Rolling Shutter Super-Resolution

Abhijith Punnappurath, Vijay Rengarajan, and Rajagopalan A.N.

Department of Electrical Engineering, Indian Institute of Technology Madras

This is a draft version of the paper accepted for presentation at International Conference on Computer Vision, December 2015.

Rolling Shutter Super-Resolution
SUPPLEMENTARY MATERIAL

Abhijith Punnappurath, Vijay Rengarajan, and Rajagopalan A.N.
Department of Electrical Engineering
Indian Institute of Technology Madras, Chennai, India
jithuthatswho@gmail.com, {vijay.ap,raju}@ee.iitm.ac.in

In this supplementary material, we provide additional real results to further demonstrate the effectiveness of our proposed RS-SR scheme. We also verify experimentally the correctness of our implementation of the RS rectification method in [15] [1].

S1. Real results

We show two more super-resolution results (for an SR factor of two) for images captured using a hand-held RS camera in Figs. S1 and S3. We follow the same notation (a-h) for the figures as in the main paper. Zoomed-in patches from Figs. S1 and S3 are shown in Figs. S2 and S4, respectively. The RS effect is clearly visible in the red LR patches. Observe that in both examples, the text is clearly readable in our green SR output patch (h+).

S1.1. Implementation details

For all our real experiments, including the ones in the main paper, we selected a large coarse 5D search space around the middle row for the first iteration of our AM algorithm as $[-15 : 3 : 15]$ pixels for in-plane translations, $[-3^\circ : 1^\circ : 3^\circ]$ for in-plane rotations, $[-1^\circ : 0.5^\circ : 1^\circ]$ for out-of-plane rotations about the vertical axis, and $[0.9 : 0.1 : 1.1]$ scaling for out-of-plane translations. Such a coarse binning of the pose space is sufficient to initialize the algorithm because this estimate is refined with iterations. For other rows, we chose a smaller and finer pose space around the centroid pose of the middle row as $[-3 : 1 : 3]$ pixels for in-plane translations, $[-1^\circ : 0.5^\circ : 1^\circ]$ for in-plane rotations, $[-0.5^\circ : 0.25^\circ : 0.5^\circ]$ for out-of-plane rotations about the vertical axis, and $[0.95 : 0.05 : 1.05]$ scaling for out-of-plane translations. For subsequent iterations, this smaller search space was selected around the centroid pose of the previous iteration for each row. We used a fixed value of 2500 for α in all our examples. The value of λ was set to 100 such that a highly sparse set of camera poses are selected from the search space. The centroid pose which corresponds to the actual motion is estimated using these selected poses.

[1] All reference numbers are with respect to our main paper.

S2. Our implementation of [15]

The technique in [15] was used to rectify the RS affected LR frames before providing them as input to classical GS-based SR algorithms. [15] is a state-of-the-art RS video rectification technique that is designed to handle any general motion of the camera. The scene is presumed to be planar. However, the major drawback of this algorithm is that only piecewise linear motions can be recovered. Our method, on the other hand, does not suffer from any such limitation and is capable of accurately estimating per-row motion even for complicated non-linear trajectories (see the last two plots in Fig. 4 of our main paper). Moreover, [15] being a feature-based approach is prone to errors in motion computation if sufficient number of features are not detected in certain rows of the image.

To verify the correctness of our implementation of [15], we perform the following experiment. The reference image which is free from RS effect is shown in Fig. S5(a) on which we apply uniform in-plane translatory motion to obtain the distorted image in Fig. S5(b). The rectified output of [15] is shown in Fig. S5(c). The ground truth and estimated camera motions are shown in Fig. S5(d). It can be seen that the method performs quite well – the estimated camera path closely follows the ground truth trajectory, and the rectified image in Fig. S5(c) is free from RS effect. This confirms that our implementation is correct. An example with non-linear motion (which violates the assumption in [15]) is shown in Fig. S6(a). This RS image was obtained by distorting the rows of Fig. S5(a) using the ground truth camera path shown in Fig. S6(c). In this case, it can be observed that the computed trajectory (see Fig. S6(c)) is erroneous, and there are artifacts in the rectified image (see Fig. S6(b)). This is the reason why, in our comparisons, classical GS algorithms such as [17] and [18] do not perform very well despite the pre-rectification step.

Figure S1. Real example: *Building 1*.

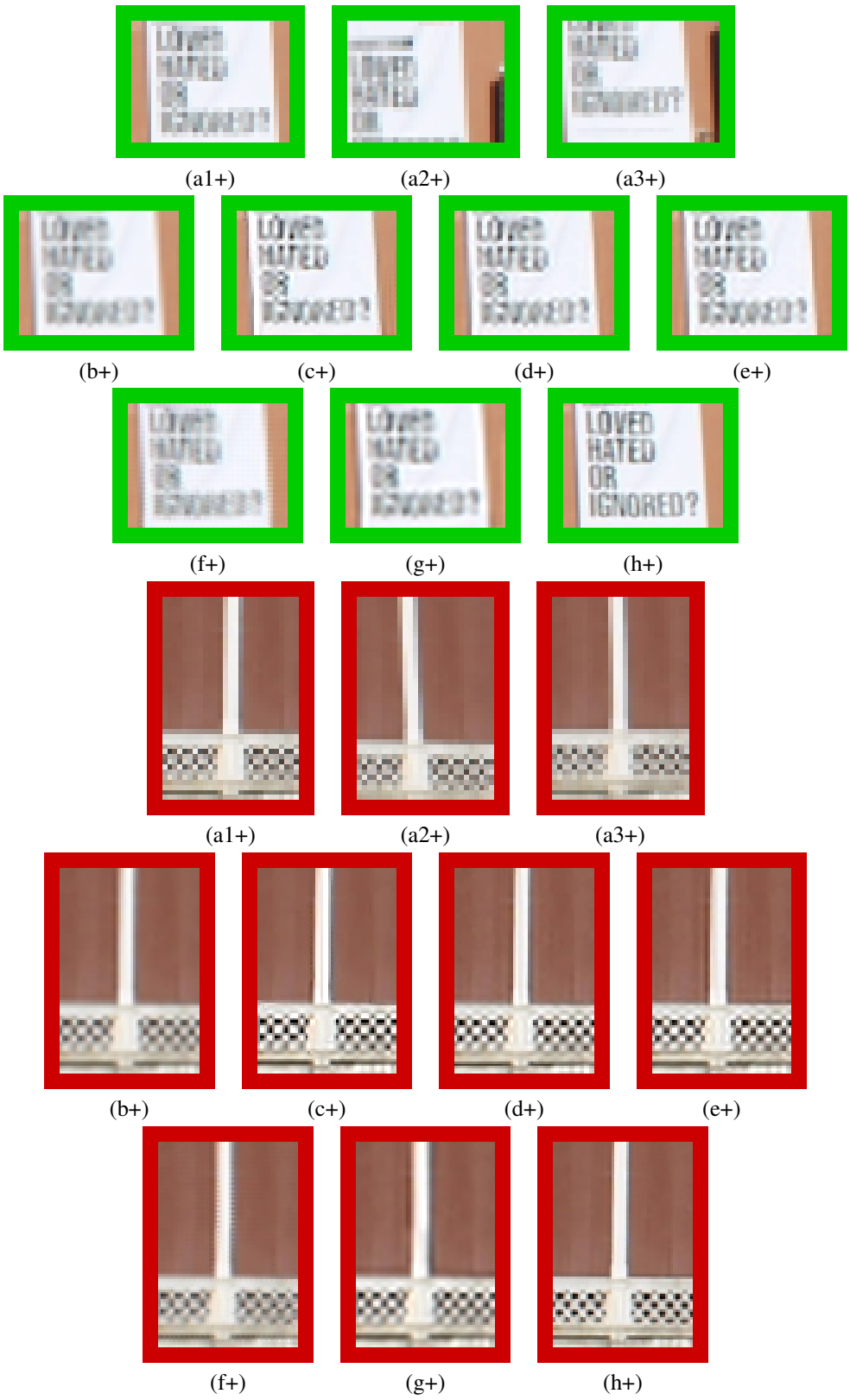


Figure S2. Zoomed-in patches from Fig. S1.

Figure S3. Real example: *Building 2*.

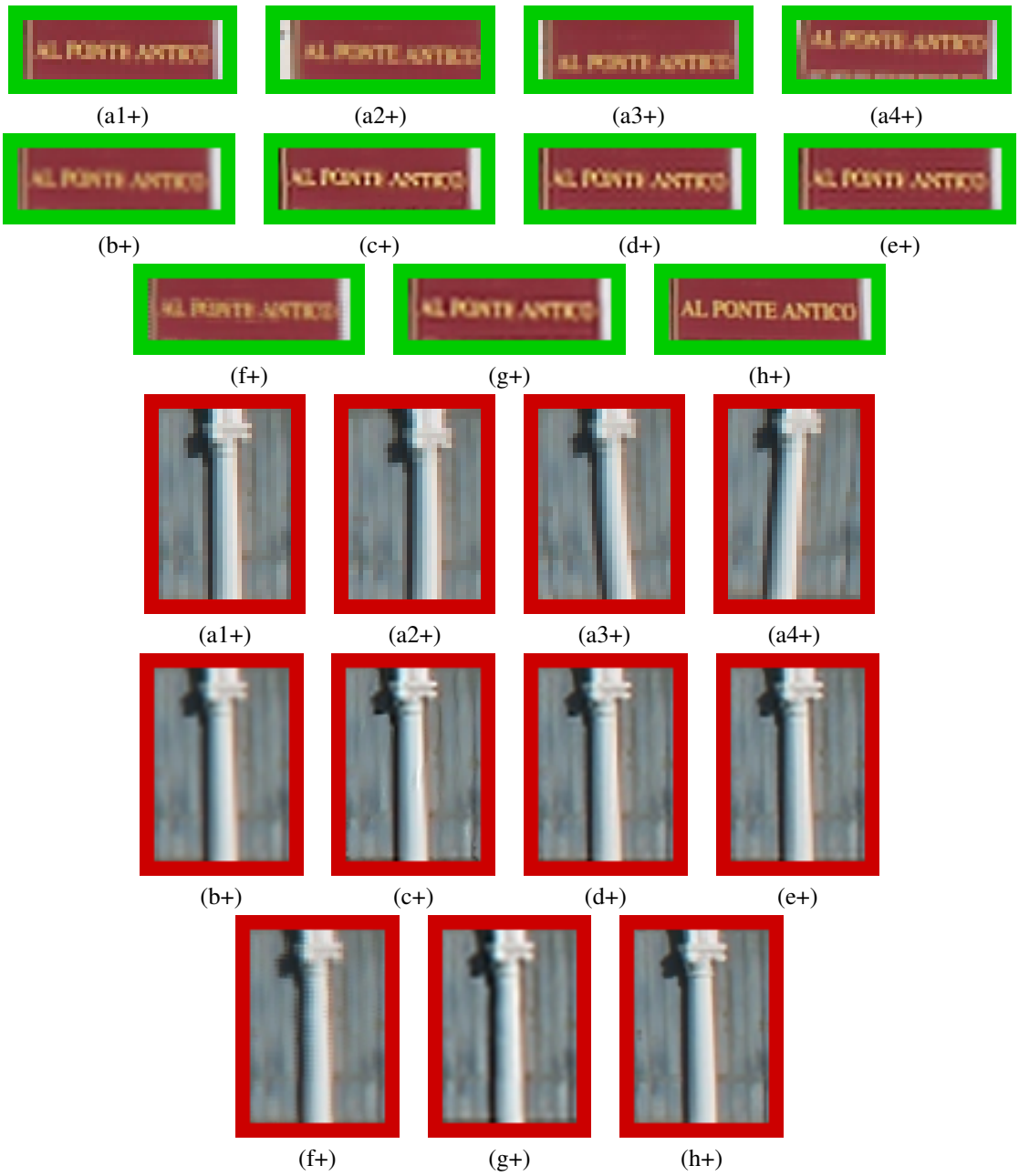


Figure S4. Zoomed-in patches from Fig. S3.

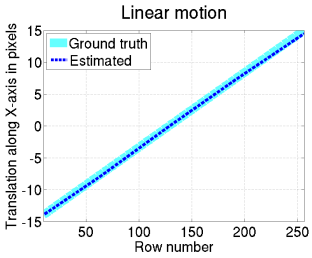
S3. Implementation of [7]

We have compared our result with the single-image SR technique in [7]. Since the authors of [7] have not made their code publicly available, we used the code downloaded from `http://web.stanford.edu/class/ee368/Project_11/` for comparison. This is an implementation of the method in [7] by Arora and Kolte. The details can be found in their report `https://stacks.stanford.edu/file/druid:my512gb2187/Kolte_Arora_Image_Superresolution.pdf` where the authors have claimed that their implementation produces results identical to those reported in [7].

(a) Reference image.

(b) RS image.

(c) Rectified using [15].

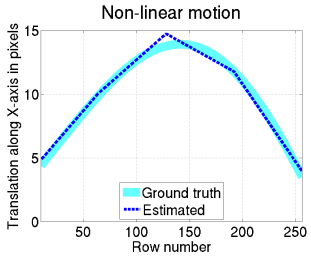


(d) Camera trajectory.

Figure S5. Rectifying linear motion using the method in [15].

(a) RS image.

(b) Rectified using [15].



(c) Camera trajectory.

Figure S6. Rectifying non-linear motion using the method in [15].