



Published in Image Processing On Line on 2022-11-00.
 Submitted on 2022-10-10, accepted on 2022-10-10.
 ISSN 2105-1232 © 2022 IPOL & the authors CC-BY-NC-SA
 This article is available online with supplementary materials,
 software, datasets and online demo at
<https://doi.org/10.5201/ipol.2022.436>

Fixed Pattern Noise Reduction: Temporal High Pass Filter

Arnaud Barral

Université Paris-Saclay, ENS Paris-Saclay, Centre Borelli, Gif-sur-Yvette, France
arnaud.barral@ens-paris-saclay.fr

Communicated by Jean-Michel Morel *Demo edited by* Arnaud Barral

Abstract

Temporal high pass filter methods are a family of methods for Fixed Pattern Noise (FPN) reduction. They are recursive real time methods that apply a high-pass temporal filter to remove the FPN. FPN is a temporally coherent noise present on video due to the non-uniformity response of the sensors. It is a common problem for infrared videos and can degrade the quality of the observation. In this work we will study and compare three classical temporal high pass filter FPNR methods.

Source Code

The reviewed source code and documentation for this algorithm are available from [the web page of this article](#)¹. Usage instructions are included in the `README.txt` file of the archive.

Keywords: denoising; FPN

1 Introduction

The goal of video denoising is to recover a clean video from a noisy video. Noise can be caused by several reasons. FPN for (Fixed Pattern Noise) is a very specific kind of noise that is mostly present in infrared videos where the ratio of signal over noise is low and is generally caused by the pixel-to-pixel non-uniformity response.

FPN is called fixed pattern noise because the noise is the same for every frame in the video. In practice, the FPN is not completely fixed and changes slowly over time but can be considered constant on a reduced time period. A linear model is widely used to describe noisy images with FPN. The linear model is defined as

$$y(n) = a \otimes x(n) + b \quad (1)$$

¹<https://doi.org/10.5201/ipol.2022.436>

where \otimes is the element-wise product, $x(n)$ is the clean frame at time n , $y(n)$ the observed frame at time n , a and b are respectively the FPN gain and offset coefficients. Equation (1) can be rewritten pixel-wise as

$$y(n)_i = a_i x(n)_i + b_i, \quad (2)$$

where i is the corresponding pixel

In general, FPN reduction (FPNR) methods try to estimate correction coefficients: $G(n)$ the gain correction coefficient and $O(n)$ the offset correction coefficient in a recursive framework. Thus, if we note $\hat{x}(n)$ the denoised image at time n ,

$$\hat{x}(n) = G(n) \otimes y(n) + O(n). \quad (3)$$

The field that tries to solve this problem is called FPNR (FPN reduction) or NUC (Non-Uniformity Correction). FPNR methods can be divided into two main families: reference-based (RB-NUC or RB-FPNR) and scene-based (SB-NUC or SB-FPNR). The reference-based methods remove noise according to fixed calibration parameters, estimated for example with a shutter. However, the FPN varies slightly over time, which requires an update of the parameters. Because of this, most of the research is focused on the other family of methods, which is based on the image itself.

In this family of SB-FPNR methods, we can distinguish several subfamilies: those that work from image statistics, temporal high pass filters, registration methods and optimization methods. Recent methods mainly use CNNs (convolutional neural networks) which take infrared images or sometimes natural gray scale images without degradation, to which artificial noise is added to simulate the FPN in order to create noisy and non-noisy training pairs to train the neural networks. These can be placed in a new category that could be called learning methods.

In this work, we will study three classical temporal high pass filter FPNR methods: plain high-pass temporal filtering [5], high-pass temporal filtering after low-pass spatial filtering [3], and high-pass temporal filtering after bilateral spatial filtering [7]. We provide an implementation, made by us, for these methods and an online demo where they can be tested.

2 Temporal High Pass Filter Methods

2.1 General Formulation

Temporal High Pass Filter (THPF) is a family of scene based methods as explained above. They work directly with the images and try to estimate correction coefficients in a recursive framework.

The main assumption considered is that temporal high-frequency information belongs to the scene, while temporal low-frequency information belongs to fixed pattern noise [5]. And so by performing a temporal high pass filtering, the noise can be removed. The second main assumption, added by another work [3], is that noise corresponds generally to spatial high frequency information whereas the content of the image has low spatial frequency and so the noise can be estimated with a spatial high pass filter.

THPF methods generally do not consider the gain parameter and only remove the additive noise and update its estimation in a linear combination of: $y(n)$ current noisy frame, $F_{HS}(n)$ spatially filtered noisy image (that contains high spatial frequency of the noisy frame $y(n)$), $f(n-1)$ estimation of the previous step.

$$f(n) = \left(1 - \frac{1}{M}\right) f(n-1) + \frac{1}{M} F_{HS}(n), \quad (4)$$

where M is a parameter and $f(n)$ the estimated FPN at time n .

The denoised image \hat{x} at time n is then

$$\hat{x} = y(n) - f(n). \quad (5)$$

Since f is estimated recursively, we can also rewrite it as

$$f(n) = \sum_{k=0}^n F_{HS}(k) h_n(k) \quad (6)$$

with

$$\forall 0 \leq k \leq n, h_n(k) = \frac{1}{M} \times \left(1 - \frac{1}{M}\right)^{n-k} \quad (7)$$

where k is the number of considered frames and n is the total number of frames.

Figure 1 shows the value of the coefficient $h_n(k)$ with respect to the values of k (the considered number of frames), n (the total number of frames) and the parameter M used in the experiments below. The value of sum, shown in the legends of the figure, is the sum of the coefficients $h_n(k)$ over k .

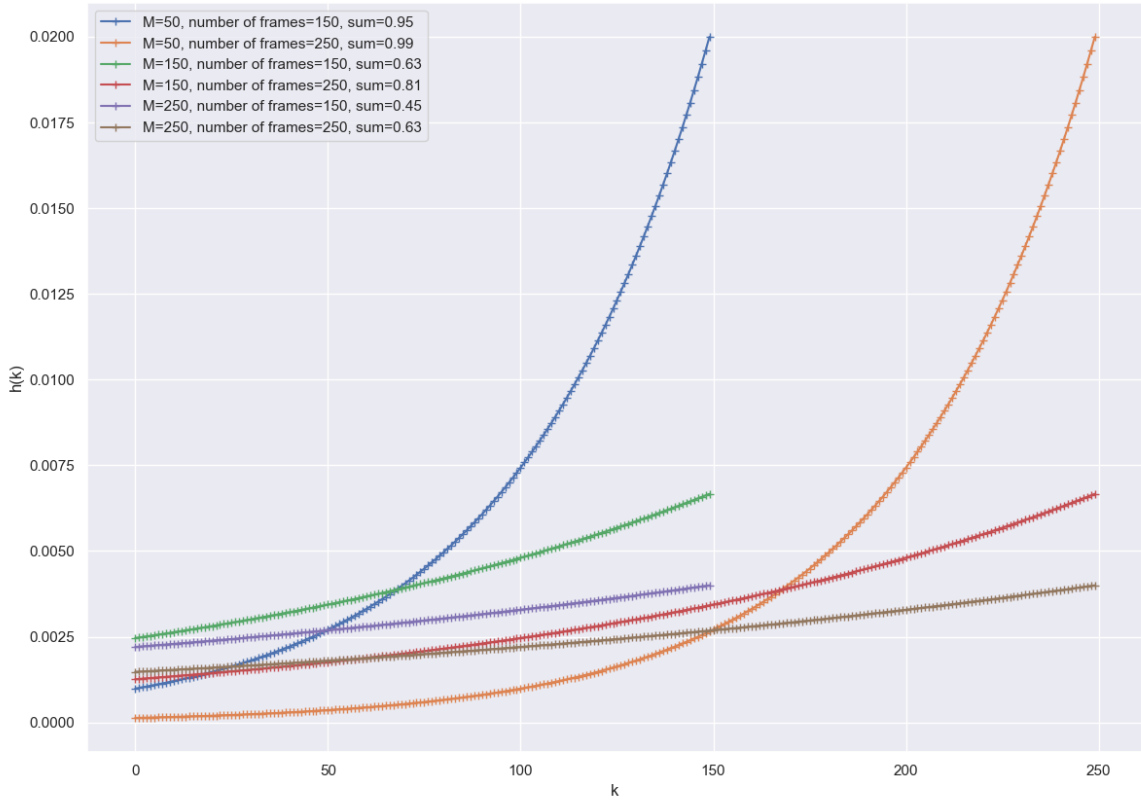


Figure 1: Weights of the temporal high pass filter as presented in Equation (7).

2.2 Discussion on the Filter

As explained above, THPF methods use a spatial filter and the main differences between them are the choice of the filter used.

No filter [5] The first paper that was published did not use any spatial filter [5], the estimated additive noise was just a recursive average of the frames. This work introduced the first main assumption.

Average filter [3] The next paper, the SLTH (spatial low temporally high) THPF method, introduced the use of spatial filtering. The authors used an average filter [3], and it has become a standard in this family of methods to use a filter since then [2, 7]. The idea as explained above is that the LSF (low spatial frequency) of the frames has the real content of the frames, while the HSF (high spatial frequency) is mainly noise.

$$f(n) = \frac{1}{M} F_{HS}(n) + \left(1 - \frac{1}{M}\right) f(n-1), \quad (8)$$

$$y^{HSF}(n) = y(n) - y^{LSF} = y(n) - y(n) \otimes A, \quad (9)$$

$$F_{HS}(n)_{i,j} = \begin{cases} y_{i,j}^{HSF}(n) & \text{if } |y_{i,j}^{HSF}(n)| < Th, \\ 0 & \text{else.} \end{cases} \quad (10)$$

where A is a spatial low-pass filter (in the original paper they used a 10×10 average filter) and Th is a threshold parameter to set.

More recent approaches proposed to replace the above filter by a bilateral filter [7] or a guided filter [2], which do not use a threshold.

3 Implementation

3.1 Pseudo Code

The pseudo code is given in Algorithm 1.

Algorithm 1: THPF

```

1 function FPNR_THPF
  Input noisy_images: A list of noisy images
  Output denoised_images: A list of denoised images
2   $N \leftarrow \text{return\_number\_of\_images}(\text{noisy\_images})$ 
3   $H, W, C \leftarrow \text{return\_shape}(\text{noisy\_images}[0])$ 
4   $f[0] \leftarrow \text{zeros}(H, W)$ 
5  for  $n$  from 1 to  $N$  do
6     $F_{HS}[n] \leftarrow \text{filter}(\text{noisy\_images}[n])$ 
7     $f[n] = \left(1 - \frac{1}{M}\right) f[n-1] + \frac{1}{M} F_{HS}[n]$ 
8     $\text{denoised\_images}[n] = \text{noisy\_images}[n] - f[n]$ 
9  return denoised_images

```

We implemented the average filter and for the bilateral filter we used the implementation from scikit-image [6].

3.2 Noise Modelling

The noisy image that contains FPN can be approximated using the following widely used linear model

$$\forall n \in \mathbb{N}, y(n) = a \otimes x(n) + b. \quad (11)$$

Another important part in our noise modelling is the spatial structure of the noise. FPN can also be spatially coherent. In the options of the demo, the user can choose to add a structured and an unstructured component for both the additive and the multiplicative part of the noise. FPN can have both a structured component and an unstructured component, it is important to consider them both since the denoising algorithm can have different behaviors depending on the structure of the noise. For the structured component, we used row and column noise. And so

$$a = a_w + a_r + a_c, \quad b = b_w + b_r + b_c, \quad (12)$$

where

$$\forall i, i', j \quad b_r(i, j) = b_r(i, j'), \quad b_c(i, j) = b_c(i', j). \quad (13)$$

So b_w, b_r, b_c are respectively the unstructured, row and column component of the noise, b_w is spatially independent and b_w, b_r, b_c are mutually independent, the same for a_w, a_r, a_c . We can model b_w, b_r, b_c

$$b_w \sim \mathcal{N}(0, \sigma_b), \quad b_r(\cdot, j) \sim \mathcal{N}(0, \sigma_b), \quad b_c(i, \cdot) \sim \mathcal{N}(0, \sigma_b), \quad (14)$$

and a_w, a_r, a_c as

$$a_w \sim \mathcal{N}(1, \sigma_a), \quad a_r(\cdot, j) \sim \mathcal{N}(1, \sigma_a), \quad a_c(i, \cdot) \sim \mathcal{N}(1, \sigma_a). \quad (15)$$

In the demo, we can select the level of noise we want to add to the frames σ_a, σ_b .

4 Influence of the Parameters on the Performance of the Algorithm

M , weight for recursive average. The weight for the moving average (which is $\frac{1}{M}$) is similar to a step size in an optimization process for minimizing an energy. If it is too big, the estimated noise will never converge to the actual noise. On the other hand, if it is too small, the convergence will be very slow. The default value for this parameter is generally the number of frames considered [5, 3, 7]. In this work, we found that in that case the convergence can be very slow, and using a lower value almost always produces better results (see Tables 3 and 4).

Size of the filter. This parameter is common for all the methods that use a spatial filter [3], [7] but the method without filter obviously does not use it. The default value is set to 10. Too high values tend to over-denoise and so over-smooth the video, and can also be much more time-consuming and resource intensive.

Choosing the Filter

No filter. Not choosing a filter does not require any extra parameters. It produces by far the worst results compared to other methods in terms of quantitative results (see Tables 1 and 2). Noise is almost never removed, and when it is, it is at the expense of much of the image content (see Figures 2 and 3).

Average filter [3]. The average filter requires another parameter to set, the threshold Th . The threshold parameter determines how much we want to denoise. The default value is set to 255. High values provide better quantitative results but also remove high spatial frequency from the image, especially if the motion between frames is rather slow (see the logo in Figure 2). In the case where

the scene is completely static, the method blurs the images (see Figure 4). The parameter should be set according to the noise level and the motion between frames.

Overall, the method produced better results in terms of PSNR. The visual results are also better; much less content of the images is removed.

Bilateral filter [7]. The bilateral filter requires two more parameters to set, the color sigma and the spatial sigma of the filter, that are fixed to the same value in the demo.. The default value for both parameters is set to 45. Higher values can produce blurry results (see Figure 4) and sometimes denoise better the images if the noise is important.

5 Results

5.1 Datasets

To test the different approaches, we used infrared datasets [1, 4], and visible datasets from the Derf’s Test Media collection².

- The bus dataset from Derf contains 150 visible images and has some motion.
- The flower dataset from Derf contains 250 visible images and has little motion.
- The 8_selma dataset from LTIR [1] contains 235 infrared images and the camera is fixed with some people walking.
- The 640_ataset [4] is a dataset of 1000 infrared images. We only use 150 images from the training dataset. Images have nothing in common.

5.2 Metrics

The metrics generally used to measure the FPNR methods performance are the peak signal-to-noise ratio (PSNR) and the roughness index ρ (RI) defined by

$$PSNR = 20 \log \left(\frac{\max_value}{RMSE} \right), \quad (16)$$

where $RMSE$ is the root-mean-square error and \max_value is the maximum value for the data, and

$$\rho = \frac{\|h_1 * x\|_1 + \|h_2 * x\|_1}{\|x\|_1}, \quad (17)$$

where $h_1 = (1 \ -1)$ and $h_2 = h_1^T$.

For the quantitative results, we report the PSNR and RI of the last frame of each dataset and for the visual results, the last frame of each dataset.

²<https://media.xiph.org/video/derf/>

5.3 Quantitative and Qualitative Results

In terms of PSNR, the THPF with no spatial filter got the worst results; in the best case, there was an increase of .08 dB (Tables 1 and 2). The RI obtained is also almost equal to the ones of the noisy frames. Since there is no spatial filter, the estimated FPN also contains the content of the images, so either we don't remove any noise from the images or we remove the content of the images.

The bilateral THPF obtained its best results for classic configurations for THPF FPNR algorithms, a scene with some motion, for example on Figure 2, where the result is sharper than for the SLTH algorithm. On the other hand, the SLTH algorithm gave the best results for more "extreme" scenarios, a fixed scene and images having nothing in common. Looking at the visual results, we think this is because of the spatially structured component of the noise that the bilateral filter has difficulty in denoising, whereas the SLTH algorithm does not have this drawback.

All methods still produce blurry results on the fixed scene (see Figure 4). In that case, it is very hard for algorithms to distinguish between noise and actual content of the image since both are fixed. Methods still achieve reasonable PSNR by blurring the images.

dataset	PSNR			
	<i>Noisy</i>	<i>THPF</i>	<i>SLTH THPF</i>	<i>Bilateral THPF</i>
bus Derf	20.100	20.125	26.501	27.897
flower Derf	20.531	20.550	25.071	26.495
8_selma LTIR	19.868	19.947	28.647	28.064
640_ataset	20.105	20.180	32.153	29.599

Table 1: Quantitative PSNR results obtained with a simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$. The best results per dataset are highlighted.

dataset	RI				
	<i>Clean</i>	<i>Noisy</i>	<i>THPF</i>	<i>SLTH THPF</i>	<i>Bilateral THPF</i>
bus Derf	0.114	0.242	0.246	0.121	0.121
flower Derf	0.0798	0.120	0.120	0.0869	0.0851
8_selma LTIR	0.0540	0.396	0.395	0.0459	0.0520
640_ataset	0.0555	0.414	0.413	0.0762	0.0881

Table 2: Quantitative RI results obtained with a simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$. The best results per dataset are highlighted.

parameters			PSNR			
size	M	σ	<i>bus derf</i>	<i>flower derf</i>	<i>8_selma ltir</i>	<i>640_ataset</i>
5	50	45	26.887	25.929	26.311	26.333
10	50	45	27.897	26.495	27.375	27.454
15	50	45	27.784	26.367	27.634	27.971
10	150	45	25.561	25.842	25.701	24.530
10	250	45	23.814	24.708	24.144	23.072
10	50	10	23.866	23.647	20.782	23.939
10	50	150	26.111	23.650	28.064	29.599

Table 3: Parameters comparison obtained with the bilateral THPF [7] and a simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$. The best results per dataset are highlighted.

parameters			PSNR			
size	M	Thres	<i>bus derf</i>	<i>flower derf</i>	<i>8_selma ltir</i>	<i>640_ataset</i>
5	50	255	26.501	24.780	27.594	28.172
10	50	255	26.033	24.052	28.647	30.875
15	50	255	24.507	23.610	28.359	32.153
10	150	255	25.622	24.664	28.116	27.040
10	250	255	24.056	24.321	26.160	24.603
10	50	200	26.033	24.053	28.647	30.875
10	50	100	26.005	25.071	28.621	30.837

Table 4: Parameters comparison obtained with the SLTH THPF algorithm [3] and a simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$. The best results per dataset are highlighted.



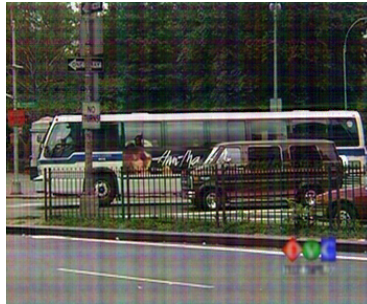
(a) ground truth



(b) noisy



(c) THPF



(d) SLTH THPF

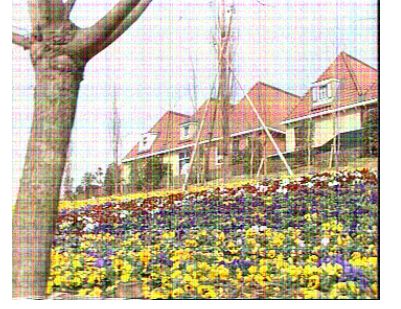


(e) Bilateral THPF

Figure 2: Comparison of the different methods on the last image of the bus dataset that contains 150 frames and some motion. Simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$ was added to the frames. (a), (b), (c), (d), (e) are respectively the ground truth clean image, the noisy image with simulated FPN, the image denoised by the original THPF method [5], the image denoised by the original SLTH THPF method [3], the image denoised by the bilateral THPF method [7]. The image (a) is rather dark, this is not a mistake, but the behavior of [5] that will remove content of the image while denoising.



(a) ground truth



(b) noisy



(c) THPF



(d) SLTH THPF



(e) Bilateral THPF

Figure 3: Comparison of the different methods on the last image of the flower dataset that contains 250 frames and little motion. Simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$ was added to the frames. (a), (b), (c), (d), (e) are respectively the ground truth clean image, the noisy image with simulated FPN, the image denoised by the original THPF method [5], the image denoised by the original SLTH THPF method [3], the image denoised by the bilateral THPF method [7].



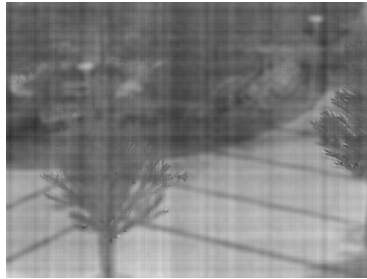
(a) ground truth



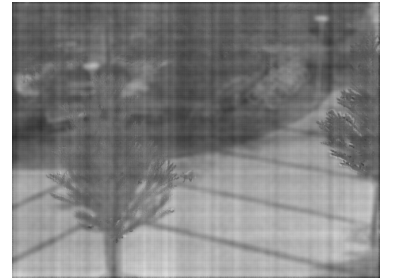
(b) noisy



(c) THPF



(d) SLTH THPF

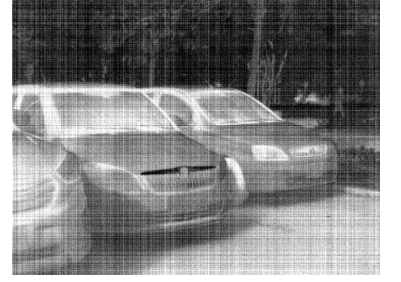


(e) Bilateral THPF

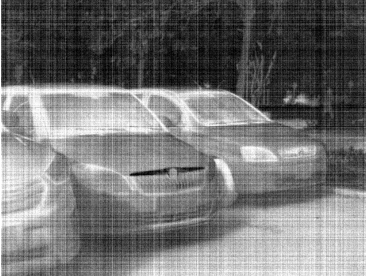
Figure 4: Comparison of the different methods on the last image of the 8_selma dataset that contains 235 frames and no motion, it is a fixed scene. Simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$ was added to the frames. (a), (b), (c), (d), (e) are respectively the ground truth clean image, the noisy image with simulated FPN, the image denoised by the original THPF method [5], the image denoised by the original SLTH THPF method [3], the image denoised by the bilateral THPF method.



(a) ground truth



(b) noisy



(c) THPF



(d) SLTH THPF



(e) Bilateral THPF

Figure 5: Comparison of the different methods on the last image of a sample of the 640_ataset dataset. The sample contains 150 images that have nothing in common. Simulated additive FPN (no temporal independent noise or multiplicative FPN), spatially structured and spatially independent with a standard deviation of $\sigma_b = 15$ was added to the frames. (a), (b), (c), (d), (e) are respectively the ground truth clean image, the noisy image with simulated FPN, the image denoised by the original THPF method [5], the image denoised by the original SLTH THPF method [3], the image denoised by the bilateral THPF method.

6 Conclusion

We tested several temporal high pass filtering methods for fixed pattern noise reduction, provided insights about their parameters and demonstrated the limitations and advantages of this family of methods.

Acknowledgment

I would like to thank the organizers of the MLBriefs workshop and the Centre Borelli who gave me the opportunity to work on this article and on this demo. I would also like to thank my supervisors for their interesting feedback about this article and the demo. This work was partially funded by HGH Infrared System.

References

- [1] A. BERG, J. AHLBERG, AND M. FELSBERG, *A thermal object tracking benchmark*, in IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2015. <https://doi.org/10.1109/AVSS.2015.7301772>.
- [2] K. CHENG, H-X. ZHOU, S. RONG, H. QIN, R. LAI, D. ZHAO, AND Q. ZENG, *Temporal high-pass filter nonuniformity correction algorithm based on guided filter for IRFPA*, 10 2015, p. 96752S. <https://doi.org/10.1117/12.2202781>.

- [3] W. QIAN, Q. CHEN, AND G. GU, *Space low-pass and temporal high-pass nonuniformity correction algorithm*, Optical Review, 17 (2010), pp. 24–29. <https://doi.org/10.1007/s10043-010-0005-8>.
- [4] R.E. RIVADENEIRA, A.D. SAPPÀ, AND B.X. VINTIMILLA, *Thermal image super-resolution: a novel architecture and dataset*, in International Conference on Computer Vision Theory and Applications, 2020, pp. 1–2. <http://dx.doi.org/10.5220/0009173601110119>.
- [5] D.A. SCRIBNER, K.A. SARKADY, J.T. CAULFIELD, M.R. KRUEER, G. KATZ, C.J. GRIDLEY, AND C. HERMAN, *Nonuniformity correction for staring IR focal plane arrays using scene-based techniques*, in Infrared Detectors and Focal Plane Arrays, vol. 1308, International Society for Optics and Photonics, SPIE, 1990, pp. 224 – 233. <https://doi.org/10.1117/12.21730>.
- [6] S. VAN DER WALT, J.L. SCHÖNBERGER, J. NÚÑEZ-IGLESIAS, F. BOULOGNE, J.D. WARNER, N. YAGER, E. GOUILLART, T. YU, AND THE SCIKIT-IMAGE CONTRIBUTORS, *scikit-image: image processing in Python*, PeerJ, 2 (2014), p. e453. <https://doi.org/10.7717/peerj.453>.
- [7] C. ZUO, Q. CHEN, G. GU, AND W. QIAN, *New temporal high-pass filter nonuniformity correction based on bilateral filter*, Optical Review, 18 (2011), pp. 197–202. <https://doi.org/10.1007/s10043-011-0042-y>.