Occlusion-Aware Rolling Shutter Rectification of 3D Scenes SUPPLEMENTARY MATERIAL

Subeesh Vasu[1], Mahesh Mohan M. R.[2], A. N. Rajagopalan[3]
Indian Institute of Technology Madras
subeeshvasu@gmail.com[1], ee14d023@ee.iitm.ac.in[2] raju@ee.iitm.ac.in[3]

The contents of this supplementary material are arranged as follows. (i) illustration of the main components involved in our image formation model, (ii) additional details on our theoretical findings and algorithm implementation, (iii) a comprehensive listing of the main steps involved in our rectification algorithm, and (iv) additional visual comparisons on both synthetic and real examples. In this supplementary material, numbers of equations and sections without 'S' are with respect to our main paper.

S1. Image formation model revisited

In Fig. S1, the main components of our proposed image formation model are illustrated with the help of a toy example. Here, we have simulated the image formation process of a three layer-scene and explain the differences between the captured images using an RS and a GS camera while undergoing translational motion. The first row contains the elementary units (latent intensities and masks for all the layers) which are used to represent the latent image of a scene. Second and third rows show the layer-wise motion involved in the image formation, if we were to capture the scene from a moving GS camera (*i.e.*, the image formation in Eq. 5). Since all the sensors in a GS camera get exposed to the scene at the same time, an image captured by a moving GS camera can be treated as equivalent to that of an image captured using a single camera pose. In order to generate the image for a given camera pose, we need to warp each layer using a layer-specific homography (Eq. 7) which is again a function of the plane parameters as well as the new camera pose ϑ. It can be observed in Fig. S1 that the pixels from each layer that are visible in the captured image (*i.e.*, pixels that are not occluded FG layers) are decided by the occluded layer masks.

For the case of an RS camera, different rows in the captured image can correspond to different camera poses. Therefore, the homography experienced by a row can potentially be different as compared to another row even if both rows come from the same layer. As shown in the fourth and fifth row of Fig. S1, the RS effect induces depth-dependent geometric distortions (*e.g.*, the vertical lines in the scene get slanted more for the layers closer to the camera) and self-occlusions (since the pixels that are visible in the image captured by a GS camera are not fully visible in the RS image) in the captured images.

S2. Algorithmic details

S2.1. Camera motion estimation and depth recovery

Optimization in Eq. 10 performs an outlier-robust estimation of camera motion and plane parameters, wherein the second term is an outlier robust cost. In each iteration, the outlier robust cost ensures that the optimization is not negatively influenced by correspondences from non-dominant layers. In each step, this optimization by default will classify the correspondences to two classes. The first class will correspond to the most dominant layer while all other correspondences goes to the second class. This leads to a robust estimation of plane parameters in each step, irrespective of the proportion of correspondences in each layer. Also, in our formulation, depth variations within a plane are already factored in. More specifically, the depth-variations induced by inclined planes are subsumed by the plane parameters (normal ($\mathbf{n}$) and perpendicular distance (d)).

S2.2. Iterative graph cut and visibility map

Due to space constraints, we provided only a brief discussion on the iterative refinement of occluded layer mask in Section 3.2.1. We will now provide a detailed discussion of our approach for iterative refinement of pixel labeling and visibility map. In the first iteration, to solve Eq. 12, we assign visibility map as 1 for all the pixels, and assign a small scalar value as the cost to o. Thus the solution of Eq. 12 automatically maps the low-confident candidate pixels to label o (which represents the pixels whose label cannot be assigned with high confidence). Since the occluded pixels will have a high cost for all the labels, they get classified under o. The estimate for $\tilde{\boldsymbol{\alpha}}^i_{r,m}$ returned by Eq. 12 is then used to compute a refined visibility map using Eq. 16. The

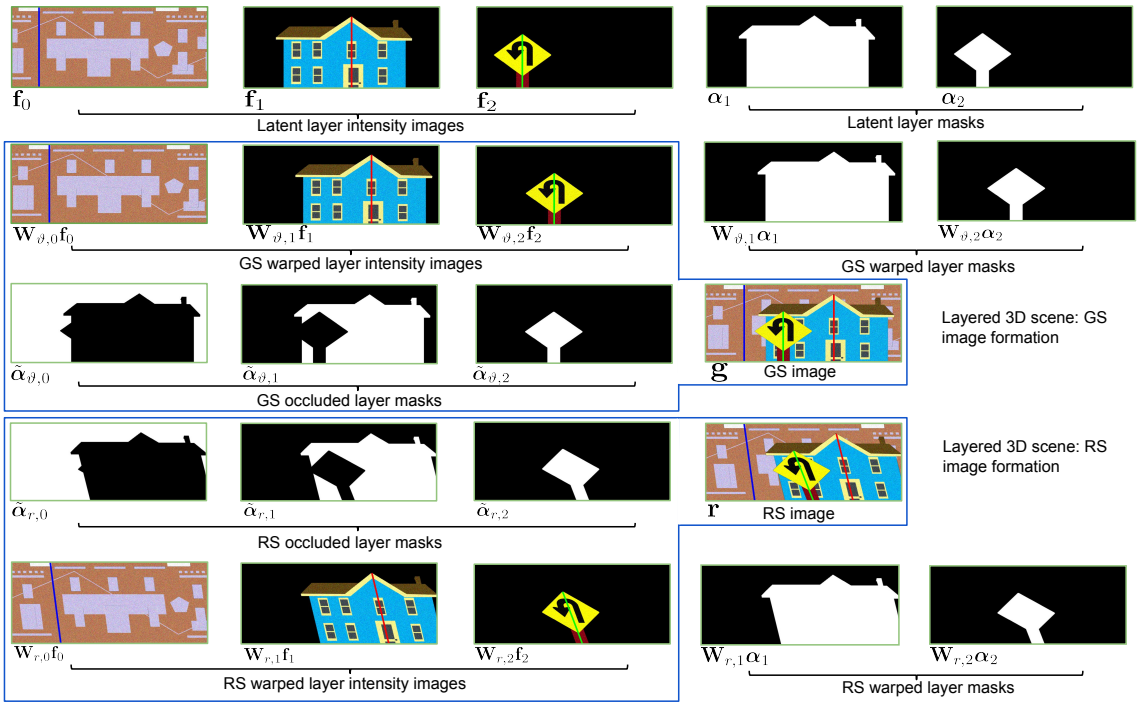

Figure S1. Layered model of 3D scenes. An example for a scene with 3 layers (with depth ordering $d_0 < d_1 < d_2$) illustrating the image formation in an RS camera. The first row represents the latent layer intensities and masks that we have used to synthesize the RS image $\mathbf{r}$ and latent (or GS) image $\mathbf{g}$. Second and third row simulate the warp effects on the layer intensities and masks in a GS camera. Fourth and fifth row illustrate the warp effects associated with an RS camera. For each case, we show the warped forms of layer intensities, masks, and finally the occluded layer masks. The occluded layer masks decide the visible pixels in the image plane, which when combined with the RS distorted layer intensities generates the captured GS/RS image. It can be observed that, the RS image contains depth-dependent geometrical distortions where the distortion experienced by the layer close to the camera ($\mathbf{f}_2$, $\boldsymbol{\alpha}_2$) is more severe as compared to BG layers.

refined visibility map helps to reduce the effect of outliers (generated due to occlusion effects) in the cost assignment. The visibility map obtained from the first iteration is then used to update the data cost used in Eq. 12. Also, since the pixels that have labels other than o were assigned with high confidence, we do not want to make any changes to these labels. Hence we freeze these pixel labels and then re-do the label assignment for the remaining pixels with the updated data cost. The updating process of both the visibility mask and data cost along with progressive freezing of high-confidence assignments is continued until convergence. In each iteration, we will also increase the occlusion cost to ensure efficient convergence of the entire optimization. In our implementation, we begin by assigning the occlusion cost as 5 and then in each iteration we increase it by a factor of 1.2.

Figs. S3(d-g) show the improvement in the label assignments (for an image sequence from our dataset S_2) as iteration progresses. In each image shown in Figs. S3(d-g), the white pixels correspond to the label o, and other labels are displayed with lower gray values, with the pixel value for BG label being zero. It can be observed that the iterative refinement allows us to handle the negative influences of occlusion effects in label assignment and the mask estimation converges to the desired solution after few iterations.

S2.3. Regularization for fractional mask recovery

As we explained in Section 3.2.2, when there exists significant translational motion of the camera, portions of the occluded regions in one input image will be visible in some other images. We have exploited this possibility to find the background layer intensity values $\xi(u)$ for the pixels in the uncertain region of the trimap. To find the values of $\xi(u)$ we use the following procedure. For each uncertain pixel (say u) in $\mathbf{r}^i$, we identify the lower most layer (say l') among the two layers it can belong to. Now, layer l' represents the background layer and we are looking for the pixel intensities from layer l' at u. For each layer (say l) and input image $\mathbf{r}^i$, we generate a binary mask $\Psi^i_{r,l}$ with ones over the region which will surely enclose all the pixels that belong to l. $\Psi^i_{r,l}$ is generated by assigning the uncertain regions in trimap $\boldsymbol{\varrho}'$ to 1. The visible pixels from l' in other images

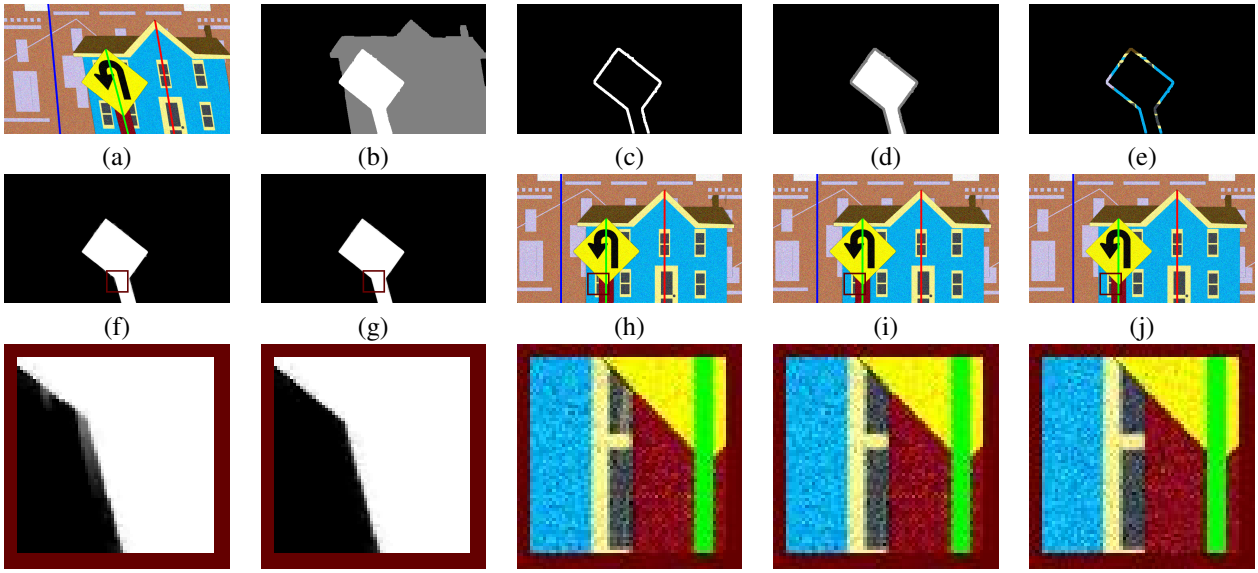

Figure S2. Synthetic experiment: Importance of the proposed regularization for the recovery of fractional layer mask. (a) Reference input image. (b) Occluded layer mask of reference frame obtained through iterative graph cut. For the foremost layer, (c) selected confusion region and (d) generated trimap for mask refinement through alpha matting. (e) Background layer estimate (obtained from neighboring frames) for the regularization of alpha matting. Fractional layer mask obtained through (f) direct alpha matting in [2] and (g) our regularized (using background layer estimate in (e)) alpha matting. Rectified image with mask estimate from (h) direct alpha matting ([2]) and (i) regularized alpha matting. (j) Ground truth latent image. (k-o) Zoomed in patches from (f-j).

are then combined to yield the true intensity value $\xi(u)$ at u. Visible pixels from l' can be found by inspecting the regions in other images that get excluded by the occlusion maps (obtained by applying Eq. 17 on $\Psi^i_{r,l}$) of all the layers l which can potentially occlude the layer l' (*i.e.*, $\forall\ l > l'$). This process is repeated to identify the true intensity values corresponding to all uncertain pixels in $\mathbf{r}^i$.

An example to reveal the importance of our proposed regularization scheme is shown in Fig. S2. Fig. S2(c) shows the uncertain region corresponding to the foremost layer over which we intend to refine the mask values. For the fractional layer mask recovery, we use the trimap shown in Fig. S2(d), and the background layer intensity (Fig. S2(e)) obtained from neighboring frames. To highlight the importance of regularization, we show the fractional masks retrieved with and without regularization in Figs. S2(f-g). It can be observed from the zoomed-in portions shown in Figs. S2(k-l) that our proposed regularization helps to mitigate the errors that generated by the direct use of the approach in [2]. Figs. S2(h-i) show the restored images obtained by using the alpha matte obtained from [2] and our proposed approach, respectively. By comparing with the ground truth image in Fig. S2(j) it can be seen that the error present in the alpha matte obtained from [2] creates visible artifacts along the layer boundaries, whereas the alpha matte obtained using our regularization scheme results in visually pleasing outputs devoid of such artifacts.

S2.4. Additional implementation details

In our experiments, among all the input images $\mathbf{r}^i|_{i=1}^{\nu}$, we choose the middle frame (say e) as the reference RS image. While estimating the camera motion using Eq. 10 we define the camera pose ($\mathbf{p}_1^e$) corresponding to the first row of $\mathbf{r}^e$ as the origin of camera trajectory. The camera motion obtained by enforcing such a condition results in a latent image corresponding to the camera pose $\mathbf{p}_1^e$. For the optimization in Eq. 10, the first row of all the input images are always considered as key-rows. For the cases with the camera motion involving significant translations, we apply RANSAC on all the point correspondences to find the most dominant translational shift and use it to initialize the translational pose vectors for the first row (all other dimensions of these pose vectors are initialized to 0) of every frame except the reference frame. The camera pose at every other key-row is initialized by values obtained through interpolation with respect to the initial pose vectors at the first rows of subsequent frames.

The optimization problems in Eq. 10 and Eq. 18 are solved using MATLAB functions *lsqnonlin* and *mldivide* respectively. We have used empirically found values of n_k, γ_o, k, μ_t, λ_s, λ_1, λ_2, λ_3, $\lambda_{\mathbf{f}}$ in our experiments. For real experiments, the value of focal length (q) was set to 2929.64 pixels (obtained via camera pre-calibration). To build W_r^i, the camera pose of each row is obtained by interpolating from adjacent key-rows (as discussed in Section. 3.1). The

Algorithm 1 Occlusion-aware RS image rectification

Input: Set of RS distorted frames $\{R^i\}_{i=1}^{\nu}$.

Output: Camera pose trajectory ($\mathbf{p}$), plane parameters ($\mathbf{n}_l$, $d_l|_{l=1}^{L-1}$), layer masks ($\boldsymbol{\alpha}_l|_{l=1}^{L-1}$), layer intensity images ($\mathbf{f}_l|_{l=0}^{L-1}$), and latent image ($\mathbf{f}$).

1: Compute optical flow across all consecutive frames. Find the camera motion and plane parameters (and true layer correspondences) of the dominant layer by setting number of layers in Eq. 10 to 1.
2: Using the estimated camera motion and remaining optical flow correspondences, solve Eq. 10 in an iterative fashion to find the number of layers as well as the plane parameters and the set of true layer correspondences for other layers.
3: Use estimates from previous steps as initialization to refine the estimates on $\mathbf{p}$, $\mathbf{n}_l$, and $d_l|_{l=1}^{L-1}$ by solving the joint-optimization in Eq. 10.
4: Estimate occluded binary layer masks $\tilde{\boldsymbol{\alpha}}_{r,l}^i|_{l=1}^{L-1}$ by solving Eq. 12.
5: Estimate occluded factional layer masks $\tilde{\boldsymbol{\varrho}}_{r,l}^i|_{l=1}^{L-1}$ by solving Eq. 18 .
6: Estimate latent layer intensity images $\mathbf{f}_l|_{l=0}^{L-1}$ by solving Eq. 21.
7: Combine warped instances of $\tilde{\boldsymbol{\varrho}}_{r,l}^i|_{l=1}^{L-1}$ according to Eq. 22 to yield latent layer masks $\boldsymbol{\alpha}_l|_{l=1}^{L-1}$.
8: Find the rectified image $\mathbf{f}$ using Eq. 4.

key-rows were spaced equal distance apart and we used 4 key-rows/frame in our experiments. As we discussed in Section 3.1, the iterative identification of plane parameters is repeated until it is impossible to identify sufficient number of true correspondences for a unique layer, wherein the sufficient number was set to 5% of all available correspondences in our experiments. While solving Eq. 10 we use the outlier correspondence cost (*i.e.*, γ_o) as a fixed scalar value of 2. While solving Eq. 12, we use k = 0.4, μ_t as a threshold derived based on the statistics of the overall gradients in the image, and the labels β were assigned with values in the range [1, L+1]. Erosion and dilation operations used in fractional layer mask recovery are done typically by 5 pixels each. For the implementation of various optimization problems in our work we set the parameter values as $\lambda_s = 3$, $\lambda_1 = 2$, $\lambda_2 = 100$, $\lambda_3 = 5$, and $\lambda_{\mathbf{f}} = 0.1$. We have used Sobel operator to find the gradients in Eq. 14 and Eq. 21.

To compare the performance with [1, 5] we have used our own implementation of their algorithm (since the code was not available). For [4, 3] we have used the code provided by the authors, and for [6] we used the results sent by the authors upon request. For the generation of the dataset corresponding to synthetic experiments, we have manually created masks of different shapes to use as spatial support for FG layers. To simulate a scene with L layers, we randomly select L Images from [18] and $L-1$ masks to form latent layer intensity images and latent layer masks (an example is shown in the first row of Fig. S1). A random number generator was used to find the depth, and normal corresponding to each layer. Random third-degree polynomials were used to generate synthetic camera trajectories. We then followed our layered-image formation (as pointed out in Section S1) to generate the synthetic images.

Efficiency: Our unoptimized implementation in MATLAB on an Intel Core i5 CPU takes around 215 seconds to run the entire algorithm for a 3 layer scene with input images of resolution 700×400, wherein the individual units for camera motion estimation, occluded layer mask estimation using graph cut, fractional layer mask recovery using alpha matting, estimation of latent layer intensities, and final rectification consumes about 126, 42, 35, 11, and 0.6 seconds respectively.

S3. Overall Algorithm

The main steps involved in our rectification pipeline are listed in Algorithm 1. In Fig. S3 we show few representative intermediate results obtained during rectification of a synthetic example, which provides more clarity on the role of each step in our algorithm. The three middle frames among all the input images used in this experiment are shown in Figs. S3(a-c). The iterative refinement of the occluded layer mask corresponding to the reference frame in Fig. S3(b) is illustrated through Figs. S3(d-g). We repeat this process to find the occluded layer masks for all three input images. At the end of occluded layer mask estimation we have the binary mask associated with the visible pixels for all the layers in all three input images. These binary masks are used to generate the trimap, which is then provided as input to the fractional occluded layer mask estimation unit. The trimap corresponding to the foremost layer of the reference frame is shown in Fig. S3(i), Fig. S3(j) shows the background layer intensities at the uncertain regions of the trimap which we use for regularization of fractional mask estimation. Fig. S3(j) shows the recovered fractional layer mask.

The processing steps involved in the aggregation of masks and intensities from multiple images are shown in Figs. S3(l-r). Figs. S3(l-n) include the fractional mask estimates corresponding to the middle layer of the input images in Figs. S3(a-c), which are then re-warped to the latent image coordinates and combined to yield the full layer mask estimate of the middle layer (Fig. S3(o)). The latent layer intensities of all the layers obtained by solving Eq. 21 are shown in Fig. S3(i). It should be noted that, we can recover the background layer intensities only for those positions for which the pixels are visible in atleast one of the input images. Recovered latent image and the ground truth image are shown in Fig. S3(s) and Fig. S3(t), respectively. A care-

ful inspection around the layer boundaries of the recovered latent image in Fig. S3(t) reveals the presence of holes at some locations. These holes corresponds to the pixels in the latent coordinates that are not visible in *any* of the input images.

S4. Additional quantitative comparisons

To quantitatively compare the performance in terms of the camera motion parameters, we report (in Table S1) the values of root mean squared error of translations (E_t, in pixels), and average geodesic distance of rotations (E_r in degrees) obtained over the synthetic datasets S_1 and S_2.

Table S1. Motion error on datasets S_1 and S_2

	Dataset S_1		Dataset S_2	
Method	E_t	E_r	E_t	E_r
[1]	0.72	0.17	6.31	1.20
[5]	0.55	0.14	5.95	1.31
[4]	2.70	0.41	8.05	2.15
[3]	2.34	0.36	6.89	1.92
Ours	0.33	0.14	0.47	0.16

For quantitative comparison with [6], we have used 5 images from our synthetic data-set S_2 for which the the authors of [6] provide us with the *rectified images* and *motion estimates*. Comparisons for those 5 images are provided in Table S2.

Table S2. Quantitative evaluation on 5 images from dataset S_2

Method	Et	Er	PSNR
[6]	2.41	0.92	25.70
Ours	0.51	0.18	29.53

S5. Additional qualitative comparisons

In this section, we provide many more visual comparisons of our rectification algorithm using both synthetic and real examples. Comparisons for three synthetic examples are shown in Fig. S4, where the input RS images are generated to simulate scenes containing two (third row) and three layers (first and second row). While the camera motion for the example in the first row contains dominant translations, the second and third rows contain both camera translations and rotations. Our method integrates information from multiple images to fill in the holes in those regions where the information in the latent image is not visible in the reference input image. In contrast, existing methods do not account for the occlusion effects, and hence attempt to recover the latent image solely from a single RS image, leading to significant errors in terms of latent image restoration. Also, since existing methods do not account for depth and occlusion dependencies, the camera motion estimation itself goes wrong. Consequently, the RS distortions are prevalent in all the layers of the rectified images too. It can be observed that, in few cases, since the background layer is dominant, existing methods end up giving a camera motion estimate close to that of the background layer, resulting in partial rectification of background layer alone.

Similar observations can be found from our real examples in Fig. S5 too, wherein the first two columns correspond to two layer scenes and the third column comprises a three-layer scene. Interestingly, while attempting to rectify the examples in the first and third column of Fig. S5, the single image based methods ([4],[3]) introduces curvature distortions in the rectified images.

In Fig. S6, we have shown results of real examples corresponding to the same scene, but with different kinds of camera motion. As can be observed, the first two columns contain images containing RS distortions induced by dominant-translations, whereas the last two columns contain images captured using dominant in-plane rotations (resulting in curvature distortion in the captured images). For the cases of camera-motion involving dominant translations, existing multi-image based methods ([1, 5]) have partially captured the background motion, whereas the single-image based method in [4] induces undesired curvature distortions. In the absence of depth-dependent distortions (third column of Fig. S6), multi-image based methods deliver accurate rectification results. While the single image based method in [4] fails to rectify, [3] retains minor distortion in the rectified image. In all the examples, our proposed approach is able to deliver accurate rectification results.

Visual comparisons for the case of camera motion involving dominant rotations are provided in Fig. S7. All input images used in Fig. S7 were taken from the dataset provided by [6]. Examples in Fig. S7 indicate that when the depth-dependent RS distortions are absent, multi-image based methods (including ours) are able to deliver accurate results, whereas the performance of single image based methods depends on the nature of the scene. While both single image based methods ([4],[3]) perform well when there exist sufficient number of straight lines in the scene (Fig. S5, first and third row), their rectification fails completely for other scenarios.

To conclude, the performance of existing single image based methods depends on the nature of the scene, since they are designed to work well only for specific classes such as images of urban scenes containing many straight lines ([4, 3]), or face images ([3]) etc. Existing multi-image based methods ([1, 5]) are quite successful in doing rectification for the cases involving no depth-dependent distortions. While none of these methods is successful in removing depth-dependent distortions, our proposed approach can handle the cases of both depth-dependent RS distortions as well as occlusion effects.

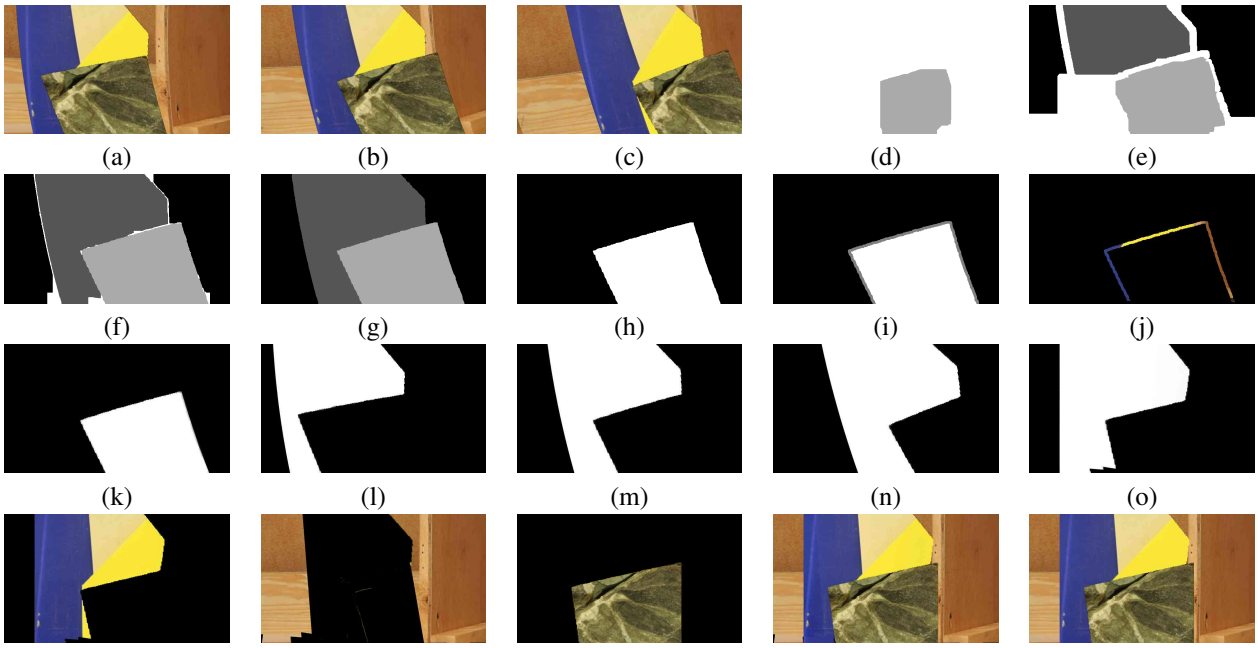

Figure S3. Detailed illustration of all the main steps involved in our RS rectification method. Synthetic example of a three layer scene. (a-c) Input images. Occluded layer mask recovery (for the reference image in (b)) through iterative graph cut: Label assignment after (d) 1^{st} iteration, (e) 3^{rd} iteration, (f) 6^{th} iteration, and (g) 11^{th} iteration. Recovering fractional layer mask: (h) Binary layer mask (of the fore-most layer) obtained through graph cut, (i) trimap with non-binary value at confusion region, (j) background layer intensity obtained from neighboring frames for regularization of alpha-matte estimation, and (k) recovered fractional layer mask. Results on the full layer mask recovery of the middle layer: (l-n) Estimated occluded fractional layer masks for RS images in (a-c) and (o) recovered latent full layer mask. Latent layer intensity recovery results for all the layers: (p) middle layer, (q) background layer, and (r) foreground layer. (s) Final rectified image. (t) Ground truth image.

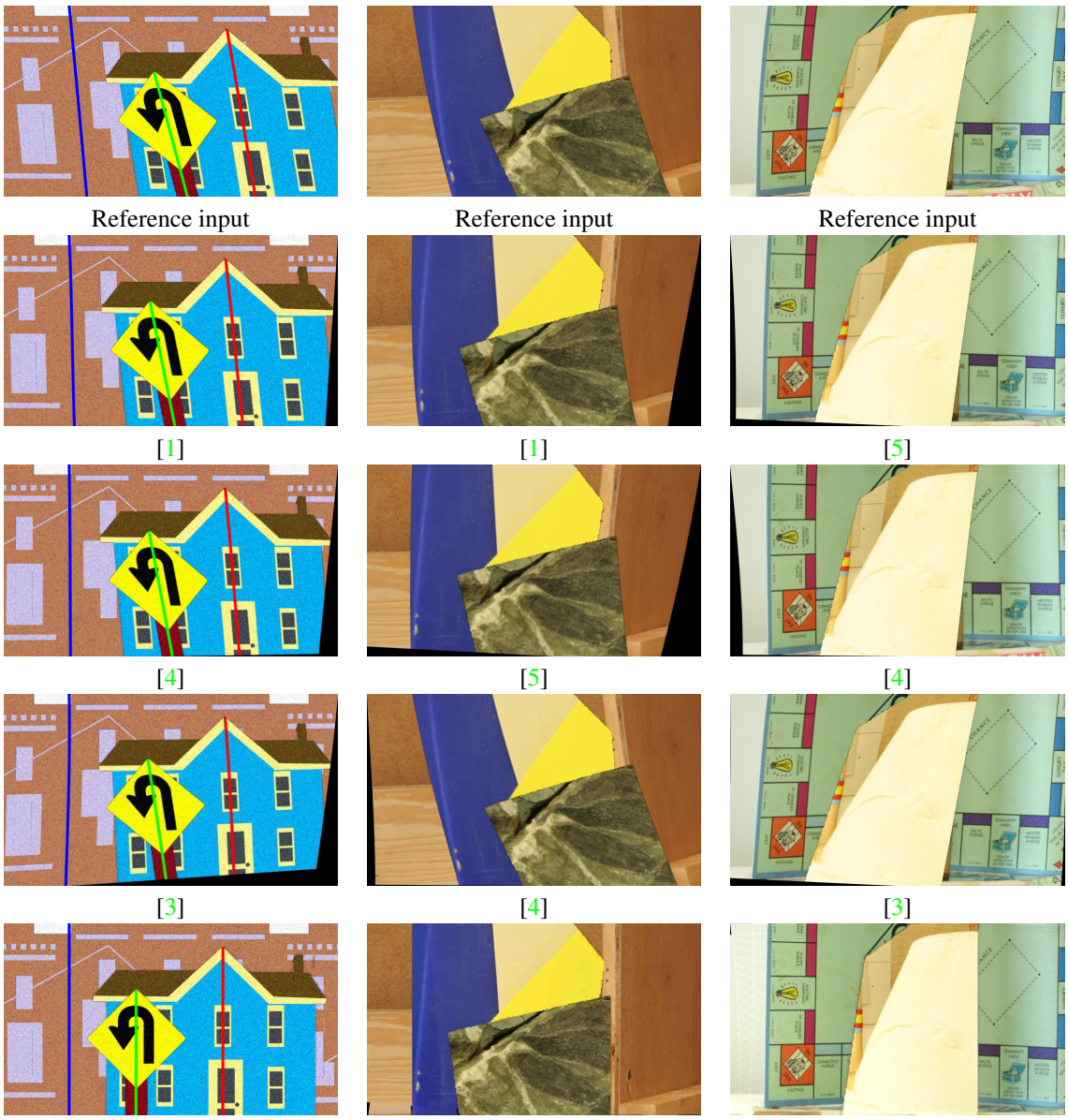

Figure S4. Synthetic examples: scenes with three (first and second column) and two layers (third column).

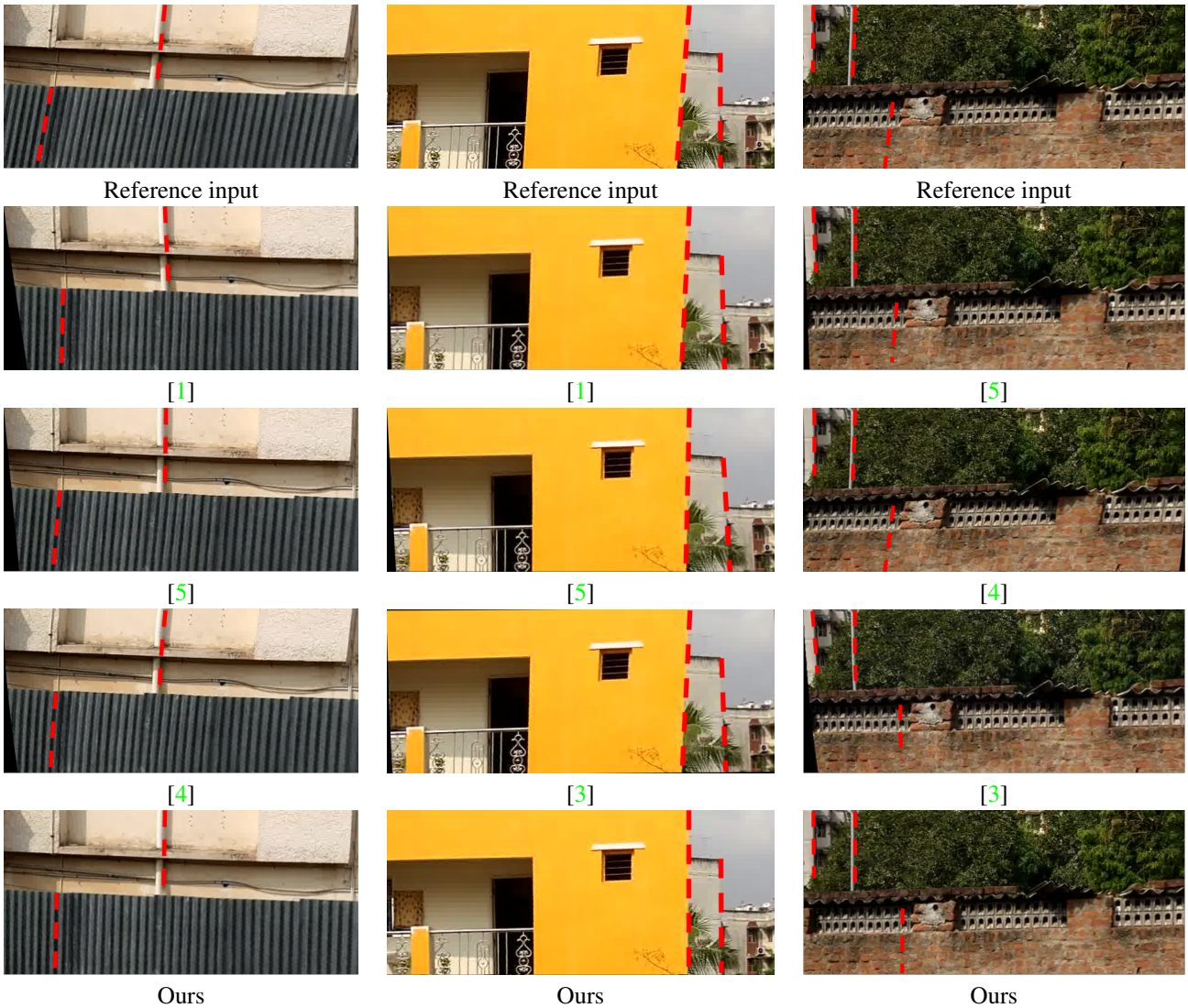

Figure S5. Real examples for scenes containing two (first and second column) and three layers (third column).

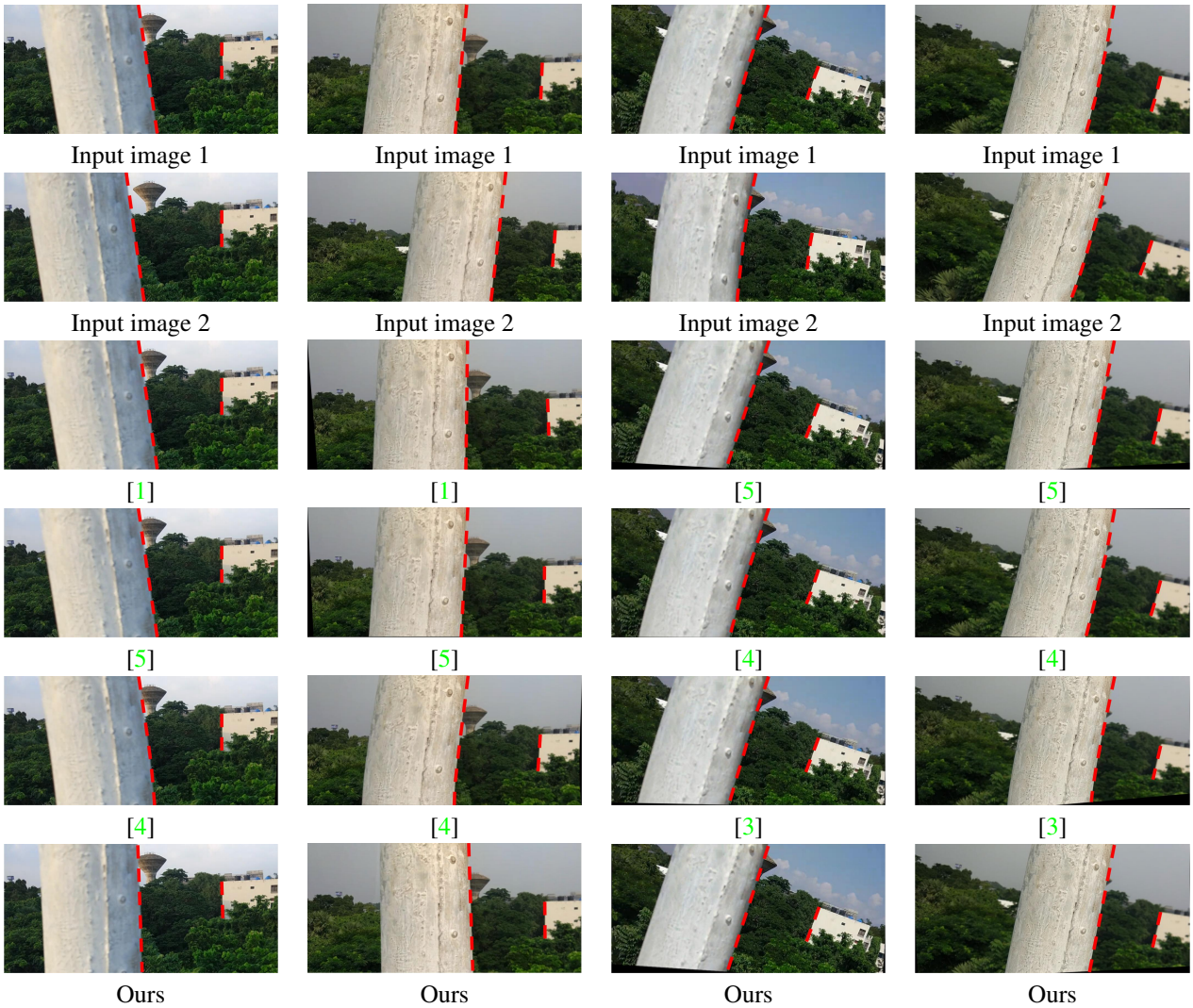

Figure S6. Real examples of a scene containing two layers, for different kinds of camera motion. Camera motion involving dominant translations from left to right (first column), dominant translations from right to left (second column), dominant in-plane rotations (*i.e.* no depth-dependent distortions) in anti-clockwise direction (third column), and dominant in-plane rotations (*i.e.* no depth-dependent distortions) in the clockwise direction (fourth column). For all the examples shown above, the input image shown in the first row is used as the reference frame.

Figure S7. Real examples from [6] with camera motion involving dominant rotations (*i.e.* no depth dependent distortions).

References

[1] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Calibration-free rolling shutter removal. In *Computational Photography (ICCP), 2012 IEEE International Conference on*, pages 1–8. IEEE, 2012. 4, 5, 7, 8, 9

[2] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):228–242, 2008. 3

[3] V. Rengarajan, Y. Balaji, and A. N. Rajagopalan. Unrolling the shutter: Cnn to correct motion distortions. In *CVPR*, July 2017. 4, 5, 7, 8, 9, 10

[4] V. Rengarajan, A. N. Rajagopalan, and R. Aravind. From bows to arrows: Rolling shutter rectification of urban scenes. In *CVPR*, June 2016. 4, 5, 7, 8, 9, 10

[5] E. Ringaby and P.-E. Forssén. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012. 4, 5, 7, 8, 9

[6] B. Zhuang, L.-F. Cheong, and G. Hee Lee. Rolling-shutter-aware differential sfm and image rectification. In *ICCV*, Oct 2017. 4, 5, 10