, and (c2-d2) detected changes.

Fig. 10. Comparisons for the example in Fig. 9[top].

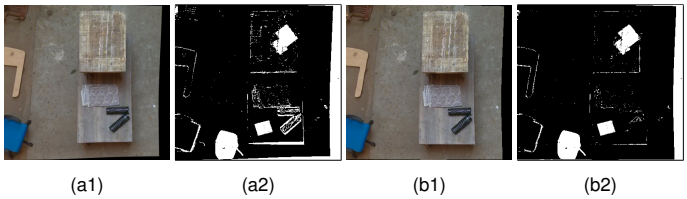


Fig. 11. Comparisons with locally adaptive registration methods, [33] and [32], for the example in Fig. 9[bottom]: (a1-b1) Registered images, and (a2-b2) detected changes.

Fig. 9[top](b) exhibits a predominant shearing effect (2D motion) and has negligible inplane rotations, it is expected to perform better than the earlier example. Fig. 10(a1) shows the estimated local blur kernels which are horizontal as expected, though the overall shearing effect is not captured completely, since the method operates only on a single image. Hence, we globally align the deblurred result with the reference image and then detect the changes. The deblurred image and detected changes are shown in Fig. 10(a2) and (a3), respectively. The performance is not as good due to the deblurring artifacts which creates much noise in the change detection result. We also detect changes between the reblurred image and the observed RSMB image (after global alignment). This result as shown in Fig. 10(b2) is better, though not on par with our result.

We also compare our output with locally adaptive registration algorithms of [33] and [32]. We give as inputs, the reference and deblurred RSMB images. The registered images using [33] and [32] are shown in Figs. 10(c1) and (d1), respectively. Both adapt the homographies of the reference image locally to try to register with the distorted image. But these adaptations are not perfect as can be observed by the difference images in Figs. 10(c2) and (d2).

Another example captured using handheld camera is shown in Fig. 9[bottom], in which there is minimal motion blur. The ground has very less texture in this example, and hence it gets registered with tables as the background layer. The objects on the ground are marked as changes after background registration (in Fig. 9[bottom](d)). This is also the output of the 2D approach [7]. The extracted objects on the ground get registered at a farther depth (with relative depth of 1.1) than that of the layer that has tables. The estimated depth map is shown in Fig. 9[bottom](f), and the detected changes are marked in red in the same figure. Since this example contains very minimal MB, the expectation is that local adaptive homography-based registration algorithms should work well. The registered and difference images using [33] and [32] are shown in Fig. 11. The method of [33] is not able to register local variations due to the RS effect in this example too, while [32] works reasonably well.

We also show the results of our registration for images captured from drones in Fig. 12[top] and [middle]. Both the reference and distorted images are picked from drone capture which are shown in Figs. 12(a) and (b). The background registered image is shown in Fig. 12(c), and the result of the 2D approach [7] is shown in (d). The 3D objects are

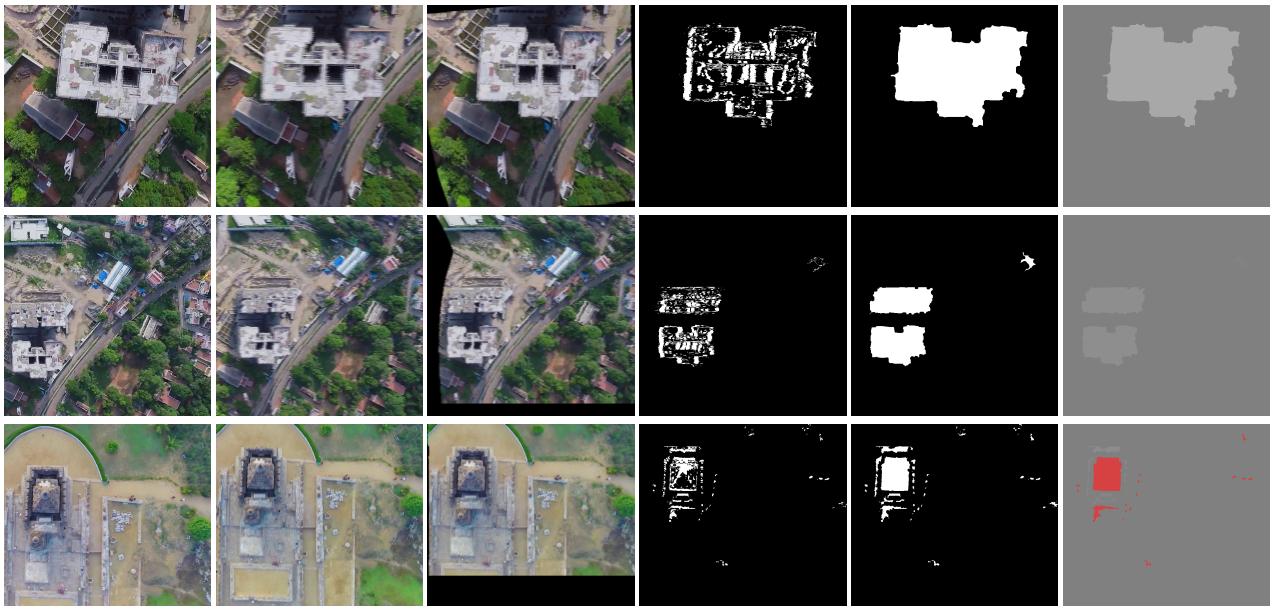


Fig. 12. **Drone imaging**: (a) Reference image, (b) RSMB image, (c) background registration, (d) result of the 2D registration method [7]), (e) detected 3D objects, and (f) estimated depth map (no changes are detected in top and middle rows).

also detected as changes, since they could not be registered using the ground layer poses. We then detect object layers (Fig. 12(e)), and each of the extracted object region from the reference image gets re-registered with the corresponding region of the RSMB image at a different scale corresponding to its relative depth. The building in Fig. 12[top] has the estimated relative depth of 0.68, and the relative depths of the three structures in Fig. 12[middle] are 0.9, 0.91, and 0.99. The estimated depth map is shown in Fig. 12(f). In [top] and [middle], there are no changes between the reference and the RSMB images. Our algorithm successfully registers the background layer and all 3D layers.

Fig. 12[bottom] shows an example imaging a temple complex where the two temple towers are pyramid-like structures. Due to the viewpoint change between Fig. 12[bottom](a) and (b), the structure of the temple towers in the two images are very different (note the change in the position of the tip of the towers). The camera motion is predominantly vertical, and there is a vertical stretch in the RSMB image. Our layered assumption does not adhere to this example and our method marks the temple towers also as changes as shown in Fig. 12[bottom](f). The ground layer is correctly registered and the other changes in the scene are also correctly detected.

4 Discussions

In this section, we discuss about choice of parameters, algorithm complexity, and sparsity. In the supplementary material, we discuss about how our RSMB model subsumes the GSMB model, tackling illumination variations, and the effect of the size of changed regions in the spatial domain.

Choice of Parameters: The bounds of the camera pose space and the step sizes of rotations and translations used here, work well on various real images that we have tested. In real experiments, we start with the registration

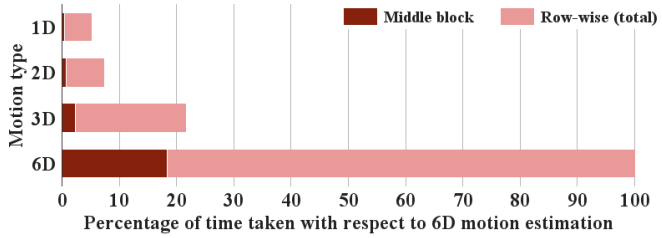


Fig. 13. Comparison of times taken to estimate camera motion with different degrees of freedom with respect to that of 6D motion.

of middle block of seven rows using a large pose space: $t_x, t_y = N(0, 8, 2)$ pixels, $R_x, R_y = N(0, 0.3, 0.1)^\circ$, and $R_z = N(0, 4, 1)^\circ$. The relatively smaller pose space is adaptively chosen for row-by-row registration: $N(t_{cx}, 3, 1)$ pixels, $N(t_{cy}, 3, 1)$ pixels, $N(t_{cz}, 0.1, 0.1)$ scale, $N(R_{cx}, 0.1, 0.1)^\circ$, $N(R_{cy}, 0.1, 0.1)^\circ$ and $N(R_{cz}, 1, 0.5)^\circ$. Decreasing the step sizes further increases the complexity, but provides little improvement for practical scenarios. The large bounding values for the middle block used suffice for most real cases. However, for extreme viewpoint changes, those values can be increased further, if necessary. We work on the intensity range [0–255]. We use $255\mathbf{I}$ in place of the identity matrix and thus $\boldsymbol{\chi}$ acts as a scaling factor on $255\mathbf{I}$ to compensate for the changed pixels. Hence, the value of the brightness constancy term in our framework is high, and the values of ℓ_1-norms are lower. Therefore, we employ high values for λ_1 and λ_2. λ_1 weights the sum of pose weight vector and λ_2 weights the ℓ_1-norm of $\boldsymbol{\chi}$. We observed that $\lambda_1 = 10^4$ and $\lambda_2 = 10^3$ work uniformly well in all our experiments.

Algorithm Complexity and Run-time: We use a gradient projection based approach to solve the ℓ_1-minimisation problem (13) using SLEP [34]. It requires a sparse matrix-vector multiplication with order less than $O(N(|\mathcal{S}| + N))$ and a projection onto a subspace with order $O(|\mathcal{S}| + N)$ in

each iteration with convergence rate of $O(1/k^2)$ for the kth iteration. Here N is the number of columns and $|\mathcal{S}|$ is the cardinality of the pose space (which is higher for the middle block). Run-times for our algorithm for different types of motion using an unoptimised MATLAB code without any parallel programming on a 3.4GHz PC with 16GB RAM are shown in Fig. 13 as a percentage taken for 6D motion. These values correspond to the examples shown in synthetic experiments section with RSMB image containing 256 rows and 384 columns. The time taken considering a 6D motion space is 1302 seconds ($\sim 4.5\times$ compared to 3D) and this is still tractable. As the number of free parameters in the camera motion increases, the run-time increases as expected, since the camera pose search space gets larger. In addition, the optimisation problem takes longer for the middle block, since the pose space chosen is larger. We do note here that, since the motion blur estimation of rows in the top-half and bottom-half of the image are independent, they are amenable to parallelisation.

Sparsity: Finally, we would like to add that imposition of sparsity on the region of changes which serves to accommodate differences due to 3D scenes or new/moving objects between images can be exploited even for applications such as mosaicing and super-resolution. However, for aerial imagery, this does limit the capture duration between the reference and probe image.

5 Conclusions

In this work, we formulated a novel unified framework to model the combined effect of rolling shutter and motion blur caused by camera motion for registration and change detection in 3D scenes. We proposed a sparsity-based optimisation algorithm which performs row-wise registration of the reference and distorted images and detects changes between them. Our layered approach could comfortably handle registration of both planar and 3D scenes. Experiments showed that our method performed better than individually employing rolling shutter rectification and motion deblurring as well as the existing joint formulation framework.

Acknowledgements We thank the reviewers and the associate editor for providing valuable comments and suggestions that have significantly improved the contents of the paper. The grant from the Asian Office of Aerospace Research and Development (AOARD) is gratefully acknowledged. The results and interpretations presented in this paper are that of the authors, and do not necessarily reflect the views or priorities of the sponsor or the US AFRL.

References

[1] B. Zitov and J. Flusser, "Image registration methods: A survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977 – 1000, 2003.

[2] P. Anuta, "Spatial registration of multispectral and multitemporal digital imagery using Fast Fourier Transform techniques," *IEEE Transactions on Geoscience Electronics*, vol. 8, no. 4, pp. 353–368, Oct 1970.

[3] W. Pratt, "Correlation techniques of image registration," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-10, no. 3, pp. 353–358, May 1974.

[4] X. Liu, Z. Tian, H. Xu, C. Leng, and F. He, "Registration of remote sensing images with steerable pyramid transform and robust sift features," in *International Workshop on Information Security and Application*, 2009, pp. 214–217.

[5] A. Cole-Rhodes, K. Johnson, J. LeMoigne, and I. Zavorin, "Multiresolution registration of remote sensing imagery by optimization of mutual information using a stochastic gradient," *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1495–1511, Dec 2003.

[6] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich, "Synchronization and rolling shutter compensation for consumer video camera arrays," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2009, pp. 1–8.

[7] V. Rengarajan, A. Rajagopalan, and A. Rangarajan, "Change detection in the presence of motion blur and rolling shutter effect," in *Computer Vision - ECCV 2014*, ser. LNCS. Springer, 2014, vol. 8695, pp. 123–137.

[8] J. Shade, S. Gortler, L.-w. He, and R. Szeliski, "Layered depth images," in *Proceedings of the 25th Annual Conference on Computer graphics and Interactive Techniques*, ser. SIGGRAPH '98. New York, NY, USA: ACM, 1998, pp. 231–242.

[9] S. W. Hasinoff and K. N. Kutulakos, "A layer-based restoration framework for variable-aperture photography," in *International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[10] A. Gupta, N. Joshi, C. Lawrence Zitnick, M. Cohen, and B. Curless, "Single image deblurring using motion density functions," in *Computer Vision - ECCV 2010*, ser. LNCS. Springer, 2010, vol. 6311, pp. 171–184.

[11] Y.-W. Tai, P. Tan, and M. S. Brown, "Richardson-Lucy deblurring for scenes under a projective motion path," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1603–1618, 2011.

[12] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce, "Non-uniform deblurring for shaken images," *International Journal of Computer Vision*, vol. 98, no. 2, pp. 168–186, 2012.

[13] Z. Hu and M.-H. Yang, "Fast non-uniform deblurring using constrained camera pose subspace," in *British Machine Vision Conference*, 2012, pp. 1–11.

[14] N. Joshi, S. B. Kang, C. L. Zitnick, and R. Szeliski, "Image deblurring using inertial measurement sensors," *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 30:1–30:9, Jul. 2010.

[15] S. Cho, H. Cho, Y.-W. Tai, and S. Lee, "Registration based non-uniform motion deblurring," *Computer Graphics Forum*, vol. 31, no. 7, pp. 2183–2192, 2012.

[16] H. Zhang and L. Carin, "Multi-shot imaging: Joint alignment, deblurring, and resolution-enhancement," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 2925–2932.

[17] L. Xu, S. Zheng, and J. Jia, "Unnatural l0 sparse representation for natural image deblurring," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2013, pp. 1107–1114.

[18] Z. Hu, L. Xu, and M.-H. Yang, "Joint depth estimation and camera shake removal from single blurry image," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 2893–2900.

[19] V. Rengarajan, A. Punnappurath, A. Rajagopalan, and G. Seetharaman, "Efficient change detection for very large motion blurred images," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2014, pp. 315–322.

[20] C.-K. Liang, L.-W. Chang, and H. Chen, "Analysis and compensation of rolling shutter effect," *IEEE Transactions on Image Processing*, vol. 17, no. 8, pp. 1323–1330, Aug 2008.

[21] S. Baker, E. Bennett, S. B. Kang, and R. Szeliski, "Removing rolling shutter wobble," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 2392–2399.

[22] E. Ringaby and P.-E. Forssén, "Efficient video rectification and stabilisation for cell-phones," *International Journal of Computer Vision*, vol. 96, no. 3, pp. 335–352, 2012.

[23] M. Grundmann, V. Kwatra, D. Castro, and I. Essa, "Calibration-free rolling shutter removal," in *International Conference on Computational Photography*, April 2012, pp. 1–8.

[24] M. Meilland, T. Drummond, and A. Comport, "A unified rolling shutter and motion blur model for 3d visual registration," in *International Conference on Computer Vision*, Dec 2013, pp. 2016–2023.

[25] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *International Journal of Computer Vision*, pp. 1–12, 2015.

[26] S. Su and W. Heidrich, "Rolling shutter motion deblurring," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2015, pp. 1529–1537.

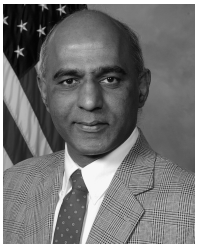
[27] J. Wulff and M. J. Black, "Modeling blurred video with layers," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 236–252.

[28] P. Bhat, K. Zheng, N. Snavely, A. Agarwala, M. Agrawala, M. Cohen, and B. Curless, "Piecewise image registration in the presence of multiple large motions," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 2491–2497.
[29] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 24:1–24:11, Jul. 2009.
[30] Z. Wu and A. Goshtasby, "Adaptive image registration via hierarchical voronoi subdivision," *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2464–2473, May 2012.
[31] F. Zhang and F. Liu, "Parallax-tolerant image stitching," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 3262–3269.
[32] J. Zaragoza, T.-J. Chin, Q.-H. Tran, M. Brown, and D. Suter, "As-projective-as-possible image stitching with moving DLT," *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. 36, no. 7, pp. 1285–1298, July 2014.
[33] M. Linger and A. Goshtasby, "Aerial image registration for tracking," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 4, pp. 2137–2145, April 2015.
[34] J. Liu, S. Ji, and J. Ye, *SLEP: Sparse Learning with Efficient Projections*, Arizona State University, 2009. [Online]. Available: http://www.public.asu.edu/~jye02/Software/SLEP
[35] J. Kapur, P. Sahoo, and A. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Computer Vision Graphics and Image Processing*, vol. 29, no. 3, pp. 273 – 285, 1985.
[36] P. L. Rosin and E. Ioannidis, "Evaluation of global image thresholding for change detection," *Pattern Recognition Letters*, vol. 24, no. 14, pp. 2345 – 2356, 2003.
[37] C. Paramanand and A. Rajagopalan, "Shape from sharp and motion-blurred image pair," *International Journal of Computer Vision*, vol. 107, no. 3, pp. 272–292, 2014.
[38] Wikipedia, "List of tallest buildings in different cities in India," Mar 2016. [Online]. Available: https://en.wikipedia.org/wiki/List_of_tallest_buildings_in_different_cities_in_India
[39] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "A novel video dataset for change detection benchmarking," *IEEE Transactions on Image Processing*, vol. 23, no. 11, pp. 4663–4679, Nov 2014.
[40] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

Vijay Rengarajan received his B.E. degree in electronics and communication engineering from PSG College of Technology, India in 2008. After working in Motorola India Private Limited, he is now pursuing his Ph.D. degree with Indian Institute of Technology Madras. His research interests lie in the areas of image processing and computer vision with applications to image registration and geometrical correction in rolling shutter cameras. He also has interests in computational photography.

Rangarajan Aravind received the B.Tech. degree from IIT Madras, and the M.S. and Ph.D degrees from U.C. Santa Barbara, all in Electrical Engineering. He was a Member of Technical Staff at AT & T Bell Laboratories where his work was in image and video compression algorithms. He was an active participant in the MPEG-1 and MPEG-2 video codec standardization activities. From September 1995, he has been a faculty member in the Department of Electrical Engineering, IIT Madras. He currently serves as a Professor. His R&D activities here includes real-time speech and video coding as well as modem simulation studies.

Ambasamudram Narayanan Rajagopalan received the Ph.D. degree in electrical engineering from IIT Bombay, in 1998. He was with the Center for Automation Research, University of Maryland, as a Research Faculty Member, from 1998 to 2000. He joined the Department of Electrical Engineering, IIT Madras, in 2000, where he is currently a Professor. He co-edited a book entitled Motion Deblurring: Algorithms and Systems (Cambridge University Press, 2014) along with Prof. R. Chellappa with the University of Maryland. He has co-authored a book entitled Depth From Defocus: A Real Aperture Imaging Approach (New York: Springer-Verlag, 1999). His research interests include 3D structure from blur, motion deblurring, registration, superresolution, video microscopy, inpainting, matting, heritage resurrection, face recognition, image forensics, and underwater imaging. He is a fellow of the Alexander von Humboldt Foundation, Germany, the Indian National Academy of Engineering, and the Institution of Electronics and Telecommunication Engineers, India (by invitation). He was a recipient of the DAE-SRC Outstanding Investigator Award in 2012, the VASVIK Award in 2013, and the Mid-Career Research and Development Award from IIT Madras in 2014. He served as an Associate Editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence from 2007 to 2011, and is currently an Associate Editor of the IEEE Transactions on Image Processing. He was the Area Chair of CVPR 2012 and ICPR 2012, and the Program Co-Chair of ICVGIP 2010.



Guna Seetharaman (F'15) served as an Associate Professor of Computer Science and Engineering with the University of Louisiana at Lafayette from 1988 to 2003, and the Air Force Institute of Technology from 2003 to 2008. He was a CNRS Invited Professor with the University of Paris XI, where he was involved in multiple tenures from 1998 to 2005, and held a Visiting Distinguished Professorship with IIT Mumbai in 2006. He is currently a Principal Engineer in Computing Architectures, and Video Information Processing and Exploitation with the Information Directorate, Air Force Research Laboratory, Rome, NY. He is focused on high performance computing, computer vision, machine learning, content-based image retrieval, persistent surveillance, and computational science and engineering. He initiated and established Computer Vision and Robotics Laboratories with the University of Louisiana at Lafayette. He co-founded the Team CajunBot, a participant in DARPA Grand Challenge. He led the LiDAR data processing and obstacle detection efforts with the Team CajunBot, demonstrated in DARPA Grand Challenges in 2005 and 2007. He was a member of the AFIT-Based Core Team for demonstrating and transitioning a wide area persistent imaging and surveillance system known as Angel Fire. He and his team received the IEEE CVPR 2014 Video Challenge for Change-Detection.

Dr. Seetharaman has authored over 175 peer-reviewed articles in computer vision, low-altitude aerial imagery, parallel computing, very large scale integration signal processing, 3D displays, nanotechnology, micro-optics, and 3D video analysis. He is also a member of Tau Beta Pi, Eta Kappa Nu, and Upsilon Pi Epsilon. He is a Paul Harris Fellow of the Rotary International. He co-organized the DOE/ONR/NSF Sponsored Second International Workshop on Foundations of Decision and Information Fusion in 1996 (Washington DC), and the IEEE Sixth International Workshop on Computer Architecture for Machine Perception, New Orleans, in 2003. He was a Guest Editor of the IEEE Computer special issue devoted to Unmanned Intelligent Autonomous Vehicles in 2006. He was also a Guest Editor of a special issue of the EURASIP Journal on Embedded Computing in the topics of intelligent vehicles. He is an active member of the Association for Computing Machinery. He served as the Section Chair of the IEEE Mohawk Valley Section from 2013 to 2015. He is an Associate Editor of the ACM Computing Surveys and a Fellow of the IEEE.

Supplementary Material

Image Registration and Change Detection under Rolling Shutter Motion Blur

In this supplementary material, we show (i) evaluation of camera motion estimation in the absence of changes, (ii) change detection in planar scenes and comparisons, (iii) change detection in global shutter motion blur images, (iv) handling illumination variation, (v) the effect of size of the changes, and (vi) evaluation metrics.

1 Camera Motion Estimation

We first compare the performance of our method with other methods on camera motion estimation (with no changes in the scene). In Fig. 14, we show two images, a reference clean image in Fig. 14(a) and an image of the same scene distorted with RS and MB in Fig. 14(b). The camera motion applied synthetically on the image is in-plane translatory motion but of nonuniform velocity. The scene is assumed to be planar with only one layer.

(a) (b) (c) (d)

Fig. 14. Synthetic experiment: Camera motion estimation. (a) Reference image, (b) RSMB image, (c) Registered image, and (d) Difference image.

To register these two images, we need to solve (13). We consider the change vector to be absent here to study the efficiency of our camera motion estimation. In general, our method does not require the prior information about the presence or absence of changes in the scene. The change vector values will be zero if there are no changes in the scene. In this experiment, we ignore the identity matrix part of $\mathbf{B}$ and estimate only $\boldsymbol{\omega}$. We start with estimating the camera motion for the middle row assuming the following large pose space $\mathcal{S}^{(M/2)}$: x-translation $N(0, 20, 1)$ pixels, and y-translation $N(0, 20, 1)$ pixels. We restrict the out-of-plane translation values and rotation values to zero. Once the camera pose weight vector is estimated for the middle row, we select the following neighbourhood search space for the other rows following Section 2.4: x-translation $N(t_{cx}, 3, 1)$ pixels, and y-translation $N(t_{cy}, 3, 1)$ pixels. The registered image is got by applying the estimated camera motion on the reference image and is shown in Fig. 14(c). The thresholded difference between the registered image and the distorted image is shown in Fig. 14(d).

Comparisons: We first deblur the RSMB image using [17]; this result is shown in Fig. 15(a). We then estimate the camera motion between the reference and deblurred images using the RS methods of [20] and [22]. We then apply the camera motion on the reference image to register with the deblurred image. The registered images using [20] and [22] are shown in Figs. 15(b) and (c), respectively. The difference images of (a)-(b) and (a)-(c), shown respectively in Figs. 15(d) and (e), contain significant nonzero values depicting misregistration. The RMSEs for our method, [17]+[20], and [17]+[22] in the valid region of the registered images are 2.22, 7.46, and 6.87, respectively.

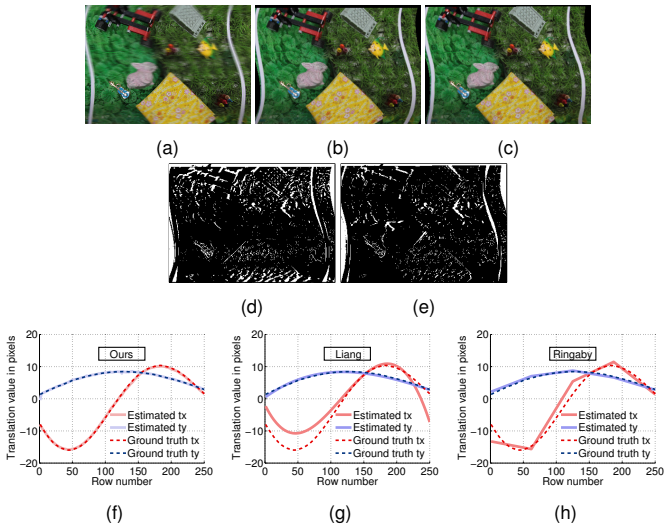


Fig. 15. Comparison with deblur-register MB-RS framework: (a) Deblurred image using [17]. Registered images after RS motion estimation using (b) [20], and (c) [22], and Difference images (d): (a)-(b), (e): (a)-(c). Camera motion estimation for nonuniform camera path using (f) our method, (g) [20], and (h) [22].

In Fig. 15(f-h), we show the camera motion trajectories estimated by our method and other methods. The camera motion estimated by our method, shown in Fig. 15(f) as a continuous line, follows correctly the ground truth trajectory shown as a dotted line. The camera motion estimated by the methods of [20] and [22] are given in Figs. 15(g) and (h), respectively. Since the camera motion has a nonuniform velocity, both the competing methods result in inaccurate motion estimation. Liang's method fits a Bézier curve for the trajectory using the estimated local motion vectors, which results in incorrect fitting. Ringaby's method follows the ground truth trajectory, but it is inaccurate due to the use of interpolation. The RMSEs of the translation motion estimates (t_x,t_y) in pixels of Liang's, Ringaby's, and our method are: (3.45,0.28), (1.79,0.37), and (0.18,0.14), respectively, for the range of [-15,15]pixels for t_x and [0,10]pixels for t_y. The corresponding maximum errors are (8.81,0.81), (5.29,0.82), and (0.80,0.44), respectively. Clearly, the performance of our method is better. Since there are no ground truth changes, the percentage of wrong classification (PWC) provides a quantitative measure for number of estimated wrong changes. The PWC of Liang's, Ringaby's, and our method are 13.9, 10.4, and 0.3, respectively, which indicates that our method detected very less false positives. However, both these methods work well if the camera velocity is uniform *and* the motion blur is negligible.

To confirm the correctness of our implementations of Liang [20] and Ringaby [22], we show an example in Fig. 16 involving registration of an RS-only image. The simulated camera velocity is *uniform*, and each row of the RS image in Fig. 16(b) corresponds only to a single warp of the reference image in Fig. 16(a). The registered image using our method and the other two methods are shown in Figs. 16(c) to (e). The corresponding RMSEs are 0.54, 3.58, 3.67, which show that the registrations using all the three methods are good.

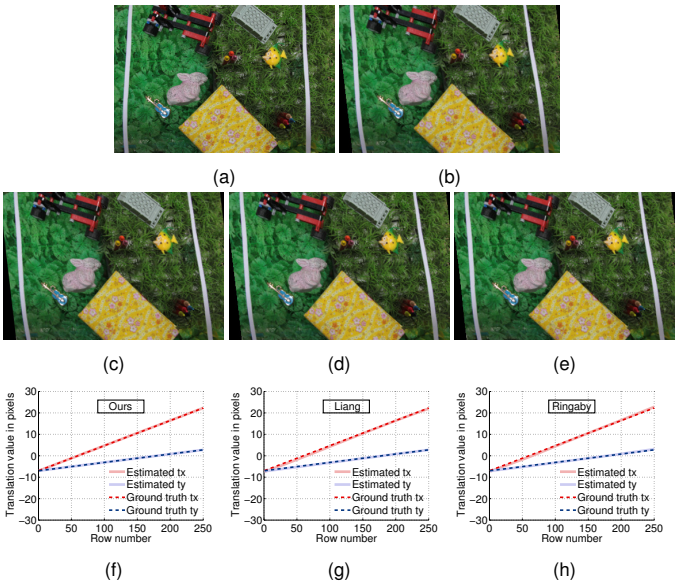


Fig. 16. Synthetic experiment: (a) Reference image, (b) RS image caused by uniform camera motion, and Registered images using (c) our method, (d) [20], and (e) [22]. Camera motion estimation for uniform camera velocity using (f) our method, (g) [20], and (h) [22].

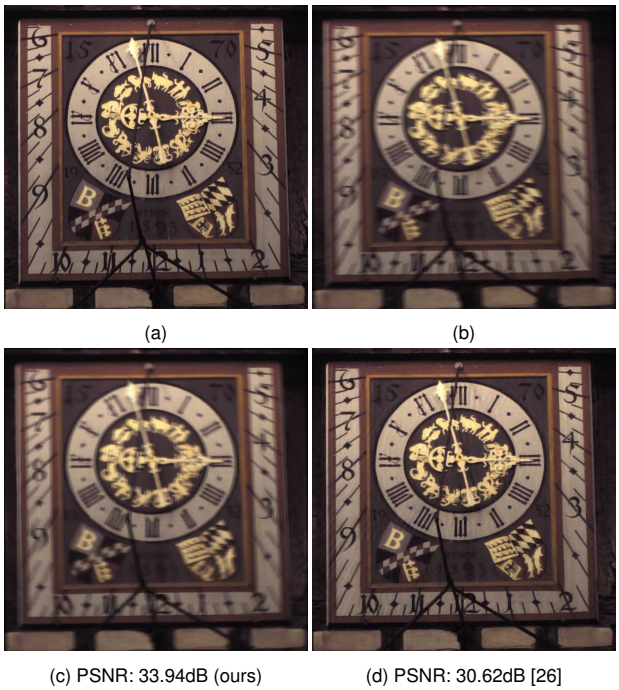


Fig. 17. Synthetic experiment: (a) Reference image, (b) RSMB image caused by 2D out-of-plane rotational motion, (c) registered image using our method, and (d) deblurred image of (a) using [26].

Competing methods perform well in the presence of only RS effect with negligible motion blur provided the camera velocity is uniform. In Figs. 16(f)-(h), we show the estimated camera motions for the simulated camera path. Our method estimates the camera trajectory very well, and so do the other two methods.

We also compare with the joint RSMB framework of [26] for the no-change scenario. We use the images from the synthetic examples provided in [26] for which the ground truth is available. Fig. 17(a) shows the reference image, Fig. 17(a) shows the RSMB image. The RSMB image contains only predominant 2D out-of-plane rotational motion and the in-plane rotational motion is very much negligible. We estimate row-wise motion using (a) and (b) and the RSMB registered image using our algorithm is shown in Fig. 17(c). The PSNR between Figs. 17(b) and (c) is 33.94dB corresponding to an RMSE of 5.12, which shows that the RSMB registration is correct. The deblurred output of Fig. 17(b) using [26] is shown in (d), and the PSNR between the reference image (a) and the deblurred image (d) is 30.62dB, which is also quite high (corresponding to an RMSE of 7.51) indicating correct deblurring. Thus, our method works as good as [26] for the case of 2D motion; for motion having inplane rotations, [26] performs poorly as will be discussed in the next section.

2 Change Detection in Planar Scenes

We now demonstrate our change detection method for planar scenes. We add new objects to the reference image in Fig. 18(a) as changes, and simulate a 3D in-plane camera motion, i.e. in-plane translations (t_x, t_y) and in-plane rotation (r_z). The RSMB image with objects is shown in Fig. 18(b). We solve (13) to estimate both the camera motion weight vector and the change vector. The pose space for the middle row is $N(0, 10, 1)$ pixels for translations, and $N(0, 5, 1)^\circ$ for rotation. The search space for other rows around the estimated

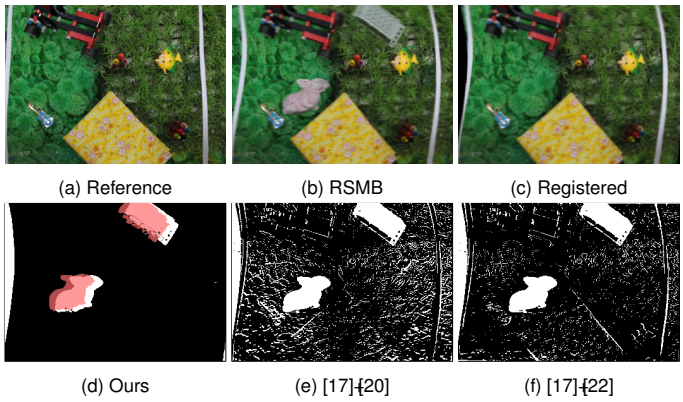


Fig. 18. Synthetic experiment: Change detection in a planar scene. (a) Reference image, (b) RSMB image, (c) Registered image, and (d) Detected changes by our method (marked in white in RSMB grid, and marked in red in reference grid). Detected changes by the method of (e) [17]+[20] and (f) [17]+[22].

centroid of their neighbour rows is $N(t_c, 3, 1)$ pixels for translations, and $N(r_c, 2, 0.5)^\circ$ for rotation. The registered image and detected changes are shown in Figs. 18(c) and (d), respectively. The white regions in Fig. 18(d) show that the detected changes are indeed correct. Note that it is also possible to obtain the locations of changes with respect to the reference image grid by marking pixels in the reference grid which map to the pixels marked as changes in the RSMB grid when the estimated camera motion is applied. The changes thus detected with respect to the reference grid are shown as a red overlay in Fig. 18(d). Figs. 18(e) and (f) show results using the deblur-register frameworks of [17]+[20] and [17]+[22], respectively. There are a high number of false positives in their outputs due to the lack of joint RSMB formulation.

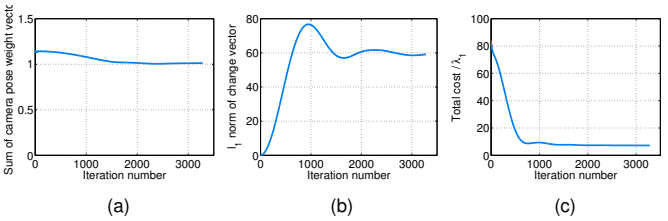


Fig. 19. Various energy values in our cost function in (13) while estimating motion for row number 121 of Fig. 18(b).

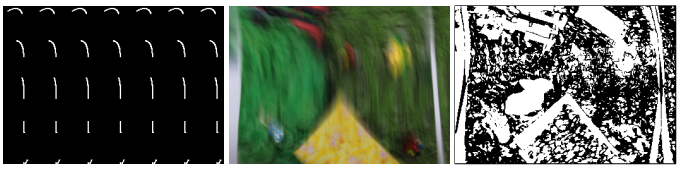


Fig. 20. Comparison with joint RSMB framework [26]: (a) Estimated local blur kernels, (b) reblurred reference image, (c) detected changes.

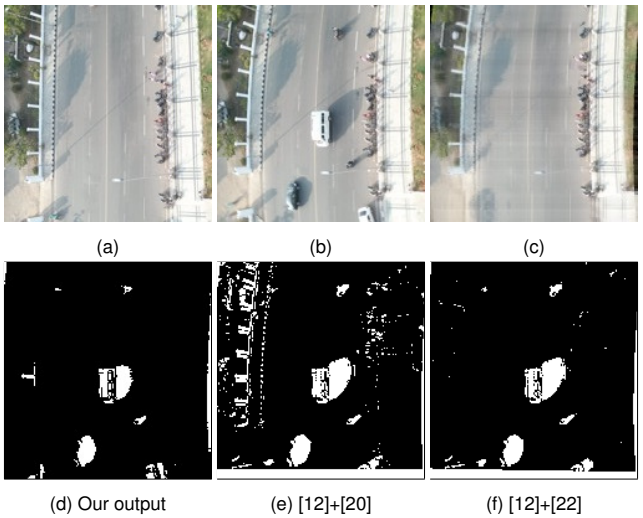


Fig. 21. Real Experiment. (a) Reference image, (b) RSMB image with prominent curves due to y-axis camera rotation, (c) Registered image, and Detected changes using (d) our method, (e) [12]+[20], and (f) [12]+[22].

We plot the various energy values in the cost function while solving (13) for 121st row (randomly chosen) in Fig. 19. Fig. 19(a) shows the value of $\|\boldsymbol{\omega}^{(121)}\|_1$ over iterations. This value converges to one conforming to the conservation of photometric energy. Fig. 19(b) shows the plot of $\|\boldsymbol{\chi}^{(121)}\|_1$ in which the changed pixels are picked correctly over iterations which can be observed by its stabilisation. The plot of the total cost (as a ratio of λ_1) over iterations in Fig. 19(c) shows the convergence of our algorithm.

To compare with the joint RSMB deblurring method [26], we first estimate the camera motion from the RSMB image, and then reblur the reference image using the estimated motion. This is to avoid artifacts due to the deblurring process. Fig. 20(a) shows the estimated local blur kernels, and Fig. 20(b) shows the reblurred image. Since the method does not model the complete camera motion (no inplane rotations), the reblurring result is very different from the RSMB image as can be seen from the detected changes in Fig. 20(c).

2.1 Change Detection – Additional Examples

We capture an image from atop a light house looking down at the road below. The reference image in Fig. 21(a) shows straight painted lines and straight borders of the road. The RSMB image is captured by rotating the mobile phone camera prominently around the y-axis (vertical axis). This renders the straight lines curved as shown in Fig. 21(b). Our algorithm works quite well to register the reference image with the RSMB image as shown in Fig. 21(c). The new objects, both the big vehicles and smaller ones, have been detected correctly as shown in Fig. 21(d). We do note here that one of the small white columns along the left edge of the road in the row where the big van runs, is detected as a false change. The RS rectification methods [20], [22] perform worse than our method, even though the motion blur in this example is not significant. The change detection outputs of [20] and [22] are shown in Figs. 21(e) and (f), respectively.

In the next scenario, the side of a building is captured with camera motion that results in curvy vertical edges in Fig. 22(b) as compared to 22(a). There is a single change between the two images. Change detection should compensate for these curves and detect the new object. Our method performs well as shown in Fig. 22(d) and is better than other methods. The registered images using our method, [20] and [22] are shown in Figs. 22(c), (e), and (g), respectively. The corresponding changes are shown in Figs. 22(d), (f), and (h) respectively.

Three examples of wide area motion imagery are shown in Fig. 23. The first row shows an example with global motion blur with ground and building depth layers, and the second row shows an example with global motion blur of a planar scene. The reference image is shown in (a), the distorted image is shown in (b), the registered image is shown in (c), and the detected changes are shown in (d). After detecting changes, we remove noise by removing components smaller than a fixed size using connected component analysis. Our method performs well on both the examples, detecting even small changes.

3 Change Detection of Motion Blur-only Images

As noted earlier in Section 2.2 of the main paper, our rolling shutter motion blur model subsumes the global shutter motion blur model. In this section, we show an example for a motion blur-only scenario arising from a global shutter camera. The images are captured using a Canon EOS 60D camera. Though this camera employs CMOS sensors, the rolling shutter effect is close to negligible for common camera motion, since unlike mobile phone cameras, the readout speed is very fast. The reference image in Fig. 24(a) is captured with no camera movement, while the distorted image in Fig. 24(b) is affected with motion blur due to camera motion. All rows of the image are affected by the same camera motion. We register the image through row-wise camera pose estimation, and the corresponding registered image and detected changes are shown in Figs. 24(c) and (d), respectively. Row-wise formulation estimates

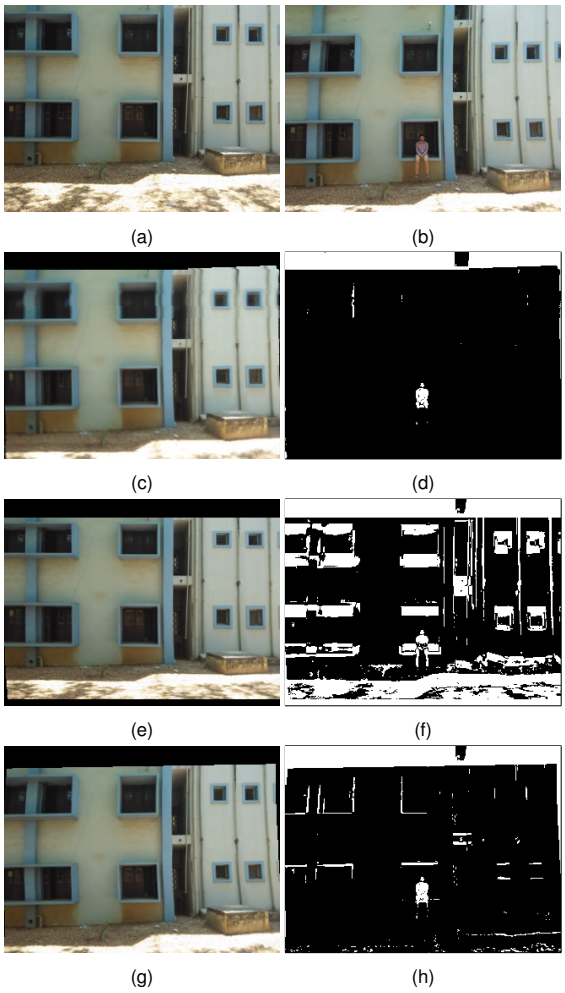


Fig. 22. Real experiment: Change detection in a 2D scene. (a) Reference image, (b) RSMB image, Registered image and detected changes using (c-d) our method, (e-f) [12]+[20], and (g-h) [12]+[22].

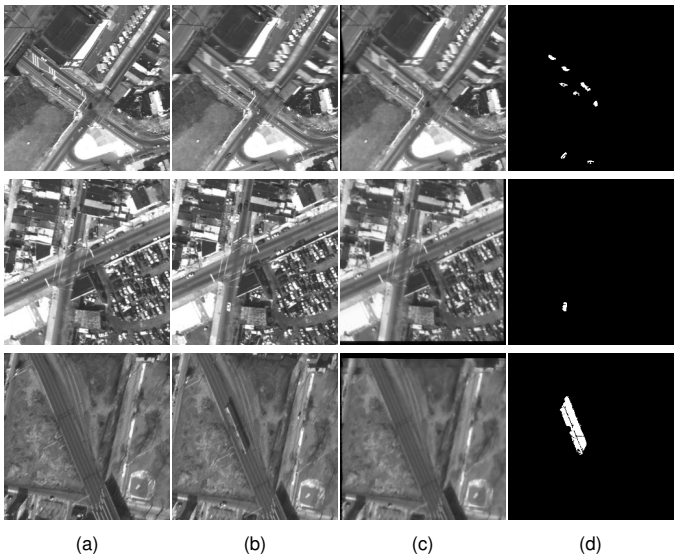


Fig. 23. Wide area motion imagery. (a) Reference image, (b) Distorted image, (c) Registered image, and (d) Detected changes.

almost same camera motion for every row, as shown in Fig. 24(e), which is indeed true for this global shutter case. With a prior knowledge of the type of distortion being global shutter motion blur, it is also possible to register the *whole* image (instead of row-wise registration) by estimating the weights for the camera poses and the change vector using (13).

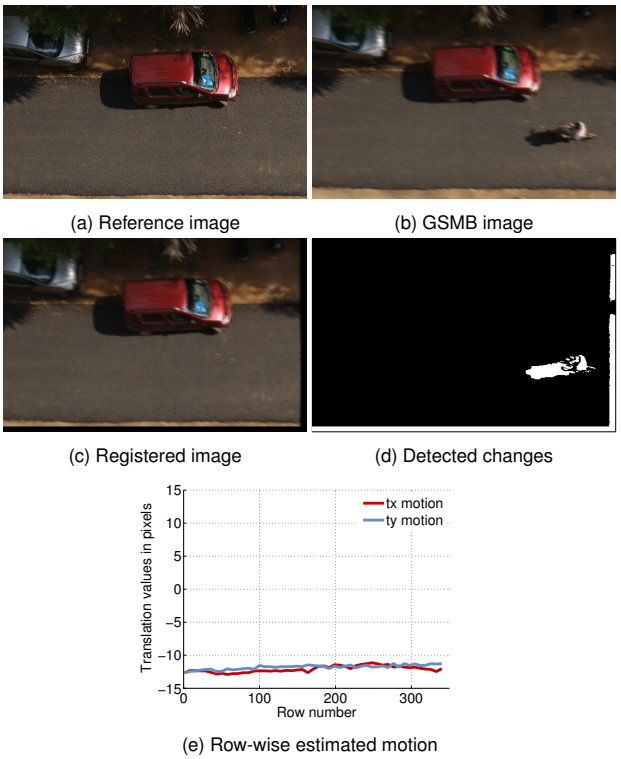


Fig. 24. Change detection in the presence of global shutter motion blur.

4 Handling Illumination Variations

The effect of illumination variations on the performance of our method is discussed next. Since the reference and observed images are captured at different times, there could be global as well as local illumination variations. Global variations primarily involve a change in overall ambient lighting, while local variations could be due to shadows or due to the introduction and removal of one or more light sources. Our method can take care of row-wise global variations through the camera pose weight vector. Under similar illumination conditions, our optimization problem results in a camera pose weight vector summing to one due to the conservation of energy between the two images. In the case when the illumination changes globally by a multiplicative factor γ, the same vector adapts itself by summing to γ. This is demonstrated in Fig. 25. The RSMB image is shown in Fig. 25(d), and the reference images for three different illumination conditions (*same*, *low*, *high*) are, respectively, shown in Figs. 25(a), (b), and (c). Changes are correctly detected at all three illumination conditions as shown in Figs. 25(e) to (g). The sum of estimated camera pose weight vector $\|\boldsymbol{\omega}^{(i)}\|_1$ of every row for these three cases are shown in Fig. 25(h). For the *same* case, this value is close to 1 for all rows; for the *low* case, it is above 1 to compensate

for the lower energy in the reference image due to a lower global illumination, and for the *high* case, it is below 1 due to a higher global illumination in the reference image.

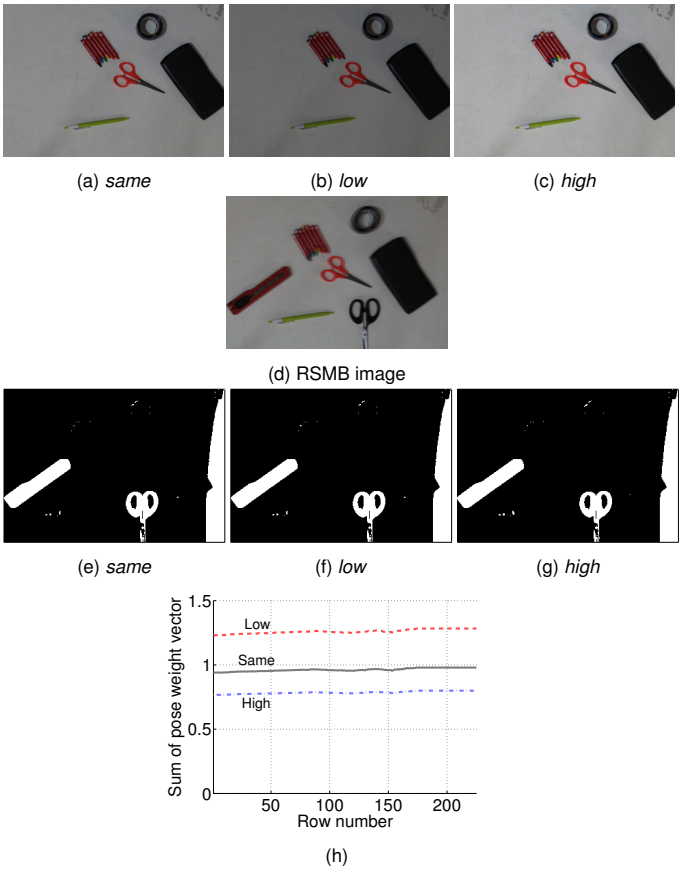


Fig. 25. Handling illumination variations. (a-c) Reference images captured under three different illumination conditions, (d) RSMB image captured under the same illumination condition as that of (a), (e-g) Detected changes, and (h) Sum of camera pose weight vector for these three cases.

5 Effect of the Size of Changed Region

The optimisation problem in (13) involves the sparsity of the changes present in the scene. In aerial imaging and surveillance scenarios, the changes are expected to cover only a small portion of the scene, and hence our algorithm works well in these scenarios. To study this behaviour, we created a random Gaussian grayscale image f size 256×256, and added the change as a block covering ten rows. The changed region is white (with intensity 255). We then introduced horizontal translatory RSMB effect on this image. We estimate the camera motion using the clean reference image and this RSMB image. The length of the change along the row is varied to study the performance of our method. The effect of length of the changed region on the motion estimation is shown as blue line in Fig. 26(a). As the length of the changed region increases, the average pixel error increases but not very much, since we use a local neighbourhood search space $\mathcal{S}^{(i)}$ for every row. A common global search space for all rows ($\mathcal{S}^{(i)} = \mathcal{S}^{(j)}$, for all i, j) results in worse motion estimation as the size of the changed region increases; this can be seen from the red line in the same figure.

We performed similar experiments for the GSMB case. The motion is estimated using the full image instead of a row-wise estimation. The resulting estimation error for different sizes of changed regions is shown in Fig. 26(b). We can observe in this scenario too, that increasing the size of the changed region affects motion estimation. Till the area covers 60% of the image area, the estimated error is within one pixel.

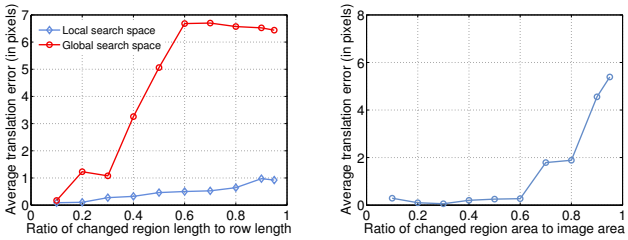


Fig. 26. Effect of the size of changed regions in motion estimation in the case of (a) RSMB and (b) GSMB.

6 Evaluation Metrics

One could use a number of metrics to compare the performance of change detection result against the ground truth. Certain metrics favour certain measures; a low False Negative Rate may be favoured by a particular metric, while a low False Positive Rate may be favoured by another. In the scenario of change detection, let TP represent the number of true positives (number of pixels that are correctly detected as changes), TN represent the number of true negatives (number of pixels that are correctly detected as nonchanges), FP represent the number of false positives (number of pixels that are wrongly detected as changes), and FN represent the number of false negatives (number of pixels that are wrongly detected as nonchanges). We use the following three metrics for comparisons in Section 3.2 of the main paper: (i) *precision*, Pr = TP / (TP + FP), (ii) *percentage of wrong classification*, PWC = 100(FN + FP)/(TP+FN+FP+TN), and (iii) *F-measure*, Fm = 2 (Pr · Re)/(Pr + Re), where *recall* Re = TP / (TP + FN). Pr penalises false positives; if FP is zero, then Pr is one. PWC penalises both FP and FN; the lower the PWC, the better the performance. Fm is proportional to the harmonic mean of recall and precision; close-to-one values are preferable.